WenQuanYi Micro Hei [Scale=0.9]WenQuanYi Micro Hei Mono song-WenQuanYi Micro Hei sfWenQuanYi Micro Hei "zh" = 0pt plus 1pt

ãĂŁLPython CookbookãĂŃçňňäÿL'çL'Ĺ

åŕŚåÿČ 2.0.0

çĘŁèČ¡

9ælJĹ 17, 2017

Contents

Contents:

Copyright

æŻt'åď'ŽåŔŚåÿČä£ąæĄŕèŕůåŔĆèĂČ
http://oreilly.com/catalog/errata.csp?isbn=9781449340377

åĽ■èÍĂ

éąźçŻőäÿżéąţ

https://github.com/yidao620c/python3-cookbook

èŕŚèĂĚçŽĎèŕİ

äžžçŤ§èŃęç§■ïijŇæĹŚçŤĺPythonïijĄ

èŕŚèĂĚäÿĂçŻť åİŽæŇĄä¡£çŤĺPython3ïijŇåŻääÿžåőČäžčèąĺäžĘPythonçŽĎæIJłæİěãĂĆèŹ¡çĎűåŘŚåŘŐ
èĂŇäÿŤPython3çŽĎæIJłæİěéIJĂèęĄæŕŔäÿłäžžçŽĎåÿőåŁl'åŠŇæŤŕæŇĄãĂĆ
çŻőåL'■åÿĆéİćäÿŁçŽĎæŢŹçĺŃäźęçś■ïijŇç¡Śäÿ£çŽĎæL'ŃåĘŇåd'ğéČĺåĹĘå§žæIJňéČ¡æŸŕ2.xçşżåĹŮçŽĎïijĺ

æIJĂè£ŚçIJŃåĹřäÿĂæIJňãĂŁPython CookbookãĂŃ3rd Edi-
tionïijŇåőŇåĚĺå§žäžŐPython3ïijŇåĘŹçŽĎäź§å¿Ĺäÿ■éŤŹãĂĆ
äÿžäžĘPython3çŽĎæŹőåŘŁïijŇæĹŚäź§äÿ■èĞłéĞŘåŁŻïijŇæČşåĄŽçĆźäżĂäźĹäžŃæČĚãĂĆäžŐæŸŕäźŐïijŃ
è£Źäÿ■æŸŕäÿĂéąźè¡żæİ¿çŽĎåůěä¡IJïijŇå■t'æŸŕäÿĂäżűåĂijå¿ŮåĄŽçŽĎåůěä¡IJïijŽäÿ■äżĚæŰźä¿£äžĘåĹńä

èŕŚèĂĚäijŽåİŽæŇĄåŕźèĞłåůśæŕŘäÿĂåŘěçŽĎç£żèŕŚèt'§èt'čïijŇåŁŻæśĆéńŸèt'ĺéĞŘãĂĆä¡ĘåŘŮèČ¡åŁŽ
åęĆæđIJèŕŚæŰĞäÿ■æIJL'äżĂäźĹéŤŹæijŘçŽĎåIJřæŰźèŕůåd'ğåőűèğĄèřËïijŇäź§æńćè£Őåd'ğåőűéŽŘæŮűæŇ
yidao620@gmail.com

ä¡IJèĂĚçŽĎèŕİ

èĞłäżŐ2008åźť äžěæİěïijŇPython3æĺłçl'žåĞžäÿŰåźűæĚćæĚćè£ŻåŇŰãĂĆPython3çŽĎætĄèąŇäÿĂçŻť
äžŃåőđäÿŁïijŇåĹřæĹŚåĘŹè£ŹæIJňäźęçŽĎ2013åźť'ïijŇçżİåd'ğéČĺåĹĘçŽĎPythonçĺŃåžŘåŚŸäż■çĎűåIJĺçŤ§
æIJĂäÿżèęĄæŸŕåŽääÿžPython3äÿ■åŘŚåŘŐåĚijåőźãĂĆæŕńæŮăçŰŚéŮőïijŇårźäžŐåůěä¡IJåIJĺéĄŮçŢŹäżčçă
ä¡ĘæŸŕæŤ¿çIJijæIJłæİěïijŇä¡ääŕšäijŽåŘŚçŐřPython3çżŹä¡ääÿęæİěäÿ■äÿĂæäůçŽĎæČŁåŰIJãĂĆ

æ■čåęĆPython3äžčèąĺæIJłæİěäÿĂæäůïijŇæŰřçŽĎãĂŁPython Cook-
bookãĂŃçL'ĹæIJńçŻÿæŕŤè¿ČäźŃåL'■çŽĎçL'ĹæIJňæIJL'äžĘäÿĂäÿłåĚĺæŰřçŽĎæŤźåŘŸãĂĆ
æIJĂéĞ■èęĄçŽĎæŸŕïijŇè£ŹäÿłæĎŘåŚşçİĂæIJňäźęæŸŕäÿĂæIJńéİđåÿÿåL'■æš£çŽĎåŘĆèĂČäźęãĂĆäźęäÿ■
åźűæšąæIJL'èĂČèŹŚäźŃåL'■èĂĄçL'ĹæIJńçŽĎåĚijåőźæĂğïijŇäź§æšąæIJL'æăĞæşĺæŮğçL'ĹæIJňäÿŃçŽĎèğ
ä¡ĘæŸŕæĹŚäžňæIJĂçżĹçŽĎçŻőçŽĎæŸŕåĘŹäÿĂæIJňåőŇåĚĺå§žäžŐæIJĂæŰřæIJĂåĚĹè£Žåůěåĕůå$Ňèŕ■èĹ
åÿŇæIJŻè£ŹæIJňäźęèČ¡æĹŘäÿžåIJĺPython3äÿŃçijŰçĂąåŠŇæČşå■ĞçžğäźŃåL'■éĄŮçŢŹäżčçăĄçŽĎçĺŃåžŘ

æŕńæŮăçŰŚéŮőïijŇçijŰåĘŹäÿĂæIJňè£ŹæăůçŽĎäźęäijŽåĘŠäÿĂåőŽçŽĎçijŰè¿ŚéčŐéŹľãĂĆåęĆæđIJå
æŕŤåęĆActiveStateâĂŹs Python recipesæĹŰèĂĚStack Over-
flowïijŇä¡ĘæŸŕçżİåd'ğéČĺåĹĘéČ¡åůšçżŘæŸŕè£ĞæŮűçŽĎäžĘãĂĆ
è£ŹäžŽæŢŹçĺŃéŹd'äžĘæŸŕå§žäžŐPython2çijŰåĘŹäźŃåd'ŰïijŇåŘŕèČ¡è£ŸæIJL'å¿Ĺåd'Žèğčå$şæŰźæąĹåIJ
åŘęåd'ŰïijŇåőČäžňè£ŸäijŽçżŘåÿÿä¡£çŤĺäÿĂäžŻè£ĞæŮűçŽĎæŁĂæIJŕïijŇè£ŹäžŽåůšçżŘåĘĚç¡őåĹřPython

è£ŹæIJňäźęçŽĎæL'ĂæIJL'äÿżécŸéČ¡æŸŕå§žäžŐåůšçżŘå■ŸåIJĺçŽĎäžčçăĄåŠŇæŁĂæIJŕïijŇèĂŃäÿ■æ
åIJĺåŐ§æIJL'äžčçăĄå§žçąĂäÿŁïijŇæĹŚäžňåőŇåĚĺä¡£çŤĺæIJĂæŰřçŽĎPythonæŁĂæIJŕåŐżæŤźéĂãĂĆ
æL'ĂäžěïijŇäżżä¡ŢæČşä¡£çŤĺæIJĂæŰřæŁĂæIJŕçijŰåĘŹäžčçăĄçŽĎçĺŃåžŘåŚŸïijŇéČ¡åŘŕäžěårĘæIJňäźęå¡Ş

åIJĺèőĺèőžçŽĎäÿżécŸéĂL'æŃl'æŰźéİćïijŇæĹŚäžňäÿ■åŘŕèČ¡åŻŁæŃňPythonécĘå§§æL'ĂæIJL'çŽĎäÿIJ
åŻäæ■d'ïijŇæĹŚäžňäijŸåĚĹéĂL'æŃl'äžĘPythonèŕ■èĺĂæäÿå£ČéČĺåĹĘïijŇäžěåŘŁäÿĂäžŻåIJĺåijĂåŘŚäÿ■åÿ
åŘęåd'ŰïijŇè£ŹéĞŃèőĺèőžçŽĎå¿Ĺåd'ŽæŁĂæIJŕéČ¡æŸŕPython
3æIJĂæŰřæL'■åĞžçŐřçŽĎïijŇæL'ĂäžěåęĆæđIJåůěä¡IJåIJĺPythonèĂĄçL'ĹæIJňäÿŃïijŇ
å■şä¿£æŸŕæIJĂæIJL'çżŔéłŇçŽĎçĺŃåžŘåŚŸåŘŕèČ¡äźŃåL'■äź§æšąèğĄè£Ğè£ŹäžŻäÿIJèě£ãĂĆ
åŘęåd'ŰïijŇè£ŹäžŻçd'ža¿ŃçĺŃåžŘäź§äijŽåĄŘåŘŚäžŐåśŢçd'žäÿĂäžŻæIJL'çŤĺçŽĎçijŰçĺŃæŁĂæIJŕ(æŕŤåę
èĂŃäÿ■æŸŕäżĚäżĚåőŽä¡■åIJĺäÿĂäžŻåĚůä¡ŞçŽĎéŮőécŸäÿŁãĂĆår¡çőąäź§æŘŘåŘŁåĹřäžĘæIJL'äÿĂäžŻçň

è£ŹæIJňäźęéĂĆåŘĹèřĄ

è£ŹæIJňäźęçŽĎçŻőæăĞèŕżèĂĚæŸŕéĆčäžŻæČşæůśåĚěçŘĘèğčPythonèŕ■èĺĂæIJžåĹűåŠŇæIJĂæŰřçijŰ
å¿Ĺåď Žèőĺèőžéč¡æŸŕæăĞåĞĘåžŞïijŇæąĘæđűåŠŇåžŤçŤĺçĺŃåžŔä¡£çŤĺåĹřçŽĎéńŸçžğæŁĂæIJŕãĂĆ
æIJňäźęæĽĂæIJĽçď žä¿ŃåİĞåĄĞèő¿èŕżèĂĚåůšçżŔæIJĽäžĘäÿĂåőŽçŽĎçijŰçĺŃèČŇæŹŕåźűäÿŤåŔŕäžěå¿Ĺ
(æŕŤåęĆå§žæIJňçŽĎèőąçőŮæIJžçğŚå■ęç§ěèŕĘïijŇæŢřæ■őçżŞæđĎç§ěèŕĘïijŇçőŮæşŢåď■æİĆåžęïijŇçşżçż§
åŘęåď Űïij ŇæŕŔäÿłçď žä¿ŃéČ¡åŔłæŸŕäÿĂäÿłåĚěéŮĺæŇĞåŕijïijŇåęĆæđIJèŕżèĂĚæČşæůśåĚěçăŤçľűïijŇéIJĂ
åŻăæ■d'ïijŇæĹŚäžňåĄĞåőŽèŕżèĂĚåŔŕäžěå¿ĹçĘ§çżČçŽĎä¡£çŤĺæŘIJçť ćåijŢæŞŐäžěåŘŁç§ěéĄŞæĂŐæăůæ

è£ŹæIJňäźęäÿ■éĂĆåŘĹPythonçŽĎåĹİå■ęèĂĚãĂĆäžŃåðđäÿŁïijŇæIJňäźęåůšçżŘåĄĞåőŽäžĘèŕżèĂĚåů
æIJňäźęäź§äÿ■æŸŕéČčçğ■å£ńéĂ§åŔĆèĂČæĽŃåĘŃ(åŔŕäžěå¿Ĺå£ńçŽĎæ§ěèŕćæ§Řäÿłæĺąå İŮäÿŃçŽĎæ§Řäÿ
æIJňäźęæŮĺåIJĺèĄŽçĎęåĞääÿłæIJĂéĞ■èęĄçŽĎäÿżéćŸïijŇæijŤçď žåĞăçğ■åŔŕèČ¡çŽĎèğčåĘşæŰźæąĹïijŇä
ä¡ăåŔŕäžěçżŔæ■d'è£ŻåĚěäÿĂäžŻæŻt'énŸçžğçŽĎäÿżéćŸïijŇè£ŹäžŻåŔŕäžěåIJĺç¡Śäÿ ŁæĹŰèĂĚåŔĆèĂČæĽ'

æIJňäźęçď žä¿ŃäžčçăĄ

æIJňäźęåĞăäźŐæĽ'ĂæIJĽ'æžŘäżčçăĄåİĞåŔŕäžěåIJĺ http://github.com/dabeaz/python-cookbook äÿŁéİćæĽ'¿åĹřãĂĆ ä¡IJèĂĚæňćè£ŐåŘĎä¡■ä£őæ■čbugïijŇæŤźè£Żäžčçă ĄåŠŇèŕĎèőžãĂĆ

æIJňäźęåřśæŸŕåÿőåŁl'ä¡ăåőŇæĹŘä¡ăçŽĎåůěä¡IJãĂĆäÿĂèĹňæİèèőšïijŇåŔłèęĄåIJĺæIJňäźęäÿŁéİćçŽĎå
ä¡ăéČ¡åŔŕäžěéŽŘæŮűæŃ£è£ĞåŐżåIJĺä¡ăçŽĎæžŘçĂĄåŠŇæŰĞæąčäÿ■ä¡£çŤĺãĂĆä¡ăäÿ■éIJĂèęĄåŘŚæĹŚäž
éŹď éİđä¡ăæŁĎèć■çŽĎåď łè£ĞåĹĘäžĘãĂĆæŕŤåęĆèŕť åď ■åĹűåĞăäÿłäžčçăĄçĽĞæőţåŐżåőŇæĹŘäÿĂäÿłçĺŃ
èť'l'å■ŰæĹŰèĂĚåĹĘåŘŚåðđä¿ŃäžčçăĄçŽĎåĚĽçŻŸäź§äÿ■éIJĂèęĄèőÿåŔŕïijŇåijŢçŤĺæIJňäźęåŠŇåðđä¿Ńä
ä¡ĘæŸŕïijŇåŘĹåźűåď ğéĞŘçŽĎäžčçăĄåĹŕä¡ăçŽĎæ■čåijŔäžğåŞĄæĹŰæŰĞæąčäÿ■åŐżå£Ěéążå¿ŮåĹřæĹŚä

æĹŚäžňäÿ■äijŽèęĄæśĆä¡ăæůżåŁääžčçăĄçŽĎåĞžåď ĎïijŇåŇĚæŃňæăĞéćŸïijŇä¡IJèĂĚïijŇåĞžçĽĹçď
æŕŤåęĆïijŽPython Cookbook, 3rd edition, by David Beazley and Brian K. Jones (OâĂŹReilly). Copyright 2013 David Beazley and Brian Jones, 978-1-449-34037-7.
ä¡ĘæŸŕåęĆæđIJä¡äè£ŹäźĹåĄŽäžĘïijŇæĹŚäžňäijŽå¿ĹæĎ§æ£ĂçŽĎãĂĆ

èĄŤçşżæĹŚäžň

èŕůåřĘåĚşäžŐæIJňäźęçŽĎèŕĎèőžåŠŇéŮőécŸåŔŚéĂĄçżŹåĞžçĽĹçď¿ïijŽ

OâĂŹReilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

æIJňäźęç¡ŚçńŹ: http://oreil.ly/python_cookbook_3e ïijŇäÿŁéİćæIJL'åŃŸèŕŕèąĺïijŇçď'žä¿ŃåŠŇäÿĂäžŽå

åęĆæđIJæČşèęĄèŕĎèőžæĹŰèĂĚæŸŕéŮőäÿĂäÿŃæIJňäźęæŁĂæIJŕæŰźéİćçŽĎéŮőécŸïijŇ
èŕůåŔŚéĂĄéĆőäžűèĞşïijŽ bookquestions@oreilly.com

æŻť'åď'ŽåĚşäžŐæĹŚäżňçŽĎäźęçś■ïijŇèőĺèőžäijŽïijŇæŰřéŮżïijŇ
èŕůèő£éŮőæĹŚäżňçŽĎç¡ŚçńŹïijŽ http://www.oreilly.com

åIJÍFacebookäÿŁæ§ěæL'¿æĹŚäżň: http://facebook.com/oreilly

åIJÍTwitteräÿŁåĚşæşĺæĹŚäżň: http://twitter.com/oreillymedia

åIJÍYouTubeäÿŁèğĆçIJŃæĹŚäżň: http://www.youtube.com/oreillymedia

æĎ§èřć

æĹŚäżňçŤśèąůçŽĎæĎ§èřćæIJňäźęçŽĎæŁĂæIJŕåőąæăÿèĂĚJake Vanderplas, Robert Kern åŠŇ Andrea CrottiçŽĎéİđåÿÿæIJL'çŤĺçŽĎèŕĎèőžåŠŇåżżèőőïijŇ è£ŸæIJL'Pythonçď'¿åŇžçŽĎåÿőåŁl'åŠŇéijŞåŁśãĂĆæĹŚäżňè£ŸæČşæĎ§èřćäÿŁäÿĂäÿłçL'ĹæIJňçŽĎçijŰè¿Ś Vanderplas, Robert Kern,and Andrea CrottiãĂĆ år¡çőąè£ŹäÿłçL'ĹæIJňæŸŕæIJĂæŰřçŽĎïijŇä¡ĘæŸŕåL'■äÿĂä æIJĂåŘŐïijŇæIJĂæIJĂéĞ■èęĄçŽĎåřśæŸŕïijŇæĹŚäżňèęĄæĎ§èřćæL'ĂæIJL'écĎèğĹçL'ĹæIJňçŽĎèŕżèĂĚïij

çňňäÿĂçńăïijŽæŢřæ■őçżŞæđĎåŠŇçőŮæşŢ

PythonæŔŘä¿ŻäžĘåď'ğéĞŔçŽĎåĘĚç¡őæŢřæ■őçżŞæđĎïijŇåŇĚæŃňåĹŮèąĺïijŇéŻĘåŘĹäżěåŔŁå■ŮåĚÿ ä¡ĘæŸŕïijŇæĹŚäżňäź§äijŽçżŔåÿÿçćřåĹřåĹřèŕÿåęĆæ§ěèŕćïijŇæŐŠåžŔåŠŇè£ĞæżďçL'ç■L'è£ŹäžŻæŹőéĄ■ åŽăæ■d'ïijŇè£ŹäÿĂçńăçŽĎçŻőçŽĎåřśæŸŕèőĺèőžè£ŹäžŻæŕŤè¿Čåÿÿèğ ĄçŽĎéŮőécŸåŠŇçőŮæşŢãĂĆ åŘęåď'ŰïijŇæĹŚäżňäź§äijŽçżŹåĞžåIJĺéŻĘåŘĹæĺąåİŮ `collections` å¡Şäÿ■æŞ■ä¡IJè£ŹäžŻæŢřæ■őçżŞæđĎçŽĎæŰźæşŢãĂĆ

Contents:

1.1 èğčåŐŃåžŔåĹŮèţŃåĂijçżŹåď'Žäÿłå ŔŸéĞR

éŮőécŸ

çŐřåIJĺæIJL'äÿĂäÿłåŇĚåŘńNäÿłåĚČçt'ăçŽĎåĚČçżĎæĹŰèĂĚæŸŕåžŔåĹŮïijŇæĂŐæăůårĘåőČéĞŇéİć

èğčåĘşæŰźæąĹ

äžżä¡ŢçŽĎåžŔåĹŮ(æĹŰèĂĚæŸŕåŔŕè£■äžčårźèśą)åŔŕäžěéĂŽè£ĞäÿĂäÿłçőĂå■ŢçŽĎèţŃåĂijèŕ■åŔěèğ åŤŕäÿĂçŽĎåL'■æŔŘårśæŸŕåŔŸéĞŔçŽĎæŢřéĞŔå£Ěéążèů§åžŔåĹŮåĚČçt'ăçŽĎæŢřéĞŔæŸŕäÿĂæăůçŽĎã

äžčçăĄçď'žä¿ŃïijŽ

```
>>> p = (4, 5)
>>> x, y = p
>>> x
4
>>> y
5
>>>
>>> data = [ 'ACME', 50, 91.1, (2012, 12, 21) ]
>>> name, shares, price, date = data
>>> name
'ACME'
>>> date
(2012, 12, 21)
>>> name, shares, price, (year, mon, day) = data
>>> name
'ACME'
>>> year
2012
>>> mon
12
>>> day
21
>>>
```

åęĆædIJåŔŸéĞŔäÿłæŢřåŠŇåžŔåĹŮåĚČçť ăçŽĎäÿłæŢřäÿ■åŇźéĚ■ïijŇäijŽäžğçŤ§äÿĂäÿłåijĆåÿÿãĂĆ äżčçăĄçď žä¿ŃïijŽ

```
>>> p = (4, 5)
>>> x, y, z = p
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: need more than 2 values to unpack
>>>
```

èőÍèőž

åőđéŹĚäÿŁïijŇè£Źçğ■èğčåŐŃèţŃåĂijåŔŕäżěçŤĺåIJÍäżżä¡ŢåŔŕè£■äżčåŕźèśąäÿŁéİćïijŇèĂŇäÿ■äżĚäżĚæåŇĚæŃňå■Ůçňęäÿšïij ŇæŰĞäžűåŕźèśąïijŇè£■äżčåŹĺåŠŇçŤ§æĹŘåŹĺãĂĆ

äżčçăĄçď žä¿ŃïijŽ

```
>>> s = 'Hello'
>>> a, b, c, d, e = s
>>> a
'H'
>>> b
'e'
>>> e
```

```
'o'
>>>
```

æIJL'æŮűåĂŹïijŇä¡ăåŔŕèČ¡åŔłæČşèğčåŐŃäÿĂéČĺåĹĘïijŇäÿćåijČåĚűäžŰçŽĎåĂijãĂĆåŕźäžŐè£Źçğ■a
ä¡ĘæŸŕä¡ăåŔŕäžěä¡£çŤĺäżżæĎŔåŔŸéĞŔåŘ■åŐżå■ăä¡■ïijŇåĹřæŮűåĂŹäÿćæŐL'è£ŹäžŻåŔŸéĞŔåřśèąŇäžĘ

äžčçăĄçď'žä¿ŃïijŽ

```
>>> data = [ 'ACME', 50, 91.1, (2012, 12, 21) ]
>>> _, shares, price, _ = data
>>> shares
50
>>> price
91.1
>>>
```

ä¡ăå£Ěéążä£ÌèŕĄä¡ăéĂL'çŤĺçŽĎéĆčäžŻå■ăä¡■åŔŸéĞŔåŘ■åIJĺåĚűäžŰåIJřæŰźæšąèćńä¡£çŤĺåĹřãĂĆ

1.2 èğčåŐŃåŔŕè£■äžčåŕźèśąèţŃåĂijçżŹåď'ŽäÿłåŔŸéĞŔ

éŮőécŸ

åęĆæđIJäÿĂäÿłåŔŕè£■äžčåŕźèśąçŽĎåĚČçt'ăäÿłæŢřèűĚè£ĞåŔŸéĞŔäÿłæŢřæŮűïijŇäijŽæŁŻåĞžäÿĂäÿł
`ValueError` ãĂĆ éĆčäźĹæĂŐæăůæL'■èČ¡äžŐè£ŹäÿłåŔŕè£■äžčåŕźèśąäÿ■èğčåŐŃåĞžNäÿłåĚČçt'ăåĞžæÌě

èğčåĘşæŰźæąĹ

PythonçŽĎæŸ§åŔůèąÍè¿¿åijŔåŔŕäžěçŤĺæÌèèğčåĘşè£ŹäÿłéŮőécŸãĂĆærŤåęĆïijŇä¡ăåIJĺå■ęäźääÿĂéŮĺ
ä¡ăæČşçż§èőąäÿŃåőűåž■ä¡IJäÿŽçŽĎåźşåİĞæĹŘçżl'ïijŇä¡ĘæŸŕæŐŠéŹď'æŐL'çňňäÿĂäÿłåŠŇæIJĂåŘŐäÿĂä
ä¡ĘåęĆæđIJæIJL'24äÿłåŚćïij§è£ŹæŮűåĂŹæŸ§åŔůèąÍè¿¿åijŔåřśæť¿äÿŁçŤĺåIJžäžĘïijŽ

```
def drop_first_last(grades):
    first, *middle, last = grades
    return avg(middle)
```

åŘęåď'ŰäÿĂçğ■æČĚåĘţïijŇåĄĞèő¿ä¡ăçŐřåIJĺæIJL'äÿĂäžŻçŤĺæĹůçŽĎèőřå¡ŢåĹŮèąÍïijŇærŔæİąèőřå¡Ţ
ä¡ăåŔŕäžěåČŔäÿŃéİćè£ŹæăůåĹĘèğčè£ŹäžŻèőřå¡ŢïijŽ

```
>>> record = ('Dave', 'dave@example.com', '773-555-1212', '847-555-
↪1212')
>>> name, email, *phone_numbers = record
>>> name
'Dave'
>>> email
'dave@example.com'
>>> phone_numbers
['773-555-1212', '847-555-1212']
>>>
```

åĂijå¿Ůæşĺæ ĎŔçŽĎæŸŕäÿŁéİćèğčåŐŃåĞžçŽĎ phone_numbers
åŔŸéĞŔæřÿè£IJéČ¡æŸŕåĹŮèąĺçśżåđŃïijŇäÿ■çőąèğčåŐŃçŽĎçŢţèŕİåŔůçăĄæŢřéĞŔæŸŕåď ŽåřŚ(åŇĚæŃň0
æĽĂäżěïijŇäżżä¡Ţä¡£çŤĺåĹř phone_numbers åŔŸéĞŔçŽĎäžčçăĄåřśäÿ■éIJĂèęĄåĄŽåď Žä¡ŹçŽĎçśżåđŃ

æŸ§åŔůèąĺè¿¿åijŔäź§èČ¡çŤĺåIJĺåĹŮèąĺçŽĎåijĂåğŃéČĺåĹĘãĂĆærŤåęĆïijŇä¡ăæIJL'äÿĂäÿłåĚňåŔÿåL'■
ä¡ĘæŸŕä¡ăæČşçIJŃäÿŃæIJĂè£ŚäÿĂäÿłæIJĹæŢřæ■őåŠŇåL'■éİć7äÿłæIJĹçŽĎåźşåİĞåĂijçŽĎåŕźærŤãĂĆä¡ăå

```
*trailing_qtrs, current_qtr = sales_record
trailing_avg = sum(trailing_qtrs) / len(trailing_qtrs)
return avg_comparison(trailing_avg, current_qtr)
```

äÿŃéİćæŸŕåIJÍPythonèğčéĞŁåŹĺäÿ■æL'ğèąŇçŽĎçżŞæđIJïijŽ

```
>>> *trailing, current = [10, 8, 7, 1, 9, 5, 10, 3]
>>> trailing
[10, 8, 7, 1, 9, 5, 10]
>>> current
3
```

èőÍèőž

æL'ľåśŢçŽĎè£■äžčèğčåŐŃèŕ■æşŢæŸŕäÿŞéŮĺäÿžèğčåŐŃäÿ■çąőåőŽäÿłæŢřæĹŰäżżæĎŔäÿłæŢřåĚČçť'ă
éĂŽåÿÿïijŇè£ŹäžŽåŔŕè£■äžčåŕźèśąçŽĎåĚČçť'ăçżŞæđĎæIJL'çąőåőŽçŽĎèğĎåĹŹïijĹæŕŤåęĆçňň1äÿłåĚČçť'ă
æŸ§åŔůèąĺè¿¿åijŔèőľåijĂåŔŚäžžåŚŸåŔŕäžěå¿Ĺåőźæ ŸŞçŽĎåĹľçŤĺè£ŹäžŽèğĎåĹŹæİèèğčåŐŃåĞžåĚČçť'ă
èĂŇäÿ■æŸŕéĂŽè£ĞäÿĂäžŽærŤè¿Čåď'■æİĆçŽĎæL'ŃæőţåŏżèŏůåŔŰè£ŹäžŽåĚşèĄŢçŽĎçŽĎåĚČçť'ăåĂijã

åĂijå¿Ůæşĺæ ĎŔçŽĎæŸŕïijŇæŸ§åŔůèąĺè¿¿åijŔåIJĺè£■äžčåĚČçť'ääÿžåŕŕåŔŸéŢ£åĚČçżĎçŽĎåžŔåĹŮæ
ærŤåęĆïijŇäÿŃéİćæŸŕäÿĂäÿłåÿęæIJL'æăĞç■¿çŽĎåĚČçżĎåžŔåĹŮïijŽ

```
records = [
    ('foo', 1, 2),
    ('bar', 'hello'),
    ('foo', 3, 4),
]

def do_foo(x, y):
    print('foo', x, y)

def do_bar(s):
    print('bar', s)

for tag, *args in records:
    if tag == 'foo':
        do_foo(*args)
    elif tag == 'bar':
        do_bar(*args)
```

æŸ§åŔůèğčåŐŃèŕ■æşŢåIJĺå■ŮçňęäÿšæŞ■ä¡IJçŽĎæŮűåĂŹäź§äijŽå¿ĹæIJL'çŤĺïijŇærŤåęĆå■Ůçňęäÿšç

äžčçăĄçď'žä¿ŃïijŽ

```
>>> line = 'nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/
↪false'
>>> uname, *fields, homedir, sh = line.split(':')
>>> uname
'nobody'
>>> homedir
'/var/empty'
>>> sh
'/usr/bin/false'
>>>
```

æIJL'æŮűåĂŹïijŇä¡ăæČşèğčåŐŃäÿĂäžŻåĚČçt'ăåŘŐäÿćåijČåőČäżňïijŇä¡ăäÿ■èČ¡çőĂå■Ţářśä¡£çŤÍ * ïijŇ ä¡ĘæŸŕä¡ăåŔŕäżěä¡£çŤĺäÿĂäÿłæŹőéĂŽçŽĎåž§åijČåŘ■çğřïijŇæŕŤåęĆ _ æĹŰèĂĚ ign ãĂĆ

äżčçăĄçd'žä¿ŃïijŽ

```
>>> record = ('ACME', 50, 123.45, (12, 18, 2012))
>>> name, *_, (*_, year) = record
>>> name
'ACME'
>>> year
2012
>>>
```

åIJĺå¿Ĺåd'ŽåĞ¡æŢřåijŔèŕ■èĺĂäÿ■ïijŇæŸ§åŔůèğčåŐŃèŕ■æşŢèů§åĹŮèąĺåd'ĎçŘĘæIJL'èőÿåd'ŽçŻÿäijijä ä¡ăåŔŕäżěå¿Ĺåőźæ ŸŞçŽĎåřĘåőČåĹĘåL'šæĹŘåL'■åŘŐäÿd'éČĺåĹĘïijŽ

```
>>> items = [1, 10, 7, 4, 5, 9]
>>> head, *tail = items
>>> head
1
>>> tail
[10, 7, 4, 5, 9]
>>>
```

åęĆædIJä¡ăåd'§èĄłæŸŐçŽĎèŕİïijŇè£ŸèČ¡çŤĺè£Źçğ■åĹĘåL'šèŕ■æşŢåŐżåůğåęŹçŽĎåőđçŐřéĂŠå¡Šçő

```
>>> def sum(items):
... head, *tail = items
... return head + sum(tail) if tail else head
...
>>> sum(items)
36
>>>
```

çĎűåŘŐïijŇçŤśäžŐèŕ■èĺĂåśĆéİćçŽĎéŹŘåĹűïijŇéĂŠå¡Šåźűäÿ■æŸŕPythonæŞĚéŢ£çŽĎãĂĆ åŻăæ■d'ïijŇæIJĂåŘŐéĆčäÿłéĂŠå¡ŠæijŤçd'žäżĚäżĚæŸŕäÿłåě¡åěĞçŽĎæŐćçt'ćç¡ćäžĘïijŇåŕźè£Źäÿłäÿ■èęĄå

1.3 ä£ÌçŢŹæIJĂåŘŐNäÿłåĚČçťă

éŮőécŸ

åIJÍè£■äžčæŞ■ä¡IJæĹŰèĂĚåĚűäžŰæŞ■ä¡IJçŽĎæŮűåĂŹïijŇæĂŐæăůåŘłä£ÌçŢŹæIJĂåŘŐæIJL'éŹŘåĞă

èğčåĘşæŰźæąĹ

ä£ÌçŢŹæIJL'éŹŘåŐĘåŘšèőřå¡Ţæ■čæŸŕ `collections.deque` åď ğæŸ¿èžńæL'ŃçŽĎæŮűåĂŹãĂĆæŕŤåęĆïijŇäÿŃéÌćçŽĎäžčçăĄåIJÍåď'ŽèąŇäÿŁéÌćåĄŽçőĂå■ŢçŽĎæŰĞæ åźűè£ŤåŻđåŇźéĚ■æL'ĂåIJÍèąŇçŽĎæIJĂåŘŐNèąŇïijŽ

```
from collections import deque


def search(lines, pattern, history=5):
    previous_lines = deque(maxlen=history)
    for li in lines:
        if pattern in li:
            yield li, previous_lines
        previous_lines.append(li)

# Example use on a file
if __name__ == '__main__':
    with open(r'../../cookbook/somefile.txt') as f:
        for line, prevlines in search(f, 'python', 5):
            for pline in prevlines:
                print(pline, end='')
            print(line, end='')
            print('-' * 20)
```

èőÍèőž

æĹŚäžňåIJÍåĘŹæ§ěèŕćåĚČçť'äçŽĎäžčçăĄæŮűïijŇéĂŽåÿÿäijŽä¡£çŤÍåŇĚåŘń `yield` èąÍè¿¿åijŔçŽĎçŤ§æĹŘåŹÍåĞ¡æŢřïijŇäź§åřśæŸŕæĹŚäžňäÿŁéÌćçď'žä¿ŃäžčçăĄäÿ■çŽĎéĆčæăůãĂĆ è£ŹæăůåŘŕäžěåřĘæŘIJçť'ćè£ĞçÍŃäžčçăĄåŠŇä¡£çŤÍæŘIJçť'ćçžŞædIJäžčçăĄèğčèĂęãĂĆåęĆædIJä¡ăè£Ÿäÿ■a

ä¡£çŤÍ `deque(maxlen=N)` ædĎéĂăåĞ¡æŢřäijŽæŰřåžžäÿĂäÿłåŽžåőŽåď'ğåřŔçŽĎéŸ§åĹŮãĂĆå¡ŞæŰ æIJĂèĂĄçŽĎåĚČçť'ääijŽèĞłåŁÍèćńçğžéŹď'æŐL'ãĂĆ

äžčçăĄçď'žä¿ŃïijŽ

```
>>> q = deque(maxlen=3)
>>> q.append(1)
>>> q.append(2)
>>> q.append(3)
>>> q
deque([1, 2, 3], maxlen=3)
```

```
>>> q.append(4)
>>> q
deque([2, 3, 4], maxlen=3)
>>> q.append(5)
>>> q
deque([3, 4, 5], maxlen=3)
```

åř¡çőąä¡ääź§åŔŕäžěæĽŃåŁĺåIJĺäÿĂäÿłåĹŮèąĺäÿŁåőđçŐřè£ŹäÿĂçŽĎæŞ■ä¡IJ(æŕŤåęĆåćđåŁããĂĄåĹäéŽ

æŻť'äÿĂèĹňçŽĎïijŇ deque çśžåŔŕäžěèćńçŤĺåIJĺäżżä¡Ţä¡ääŔłéIJĂèęĄäÿĂäÿłçőĂå■ŢéŸ§åĹŮæŢřæ■őç åęĆæđIJä¡ääÿ■èő¿ç¡őæIJĂåď'ğéŸ§åĹŮåď'ğåřŔïijŇéĆčäźĹåřśäijŽå¿ŮåĹřäÿĂäÿłæŮäéŹŘåď'ğåřŔéŸ§åĹŮïijŇ

äžčçăĄçď'ža¿ŃïijŽ

```
>>> q = deque()
>>> q.append(1)
>>> q.append(2)
>>> q.append(3)
>>> q
deque([1, 2, 3])
>>> q.appendleft(4)
>>> q
deque([4, 1, 2, 3])
>>> q.pop()
3
>>> q
deque([4, 1, 2])
>>> q.popleft()
4
```

åIJĺéŸ§åĹŮäÿd'çńŕæŔŠåĚěæĹŰåĹăéŹd'åĚČçt'ăæŮűéŮt'åd'■æİĆåžęéČ¡æŸŕ O(1) ïijŇèĂŇåIJĺåĹŮèąĺçŽĎåijĂåd't'æŔŠåĚěæĹŰåĹăéŹd'åĚČçt'ăçŽĎæŮűéŮt'åd'■æİĆåžęäÿž O(N) ãĂĆ

1.4 æ§ěæĽ¿æIJĂåď'ğæĹŰæIJĂåřŔçŽĎNäÿłåĚČçt'ă

éŮőécŸ

æĂŐæăůäžŐäÿĂäÿłéŻĘåŘĹäÿ■èŐůå¿ŮæIJĂåď'ğæĹŰèĂĚæIJĂåřŔçŽĎNäÿłåĚČçt'ăåĹŮèąĺïij§

èğčåĘşæŰźæąĹ

heapqæĺąåİŮæIJL'äÿd'äÿłåĞ¡æŢřijŽnlargest() åŠŇ nsmallest() åŔŕäžěåőŇç¿Őèğčåĕşè£ŹäÿłéŮőécŸãĂĆ

```
import heapq
nums = [1, 8, 2, 23, 7, -4, 18, 23, 42, 37, 2]
```

```
print(heapq.nlargest(3, nums)) # Prints [42, 37, 23]
print(heapq.nsmallest(3, nums)) # Prints [-4, 1, 2]
```

äÿd'äÿłåĞ¡æŢřéČ¡èČ¡æŐěåŔŮäÿĂäÿłåĚşéŤőå■ŮåŔĆæŢřïijŇçŤĺäžŐæŻt'åd'■æİĆçŽĎæŢřæ■őçżŞædĎä

```
portfolio = [
    {'name': 'IBM', 'shares': 100, 'price': 91.1},
    {'name': 'AAPL', 'shares': 50, 'price': 543.22},
    {'name': 'FB', 'shares': 200, 'price': 21.09},
    {'name': 'HPQ', 'shares': 35, 'price': 31.75},
    {'name': 'YHOO', 'shares': 45, 'price': 16.35},
    {'name': 'ACME', 'shares': 75, 'price': 115.65}
]
cheap = heapq.nsmallest(3, portfolio, key=lambda s: s['price'])
expensive = heapq.nlargest(3, portfolio, key=lambda s: s['price'])
```

èŕŚèĂĚæşĺïijŽäÿŁéİćäżčçăĄåIJĺåŕźæŕŔäÿłåĚČçt'äè£ŻèąŇåŕźæŕŤçŽĎæŮűåĂŹïijŇäijŽäžě
`price` çŽĎåĂijè£ŻèąŇæŕŤè¿ČãĂĆ

èőĺèőž

åęĆæđIJä¡ăæČşåIJĺäÿĂäÿłéŻĘåŘĹäÿ■æ§ěæL'¿æIJĂåřŔæĹŰæIJĂåd'ğçŽĎNäÿłåĚČçt'äïijŇåźűäÿŤNåřŔ
åŻääÿžåIJĺåžŢåśĆåőđçŐřéĞŇéİćïijŇéęŰåĚĹäijŽåĚĹåřĘéŻĘåŘĹæŢřæ■őè£ŻèąŇääĘæŐŠåžŔåŘŐæŤ¿åĚěäÿ

```
>>> nums = [1, 8, 2, 23, 7, -4, 18, 23, 42, 37, 2]
>>> import heapq
>>> heapq.heapify(nums)
>>> nums
[-4, 2, 1, 23, 7, 2, 18, 23, 42, 37, 8]
>>>
```

åäĘæŢřæ■őçżŞædĎæIJĂéĞ■èęĄçŽĎçL'źå¿ĄæŸŕ `heap[0]`
æřÿè£IJæŸŕæIJĂåřŔçŽĎåĚČçt'ããĂĆåźűäÿŤåL'l'ä¡ŹçŽĎåĚČçt'äåŔŕäžěå¿ĹåőźæŸ§çŽĎéĂŽè£ĞèřČçŤĺ
`heapq.heappop()` æŰźæşŢå¿ŮåĹřïijŇ èřěæŰźæşŢäijŽåĚĹåřĘçňňäÿĂäÿłåĚČçt'äåijźåĞžæİèïijŇçĎűåŘŐ
N)ïijŇNæŸŕääĘåd'ğåřŔ)ãĂĆ æŕŤåęĆïijŇåęĆædIJæČşèęĄæ§ěæL'¿æIJĂåřŔçŽĎ3äÿłåĚČçt'äïijŇä¡äåŔŕäžěè£Z

```
>>> heapq.heappop(nums)
-4
>>> heapq.heappop(nums)
1
>>> heapq.heappop(nums)
2
```

å¡ŞèęĄæ§ěæL'¿çŽĎåĚČçt'ääÿłæŢřçŻÿåŕźæŕŤè¿ČåřŔçŽĎæŮűåĂŹïijŇåĞ¡æŢř
`nlargest()` åŠŇ `nsmallest()` æŸŕå¿ĹåŘĹéĂĆçŽĎãĂĆ
åęĆædIJä¡ääžĚäžĚæČşæ§ěæL'¿åŤŕäÿĂçŽĎæIJĂåřŔæĹŰæIJĂåd'ğ(N=1)çŽĎåĚČçt'äçŽĎèŕİïijŇéĆčäźĹä¡£çŤ
`min()` åŠŇ `max()` åĞ¡æŢřäijŽæŻt'å£ńäžŽãĂĆ çśżäijijçŽĎïijŇåęĆædIJNçŽĎåd'ğåřŔåŠŇéŻĘåŘĹåd'ğåřŔæ
(`sorted(items)[:N]` æĹŰèĂĚæŸŕ `sorted(items)[-N:]`
)ãĂĆ éIJĂèęĄåIJĺæ■čçąőåIJžåŘĹä¡£çŤĺåĞ¡æŢř `nlargest()`

åŠŇ `nsmallest()` æĽ■èČ¡åŔŚæŇěåőČäžňçŽĎäijŸåŁ£(åęĆæđIJNå£ńæŐěè£ŚéŻĘåŘĹåď˘gåřŔäžĘïijŇéĆčäźĹä¡£çŤĺæŐŠåžŔæŞ■ä¡IJäijŽæŻť˘åě¡äžŻ)ãĂĆ

åř¡çőąä¡ăæšąæIJL˘å£ĚèęĄäÿĂåőŽä¡£çŤĺè£ŹéĞŇçŽĎæŰźæşŢïijŇä¡ĘæŸŕåăĘæŢřæ■őçżŞæđĎçŽĎåőđçĆå§žæIJñäÿŁåŔłèęĄæŸŕæŢřæ■őçżŞæđĎåŠŇçőŮæşŢäźęçś■éĞŇéİćéČ¡äijŽæIJL˘æŔŘåŔŁåĹřãĂĆ `heapq` æĺąåİŮçŽĎåőŸæŰźæŰĞæąčéĞŇéİcäź§èŕęçżĘçŽĎäżŇçż■äžĘåăĘæŢřæ■őçżŞæđĎåžŢåśĆçŽĎåőđçĆ

1.5 åőđçŐřäÿĂäÿłäijŸåĚĹçžğéŸ§åĹŮ

éŮőécŸ

æĂŐæăůåőđçŐřäÿĂäÿłæŇL˘äijŸåĚĹçžğæŐŠåžŔçŽĎéŸ§åĹŮïij§åźűäÿŤåIJĺè£ŹäÿłéŸ§åĹŮäÿŁéİcæŕŔæňąpopæŞ■ä¡IJæĂżæŸŕè£ŤåŽđäijŸåĚĹçžğæIJĂénŸçŽĎéĆčäÿłåĚČçt˘

èğčåĘşæŰźæąĹ

äÿŃéİćçŽĎçśżåĹ˘çŤĺ `heapq` æĺąåİŮåőđçŐřäžĘäÿĂäÿłçőĂå■ŢçŽĎäijŸåĚĹçžğéŸ§åĹŮïijŽ

```
import heapq

class PriorityQueue:
    def __init__(self):
        self._queue = []
        self._index = 0

    def push(self, item, priority):
        heapq.heappush(self._queue, (-priority, self._index, item))
        self._index += 1

    def pop(self):
        return heapq.heappop(self._queue)[-1]
```

äÿŃéİćæŸŕåőČçŽĎä¡£çŤĺæŰźåijŔïijŽ

```
>>> class Item:
...     def __init__(self, name):
...         self.name = name
...     def __repr__(self):
...         return 'Item({!r})'.format(self.name)
...
>>> q = PriorityQueue()
>>> q.push(Item('foo'), 1)
>>> q.push(Item('bar'), 5)
>>> q.push(Item('spam'), 4)
>>> q.push(Item('grok'), 1)
>>> q.pop()
Item('bar')
>>> q.pop()
```

```
Item('spam')
>>> q.pop()
Item('foo')
>>> q.pop()
Item('grok')
>>>
```

äžŤçžĘèğĆåŕ§åŔŕäžěåŔŚçŐřïijŇçňňäÿĂäÿł `pop()` æŞ■ä¡IJè£ŤåŻđäijŸåĚĹçžğæIJĂéńŸçŽĎåĚČçť'ããĂ
åŔęåđ'ŰæşĺæĎŔåĹřåęĆæđIJäÿđ'äÿłæIJL'çİĂçŻÿåŘŇäijŸåĚĹçžğçŽĎåĚČçť'ă(`foo` åŠŇ `grok`)ïijŇpopæŞ■ä¡IJæŇL'çĚğåőČäžňèćńæŔŠåĚěåĹřéŸ§åĹŮçŽĎéąžåžŔè£ŤåŻđçŽĎãĂĆ

èőĺèőž

è£ŹäÿĂåřŔèŁĆæĹŚäžňäÿżèęĄåĚşæşĺ `heapq` æĺąåİŮçŽĎä¡£çŤĺãĂĆ åĞ¡æŢř `heapq.heappush()` åŠŇ `heapq.heappop()` åĹĘåĹńåIJĺéŸ§åĹŮ `_queue` äÿŁæŔŠåĚěåŠŇåĽäéŹđ'çňňäÿĂäÿłåĚČçť'äïijŇ åźűäÿŤéŸ§åĹŮ_queueä£ÌèŕĄçňňäÿĂäÿłåĚČçť'ăæŇěæIJL'æIJĂéńŸäijŸåĚĹçžğ(1.4èŁĆåůšçżŔèőĺèőžè£ĞèŁŹ `heappop()` åĞ¡æŢřæĂżæŸŕè£ŤåŻđ"æIJĂåřŔçŽĎ"çŽĎåĚČçť'äïijŇè£ŹåřśæŸŕä£ÌèŕĄéŸ§åĹŮpopæŞ■ä¡IJè åŔęåđ'ŰïijŇçŤśäžŐpushåŠŇpopæŞ■ä¡IJæŮűéŮť'åđ'■æİĆåžęäÿžO(log N)ïijŇåĚűäÿ■NæŸŕääĘçŽĎåđ'ğåřŔïijŇåŻăæ■đ'åřśçőŮæŸŕNå¿Ĺåđ'ğçŽĎæŮűåĂŹåőČäžňè£ŘèąŇéĂ§åžęäź§

åIJĺäÿŁéİćäžčçăĄäÿ■ïijŇéŸ§åĹŮåŇĚåŘńäžĘäÿĂäÿł `(-priority, index, item)` çŽĎåĚČçżĎãĂĆ äijŸåĚĹçžğäÿžèť§æŢřçŽĎçŽőçŽĎæŸŕä¡£å¿ŮåĚČçť'ăæŇL'çĚğäijŸåĚĹçžğäžŐéńŸè£Źäÿłèů§æŹőéĂŽçŽĎæŇL'äijŸåĚĹçžğäžŐä¡ŐåĹřéńŸæŐŠåžŔçŽĎåăĘæŐŠåžŔæĄřåůğçŽÿåŔ■ãĂĆ

`index` åŔŸéĞŔçŽĎä¡IJçŤĺæŸŕä£ÌèŕĄåŘŇç■L'äijŸåĚĹçžğåĚČçť'ăçŽĎæ■čçąőæŐŠåžŔãĂĆ éĂŽè£Ğä£Ìå■ŸäÿĂäÿłäÿ■æŰ■åćđåŁăçŽĎ `index` äÿŃæăĞåŔŸéĞŔïijŇåŔŕäžěçąőä£ÌåĚČçť'ăæŇL'çĚğåőČäź èĂŇäÿŤïijŇ `index` åŔŸéĞŔäź§åIJĺçŽÿåŘŇäijŸåĚĹçžğåĚČçť'ăæŕŤè¿ČçŽĎæŰűåŹèţůåĽřéĞ■èęĄä¡IJçŤĺã

äÿžäžĘéŸŘæŸŐè£ŹäžŻïijŇåĚĹåĄĞåőŽItemåőđä¿ŃæŸŕäÿ■æŤræŇĄæŐŠåžŔçŽĎïijŽ

```
>>> a = Item('foo')
>>> b = Item('bar')
>>> a < b
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: unorderable types: Item() < Item()
>>>
```

åęĆæđIJä¡ăä¡£çŤĺåĚČçžĎ `(priority, item)` ïijŇåŔłèęĄäÿđ'äÿłåĚČçť'ăçŽĎäijŸåĚĹçžğäÿ■åŘŇåŕ ä¡ĘæŸŕåęĆæđIJäÿđ'äÿłåĚČçť'ăäijŸåĚĹçžğäÿĂæăůçŽĎèŕİïijŇéĆčäźĹæŕŤè¿ČæŞ■ä¡IJåŕśäijŽèů§äźŇåL'■äÿĂ

```
>>> a = (1, Item('foo'))
>>> b = (5, Item('bar'))
>>> a < b
True
>>> c = (1, Item('grok'))
>>> a < c
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
```

```
TypeError: unorderable types: Item() < Item()
>>>
```

éĂŽè£ĞåijŢåĚěåŔęåď'ŰçŽĎ index åŔŸéĞŔçżĎæĹŘäŸL'åĚČçżĎ (priority, index, item) ïijŇåřśèČ¡å¿Ĺåě¡çŽĎéĄ£åĚ■äŸŁéİćçŽĎéŤŹèŕŕïijŇ åŽääÿžäÿ■åŔŕèČ¡æIJL'äÿď'äÿłåĚČçť'ăæIJL'çŻÿåŘŇçŽĎ index åĂijãĂĆPythonåIJĺåĄŽåĚČçżĎæŕŤè¿ČæŮűåĂŹïijŇåęĆæđIJåL'■éİćçŽĎæŕŤè¿ČåůšçżŕåŕŕäžěçąőåőŽçżŞæđ åŘŐéİćçŽĎæŕŤè¿ČæŞ■ä¡IJåřśäÿ■äijŽåŔŚçŢ§äžĘïijŽ

```
>>> a = (1, 0, Item('foo'))
>>> b = (5, 1, Item('bar'))
>>> c = (1, 2, Item('grok'))
>>> a < b
True
>>> a < c
True
>>>
```

åęĆæđIJä¡ăæČşåIJĺåď'Žäÿłçž£çĺŃäÿ■ä¡£çŢĺåŘŇäÿĂäÿłéŸ§åĹŮïijŇéĆčäźĹä¡ăéIJĂèęĄåćđåŁăéĂĆå¡ŞçŽ åŔŕäžěæ§ěçIJŃ12.3åŕŔèŁĆçŽĎä¿Ńå■ŘæijŢçď'žæŸŕæĂŐæăůåĄŽçŽĎãĂĆ

heapq æĺąåİŮçŽĎåőŸæŰźæŰĞæąčæIJL'æŻť'èŕęçżEçŽĎä¿Ńå■ŘçĺŃåžŔäžěåŔŁåŕźäžŐåăEçŘĘèőžåŔŁ

1.6 å■ŮåĚÿäÿ■çŽĎéŤőæŸăåřĎåď'ŽäÿłåĂij

éŮőécŸ

æĂŐæăůåőđçŐřäÿĂäÿłéŤőåŕźåžŤåď'ŽäÿłåĂijçŽĎå■ŮåĚÿ(äź§åŔń multidict)ïij§

èğčåĘşæŰźæąĹ

äÿĂäÿłå■ŮåĚÿåřśæŸŕäÿĂäÿłéŤőåŕźåžŤäÿĂäÿłå■ŢåĂijçŽĎæŸăåřĎãĂĆåęĆæđIJä¡ăæČşèęĄäÿĂäÿłéŤőá ærŤåęĆåĹŮèąĺæĹŰèĂĚéŻĘåŘĹéĞŃéİćãĂĆærŤåęĆïijŇä¡ăåŔŕäžěåČŔäÿŃéİćè£ŹæăůæđĎéĂăè£ŹæăůçŽĎå

```
d = {
    'a' : [1, 2, 3],
    'b' : [4, 5]
}
e = {
    'a' : {1, 2, 3},
    'b' : {4, 5}
}
```

éĂL'æŃl'ä¡£çŢĺåĹŮèąĺè£ŸæŸŕéŻĘåŘĹåŘŰåĘşäžŐä¡ăçŽĎåőđéŹĚéIJĂæśĆãĂĆåęĆæđIJä¡ăæČşä£ÍæŇA åęĆæđIJæČşåŐżæŐL'éĞ■åď'■åĚČçť'ăåřśä¡£çŢĺéŻĘåŘĹïijĹåźűäÿŤäÿ■åĚşå£ČåĚČçť'ăçŽĎéąžåžŔéŮőécŸïijL

ä¡ăåŔŕäžěå¿ĹæŰźä¿£çŽĎä¡£çŢĺ collections æĺąåİŮäÿ■çŽĎ defaultdict æÍěæđĎéĂăè£ŹæăůçŽĎå■ŮåĚÿãĂĆ defaultdict

çŽĎäÿĂäÿłçL’źå¿ĄæŸŕåőČäijŽèĞłåŁĺåĹİåğŃåŇŰæŕŔäÿł key
åĹŽåijĂåğŃåŕźåžŤçŽĎåĂijïijŇæL’Ăäžëä¡ăåŔłéIJĂèęĄåĚşæşĺæůżåŁăåĚČçt’ăæŞ■ä¡IJäžĘãĂĆæŕŤåęĆïijŽ

```
from collections import defaultdict

d = defaultdict(list)
d['a'].append(1)
d['a'].append(2)
d['b'].append(4)

d = defaultdict(set)
d['a'].add(1)
d['a'].add(2)
d['b'].add(4)
```

éIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇ `defaultdict` äijŽèĞłåŁĺäÿžåřĘèęĄèő£éŮőçŽĎéŤő(åřśçőŮçŻőåL’■å■Ů
åęĆæđIJä¡ååźűäÿ■éIJĂèęĄè£ŹæăůçŽĎçL’źæĂğïijŇä¡ååŔŕäžěåIJĺäÿĂäÿłæŹőéĂŽçŽĎå■ŮåĚÿäÿŁä¡£çŤĺ
`setdefault()` æŰźæşŢæİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
d = {} # A regular dictionary
d.setdefault('a', []).append(1)
d.setdefault('a', []).append(2)
d.setdefault('b', []).append(4)
```

ä¡ĘæŸŕå¿Ĺåď’ŽçÍŃåžŔåŚŸèğL’å¿Ů `setdefault()` çŤĺèţůæİěæIJL’çĆźåĹńæL’■ãĂĆåŻääÿžæŕŔæňąèř
`[]`)ãĂĆ

èőÍèőž

äÿĂèĹñæİěèőšïijŇåĹŻåžžäÿĂäÿłåď’ŽåĂijæŸăářĎå■ŮåĚÿæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆä¡ĘæŸŕïijŇåęĆæđIJä
ä¡ååŔŕèČ¡äijŽåČŔäÿŃéÍcè£ŹæăůæİěåőđçŐřïijŽ

```
d = {}
for key, value in pairs:
    if key not in d:
        d[key] = []
    d[key].append(value)
```

åęĆæđIJä¡£çŤĺ `defaultdict` çŽĎèŕÍäžčçăĄåřśæŻť’åŁăçőĂæť’ĄäžĘïijŽ

```
d = defaultdict(list)
for key, value in pairs:
    d[key].append(value)
```

è£ŹäÿĂåřŔèŁĆæL’Ăèőĺèőžç ŽĎéŮőéćŸèů§æŢřæ■őåď’ĎçŘĘäÿ■çŽĎèőřå¡Ţå¡Šçśżéůőéć ŸæIJL’åď’gçŽĎ

1.7 å■ŮåĚÿæŐŠåžŔ

éŮőécŸ

ä¡ăæČşåĹŻåžžäÿĂäÿłå■ŮåĚÿïijŇåźűäÿŤåIJĺè£■äžčæĹŰåžŘåĹŮåŇŰè£Źäÿłå■ŮåĚÿçŽĎæŮűåĂŹèČ¡åd

èğčåĘşæŰźæąĹ

äÿžäžĘèČ¡æŐğåĹűäÿĂäÿłå■ŮåĚÿäÿ■åĚČçť'äçŽĎéąžåžŘïijŇä¡ăåŘŕäžěä¡£çŤĺ collections æĺąåİŮäÿ■çŽĎ OrderedDict çśžãĂĆ åIJĺè£■äžčæŞ■ä¡IJçŽĎæŮűåĂŹåőČäijŽä£İæŇĄåĚČçť'äèćńæŘŠåĚěæŮűçŽĎéąžåžŘïijŇçď'žä¿ŃåęĆäÿŃïijŽ

```
from collections import OrderedDict

d = OrderedDict()
d['foo'] = 1
d['bar'] = 2
d['spam'] = 3
d['grok'] = 4
# Outputs "foo 1", "bar 2", "spam 3", "grok 4"
for key in d:
    print(key, d[key])
```

å¡Şä¡ăæČşèęĄæđĎåžžäÿĂäÿłårĘæİěéIJĂèęĄåžŘåĹŮåŇŰæĹŰçijŰçăĄæĹŘåĚűäžŰæăijåijŘçŽĎæŸäårł OrderedDict æŸŕéİđåÿÿæIJL'çŤĺçŽĎãĂĆ ærŤåęĆïijŇä¡ăæČşçš¿çąőæŐğåĹűäžěJSONçijŰçăĄåŘŐå■Ůæő OrderedDict æİěæđĎåžžè£ŹæăůçŽĎæŢřæ■őïijŽ

```
>>> import json
>>> json.dumps(d)
'{"foo": 1, "bar": 2, "spam": 3, "grok": 4}'
>>>
```

èőĺèőž

OrderedDict åĘĚéČĺçżť'æŁď'çİĂäÿĂäÿłæăźæ■őéŤőæŘŠåĚěéąžåžŘæŐŠåžŘçŽĎåŘŇåŘŠéŞ¿èąĺãĂĆ åőČäijŽèćńæŤ¿åĹřéŞ¿èąĺçŽĎår¿éČĺãĂĆårźäžŐäÿĂäÿłåůšçżŘå■ŸåIJĺçŽĎéŤőçŽĎéĞ■åd'■ètŇåĂijäÿ■äijŽæ

éIJĂèęĄæşĺæĐŘçŽĎæŸŕïijŇäÿĂäÿł OrderedDict çŽĎåď'ğårŘæŸŕäÿĂäÿłæŹőéĂŽå■ŮåĚÿçŽĎäÿd'å æĽĂäžěåęĆædIJä¡ăèęĄædĎåžžäÿĂäÿłéIJĂèęĄåď'ğéĞŘ OrderedDict åődä¿ŃçŽĎæŢřæ■őçżŞædĎçŽĎæŮűåĂŹ(ærŤåęĆèŕżåŘŰ100,000èąŇCSVæŢřæ■őåĹräÿĂäÿł OrderedDict åĹŮèąĺäÿ■åŐż)ïijŇ éĆčäźĹä¡ăåršå¿ŮäžŤçżĘæİĆèąąäÿĂäÿŃæŸŕåŘęä¡£çŤĺ OrderedDict åÿęæİěçŽĎåě¡åď'ĎèęĄåď'ğè£ĞéćÍåď'ŰåĘĚå■ŸæűĹèĂŮçŽĎå¡šåŞ■ãĂĆ

1.8 å■ŮåĚÿçŽĎè£ŘçőŮ

éŮőécŸ

æĂŐæăůåIJĺæŢřæ■őå■ŮåĚÿäÿ■æĽ'ğèąŇäÿĂäžŻèőąçőŮæŞ■ä¡IJ(ærŤåęĆæśĆæIJĂărŘåĂijãĂĄæIJĂåď'ğ

èğčåEşæŰźæąĹ

èĂČèŹŚäÿŃéİćçŽĎèĆąçěĺåŘ■åŠŇäžůæäijæŸăářĎå■ŮåĚÿïijŽ

```
prices = {
    'ACME': 45.23,
    'AAPL': 612.78,
    'IBM': 205.55,
    'HPQ': 37.20,
    'FB': 10.75
}
```

äÿžäžĘářźå■ŮåĚÿåĂijæĽğèąŇèőąçőŮæŞ■ä¡IJïijŇéĂŽåÿÿéIJĂèęĄä¡£çŤÍ zip() åĞ¡æŢřåĚĹářĘéŤőåŠŇåĂijåŘ■è¡ňè£ĞæİěãĂĆ æŕŤåęĆïijŇäÿŃéİćæŸŕæ§ěæĽ¿æIJĂåřŔåŠŇæIJĂåd'ğèĆąçěĺ

```
min_price = min(zip(prices.values(), prices.keys()))
# min_price is (10.75, 'FB')
max_price = max(zip(prices.values(), prices.keys()))
# max_price is (612.78, 'AAPL')
```

çśżäijijçŽĎïijŇåŔŕäżěä¡£çŤÍ zip() åŠŇ sorted() åĞ¡æŢřæİěæŐŠåĹŮå■ŮåĚÿæŢřæ■őïijŽ

```
prices_sorted = sorted(zip(prices.values(), prices.keys()))
# prices_sorted is [(10.75, 'FB'), (37.2, 'HPQ'),
#                   (45.23, 'ACME'), (205.55, 'IBM'),
#                   (612.78, 'AAPL')]
```

æĽğèąŇè£ŹäžŻèőąçőŮçŽĎæŮűåĂŹïijŇéIJĂèęĄæşĺæĎŘçŽĎæŸŕ zip() åĞ¡æŢřåĹŻåżžçŽĎæŸŕäÿĂäÿłåŘłèČ¡èő£éŮőäÿĂæňąçŽĎè£■äžčåŹĺãĂĆ æŕŤåęĆïijŇäÿŃéİćçŽĎäžčçăĄåŕšäijŽäžğçŤ§éŤŹèŕŕïijŽ

```
prices_and_names = zip(prices.values(), prices.keys())
print(min(prices_and_names)) # OK
print(max(prices_and_names)) # ValueError: max() arg is an empty␣
↪sequence
```

èőĺèőž

åęĆæđIJä¡ăåIJĺäÿĂäÿłå■ŮåĚÿäÿŁæĽğèąŇæŹőéĂŽçŽĎæŢřå■ęè£ŘçőŮïijŇä¡ääijŽåŔŚçŐřåőČäžňäžĚäž

```
min(prices) # Returns 'AAPL'
max(prices) # Returns 'IBM'
```

è£ŹäÿłçżŞæđIJåźűäÿ■æŸŕä¡äæČşèęĄçŽĎïijŇåŻääÿžä¡äæČşèęĄåIJĺå■ŮåĚÿçŽĎåĂijéŻĘåŘĹäÿŁæĽğèą æĹŰèőÿä¡ääijŽářİèŕŢçİĂä¡£çŤĺå■ŮåĚÿçŽĎ values() æŰźæşŢæİèèğčåĘşè£ŹäÿłéŮőécŸïijŽ

```
min(prices.values()) # Returns 10.75
max(prices.values()) # Returns 612.78
```

äÿ■åźÿçŽĎæŸŕïijŇéĂŽåÿÿè£Źäÿłçżş æđIJåŘŇæăůäź§äÿ■æŸŕä¡ăæČşèęĄçŽĎãĂĆ
ä¡ăåŔŕèČ¡è£ŸæČşèęĄç§ěéĄŞåŕźåžŤçŽĎéŤőçŽĎä£ąæĄŕ(æŕŤåęĆéĆčçğ■èĆąçĚĺäžůæăijæŸŕæIJĂä¡ŐçŽĎïij§

ä¡ăåŔŕäżěåIJÍ `min()` åŠŇ `max()` åĞ¡æŢřäÿ■æŘŘä¿Ż `key` åĞ¡æŢřåŔĆæŢřæİěèŐůåŘŰæIJĂåřŔåĂijæĹŰæIJĂåď ğåĂijåŕźåžŤçŽĎéŤőçŽĎä£ąæĄŕãĆĆæŕŤåęĆïijŽ

```
min(prices, key=lambda k: prices[k]) # Returns 'FB'
max(prices, key=lambda k: prices[k]) # Returns 'AAPL'
```

ä¡ĘæŸŕïijŇåęĆæđIJè£ŸæČşèęĄå¿ŮåĹřæIJĂåřŔåĂijïijŇä¡ăåŔĹå¿ŮæL'ğèąŇäÿĂæňąæ§ěæL'¿æŞ■ä¡IJãĂ

```
min_value = prices[min(prices, key=lambda k: prices[k])]
```

åL'■éİćçŽĎ `zip()` åĞ¡æŢřæŰźæąĹéĂŽè£ĞåřĘå■ŮåĚÿ"åŘ■è¡ň"äÿž(åĂijïijŇéŤő)åĚČçżĎåžŔåĹŮæİěè
å¡ŞærŤè¿Čäÿď äÿłåĚČçżĎçŽĎæŮűåĂŹïijŇåĂijäijŽåĚĹè£ŻèąŇærŤè¿ČïijŇçĎűåŘŐæL'■æŸŕéŤőãĂĆ
è£ŹæăůçŽĎèŕİä¡ăåŕśèČ¡éĂŽè£ĞäÿĂæİąçőĂå■ŢçŽĎèŕ■åŘěåŕśèČ¡å¿Ĺè¡żæİ¿çŽĎåőđçŐřåIJĺå■ŮåĚÿäÿŁçŽĎ

éIJĂèęĄæşĺæĎŔçŽĎæŸŕåIJĺèőąçőŮæŞ■ä¡IJäÿ■ä¡£çŤĺåĹřäžĘ(åĂijïijŇéŤő)åŕźãĂĆå¡Şåď Žäÿłåőđä¡ŞæŇ
ærŤåęĆïijŇåIJĺæL'ğèąŇ `min()` åŠŇ `max()` æŞ■ä¡IJçŽĎæŮűåĂŹïijŇåęĆæđIJæĄřåůğæIJĂåřŔæĹŰæIJĂåď

```
>>> prices = { 'AAA' : 45.23, 'ZZZ': 45.23 }
>>> min(zip(prices.values(), prices.keys()))
(45.23, 'AAA')
>>> max(zip(prices.values(), prices.keys()))
(45.23, 'ZZZ')
>>>
```

1.9 æ§ěæL'¿äÿď å■ŮåĚÿçŽĎçŻÿåŘŇçĆź

éŮőécŸ

æĂŐæăůåIJĺäÿď äÿłå■ŮåĚÿäÿ■årżårżæL'¿çŻÿåŘŇçĆź(ærŤåęĆçŻÿåŘŇçŽĎéŤőãĂĄçŻÿåŘŇçŽĎåĂijç■

èğčåĘşæŰźæąĹ

èĂČèŹŚäÿŃéİćäÿď äÿłå■ŮåĚÿïijŽ

```
a = {
    'x' : 1,
    'y' : 2,
    'z' : 3
}

b = {
    'w' : 10,
    'x' : 11,
    'y' : 2
}
```

äÿžäžĘåŕżæL’¿äÿd’äÿłå■ŮåĚÿçŽĎçŻÿåŘŇçĆźïijŇåŔŕäżěçőĂå■ŢçŽĎåIJÍäÿd’å■ŮåĚÿçŽĎ
keys() æĹŰèĂĚ items() æŰźæşŢè£ŤåŻđçżŞæđIJäÿŁæL’ğèąŇéŻĘåŘĹæŞ■ä¡IJãĂĆæŕŤåęĆïijŽ

```
# Find keys in common
a.keys() & b.keys() # { 'x', 'y' }
# Find keys in a that are not in b
a.keys() - b.keys() # { 'z' }
# Find (key,value) pairs in common
a.items() & b.items() # { ('y', 2) }
```

è£ŹäžŻæŞ■ä¡IJäź§åŔŕäżěçŤĺäžŐä£őæŤźæĹŰèĂĚè£Ğæżd’å■ŮåĚÿåĚČçt’ããĂĆ
æŕŤåęĆïijŇåĄĞåęĆä¡ăæČşäżěçŐřæIJL’å■ŮåĚÿæđĎéĂääÿĂäÿłæŐŠéŹd’åĞääÿłæŇĞåőŽéŤőçŽĎæŰřå■ŮåĚ
äÿŃéİćåĹl’çŤĺå■ŮåĚÿæŐĺåŕijæİěåőđçŐřè£ŹæăůçŽĎéIJĂæśĆïijŽ

```
# Make a new dictionary with certain keys removed
c = {key:a[key] for key in a.keys() - {'z', 'w'}}
# c is {'x': 1, 'y': 2}
```

èőĺèőž

äÿĂäÿłå■ŮåĚÿåŕśæŸŕäÿĂäÿłéŤőéŻĘåŘĹäÿŐåĂijéŻĘåŘĹçŽĎæŸăåřĎåĚşçşżãĂĆ
å■ŮåĚÿçŽĎ keys() æŰźæşŢè£ŤåŻđäÿĂäÿłåśŢçŐřéŤőéŻĘåŘĹçŽĎéŤőèĞĘåŻ¿åŕźèśąãĂĆ
éŤőèĞĘåŻ¿çŽĎäÿĂäÿłå¿ĹåřŚèćńäžĘèğčçŽĎçL’źæĂğåŕśæŸŕåőČäžňäź§æŤŕæŇĄéŻĘåŘĹæŞ■ä¡IJïijŇæŕŤåęĆ
æL’Ăäżěïij Ňåę Ćæđ IJä¡ăæČşåŕźéŻĘåŘĹçŽĎéŤőæL’ğèąŇäÿĂäžŻæŹőéĂŽçŽĎéŻĘåŘĹæŞ■ä¡IJïijŇåŔŕäżěçŻt’

å■ŮåĚÿçŽĎ items() æŰźæşŢè£ŤåŻđäÿĂäÿłåŇĚåŘń(éŤőïijŇåĂij)åŕźçŽĎåĚČçt’äèğĘåŻ¿åŕźèśąãĂĆ
è£Źäÿłåŕźèśąå ŘŇæăůäź§æŤŕæŇĄéŻĘåŘĹæŞ■ä¡IJïijŇåźűäÿŤåŔŕäżěèćńçŤĺæİěæ§ěæL’¿äÿd’äÿłå■ŮåĚÿæIJL’å

år¡çőąå■ŮåĚÿçŽĎ values() æŰźæşŢäź§æŸŕçśżäijijïijŇä¡ĘæŸŕåőČåźűäÿ■æŤŕæŇĄè£ŹéĞŇäżŃçż■ç
æ§Řçğ■çÍŇåžęäÿŁæŸŕåŻääÿžåĂijèğĘåŻ¿äÿ■èČ¡ä£İèŕĄæL’ĂæIJL’çŽĎåĂijäžŠäÿ■çŽÿåŘŇïijŇè£ŹæăůäijŽår
äÿ■è£ĞïijŇåęĆæđIJä¡ăçąňèęĄåIJĺåĂijäÿŁéİćæL’ğèąŇè£ŹäžŻéŻĘåŘĹæŞ■ä¡IJçŽĎèŕÍïijŇä¡ääŔŕäżěåĚĹåřĘåĂ

1.10 åĹåéŹd’åžŔåĹŮçŽÿåŘŇåĚČçt’ääźűä£ÌæŇĄéąžåžŔ

éŮőécŸ

æĂŐæăůåIJÍäÿĂäÿłåžŔåĹŮäÿŁéİćä£ÌæŇĄåĚČçt’ăéąžåžŔçŽĎåŘŇæŮűæűĹéŹd’éĞ■åd’■çŽĎåĂijïij§

èğčåĘşæŰźæąĹ

åęĆæđIJåžŔåĹŮäÿŁçŽĎåĂijéČ¡æŸŕ hashable çśżåđŃïijŇéĆčäźĹåŔŕäżěå¿ĹçőĂå■ŢçŽĎåĹl’çŤĺéŽĘå

```
def dedupe(items):
    seen = set()
    for item in items:
        if item not in seen:
```

```
            yield item
        seen.add(item)
```

äÿŃéÌćæŸŕä¡£çŤÍäÿŁè£råĞ¡æŢřçŽĎä¿Ńå■ŘïijŽ

```
>>> a = [1, 5, 2, 1, 9, 1, 5, 10]
>>> list(dedupe(a))
[1, 5, 2, 9, 10]
>>>
```

è£ŹäÿłæŰźæşŢäžĚäžĚåIJÍåžŔåĹŮäÿ■åĚČçť'ääÿž hashable çŽĎæŮűåĂŹæL'■çőąçŤÍãĂĆ åęĆæđIJä¡äæČşæűĹéŹď'åĚČçť'ääÿ■åŔŕåŞĹåÿŃ(æŕŤåęĆ dict çśžåđŃ)çŽĎåžŔåĹŮäÿ■éĞ■åď'■åĚČçť'ăçŽĎèŕÏïijŇä¡äéIJĂèęĄåřĘäÿŁè£řäžčçăĄçÍ■å¿őæŤźåŔŸäÿĂäÿŃïijŇä

```
def dedupe(items, key=None):
    seen = set()
    for item in items:
        val = item if key is None else key(item)
        if val not in seen:
            yield item
            seen.add(val)
```

è£ŹéĞŃçŽĎkeyåŔĆæŢřæŇĞåőŽäžĘäÿĂäÿłåĞ¡æŢřïijŇårĘåžŔåĹŮåĚČçť'äè¡ňæ■ćæĹŘ hashable çśžåđŃãĂĆäÿŃéÌćæŸŕåőČçŽĎçŤÍæşŢçď'žä¿ŃïijŽ

```
>>> a = [ {'x':1, 'y':2}, {'x':1, 'y':3}, {'x':1, 'y':2}, {'x':2, 'y
↪':4}]
>>> list(dedupe(a, key=lambda d: (d['x'],d['y'])))
[{'x': 1, 'y': 2}, {'x': 1, 'y': 3}, {'x': 2, 'y': 4}]
>>> list(dedupe(a, key=lambda d: d['x']))
[{'x': 1, 'y': 2}, {'x': 2, 'y': 4}]
>>>
```

åęĆæđIJä¡äæČşå§žäžŐå■Ţäÿłå■Ůæőţã ĂĄåśđæĂğæĹŰèĂĚæ§ŘäÿłæŻť'åď'ğçŽĎæŢřæ■őçżŞæđĎæİěæű

èőÍèőž

åęĆæđIJä¡ääžĚäžĚåřśæŸŕæČşæűĹéŹď'éĞ■åď'■åĚČçť'äïijŇéĂŽåÿÿåŔŕäžěçőĂå■ŢçŽĎæđĎéĂääÿĂäÿłé

```
>>> a
[1, 5, 2, 1, 9, 1, 5, 10]
>>> set(a)
{1, 2, 10, 5, 9}
>>>
```

çĎűèĂŇïijŇè£Źçğ■æŰźæşŢäÿ■èČ¡çżť'æŁď'åĚČçť'ăçŽĎéąžåžŔïijŇçŤ§æĹŘçŽĎçżŞæđIJäÿ■çŽĎåĚČçť'

åIJÍæIJňèŁĆäÿ■æĹŚäžňä¡£çŤÍäžĘçŤ§æĹŘåŹÍåĞ¡æŢřèől'æĹŚäžňçŽĎåĞ¡æŢřæŻť'åŁäéĂŽçŤÍïijŇäÿ■äžł
æŕŤåęĆïijŇåęĆæđIJåęĆæđIJä¡äæČşèŕžåŔŰäÿĂäÿłæŰĞäžűïijŇæűĹéŹď'éĞ■åď'■èąŇïijŇä¡ääŔŕäžěå¿ĹåőźæŸ

```
with open(somefile,'r') as f:
for line in dedupe(f):
    ...
```

äÿŁè£řkeyåĞ¡æŢřåŔĆæŢřæĺąäż£äžĘ sorted() , min() åŠŇ max() ç■L'åĘĕç¡őåĞ¡æŢřçŽĎçŻÿäijijåŁ§èČ¡ãĂĆ åŔŕäżěåŔĆèĂČ1.8åŠŇ1.13åřŔèŁĆäžĘèğčæŻť'åď'ŽãĂĆ

1.11 åŚ¡åŘ■åĹĞçL'Ğ

éŮőécŸ

ä¡ăçŽĎçÍŃåžŔåůšçżŔåĞžçŐřäÿĂåď'ğåäĘåůšæŮăæşŢçŻť'èğĘçŽĎçąñçijŰçăĄåĹĞçL'ĞäÿŃæăĞïijŇçĎű

èğčåĘşæŰźæąĹ

åĄĞåőŽä¡ăæIJL'äÿĂæőţäżčçăĄèęĄäżŐäÿĂäÿłèőřå¡Ţå■Ůçñęäÿšäÿ■åĞääÿłåŽžåőŽä¡■ç¡őæŔŘåŔŰåĞžç

```
###### 01234567890123456789012345678901234567890123456789012345678901234567890
↪'
record = '.....................100 .......513.25 ..........'
cost = int(record[20:23]) * float(record[31:37])
```

äÿŐåĚűéĆčæăůåĘŹïijÑäÿžäżĂäźĹäÿ■æČşè£ŹæăůåŚ¡åŘ■åĹĞçL'ĞåŚćïijŽ

```
SHARES = slice(20, 23)
PRICE = slice(31, 37)
cost = int(record[SHARES]) * float(record[PRICE])
```

çñňäžŇçğ■çL'ĹæIJñäÿ■ïijŇä¡ăéĄ£åĚ■äžĘåď'ğéĞŔæŮăæşŢçŘĘèğčçŽĎçąñçijŰçăĄäÿŃæăĞïijŇä¡£å¿Ůă

èőĺèőž

äÿĂèĹñæİěèőšïijŇäžčçăĄäÿ■åęĆæđIJåĞžçŐřåď'ğéĞŔçŽĎçąñçijŰçăĄäÿŃæăĞåĂijäijŽä¡£å¿ŮåŔŕèŕžæ
æŕŤåęĆïijŇåęĆæđIJä¡ăåŽđè£ĞæİěçIJŃçIJŃäÿĂåźť'åL'■ä¡ăåĘŹçŽĎäžčçăĄïijŇä¡ăäijŽæŚÿçİĂèĎŚècŃæČşéĆ
è£ŹéĞŇçŽĎèğčåĘşæŰźæąĹæŸŕäÿĂäÿłå¿ĹçőĂå■ŢçŽĎæŰźæşŢèőľä¡ăæŽť'åŁăæÿĚæŹŕçŽĎèąĺè¿¿äžčçăĄåĹ

åĘĚç¡őçŽĎ slice() åĞ¡æŢřåĹŽåžžäžĘäÿĂäÿłåĹĞçL'Ğåŕźèśąïij ŇåŔŕäžěèćńçŤĺåIJĺäžża¡ŢåĹĞçL'ĞåĚ

```
>>> items = [0, 1, 2, 3, 4, 5, 6]
>>> a = slice(2, 4)
>>> items[2:4]
[2, 3]
>>> items[a]
[2, 3]
>>> items[a] = [10,11]
>>> items
[0, 1, 10, 11, 4, 5, 6]
```

```
>>> del items[a]
>>> items
[0, 1, 4, 5, 6]
```

åęĆædIJä¡ăæIJL'äÿĂäÿłåĹĞçL'ĞåŕźèśąaïijŇä¡ăåŔŕäżěåĹĘåĹńèřČçŤĺåőČçŽĎ `a.start`, `a.stop`, `a.step` åśđæĂğæİěèŐůåŔŰæŻt'åd'ŽçŽĎä£ąæĄŕãĂĆæŕŤåęĆïijŽ

```
>>> a = slice(5, 50, 2)
>>> a.start
5
>>> a.stop
50
>>> a.step
2
>>>
```

åŔęåd'ŰïijŇä¡ăè£ŸèČ¡éĂŽè£ĞèřČçŤĺåĹĞçL'ĞçŽĎ `indices(size)` æŰźæşŢåřĘåőČæŸăåřĎåĹřäÿĂäÿłçąőåőŽåd'ğåřŔçŽĎåžŔåĹŮäÿŁïijŇ è£ŹäÿłæŰźæşŢè£ŤåŻđäÿĂäÿłäÿL'åĚČçżĎ `(start, stop, step)` ïijŇæL'ĂæIJL'åĂijéČ¡äijŽèćńåŘĹéĂĆçŽĎçijl'åřŔäżěæżąèűşè¿źçŢŇéŹŘåĹűïijŇ äżŐèĂŇä¡£çŤĺçŽĎæŮűåĂŹéĄ£åĚ■åĞžçŐř `IndexError` åijĆåÿÿãĂĆæŕŤåęĆïijŽ

```
>>> s = 'HelloWorld'
>>> a.indices(len(s))
(5, 10, 2)
>>> for i in range(*a.indices(len(s))):
... print(s[i])
...
W
r
d
>>>
```

1.12 åžŔåĹŮäÿ■åĞžçŐřæňąæŢřæIJĂåd'ŽçŽĎåĚČçt'ă

éŮőécŸ

æĂŐæăůæL'¿åĞžäÿĂäÿłåžŔåĹŮäÿ■åĞžçŐřæňąæŢřæIJĂåd'ŽçŽĎåĚČçt'ăåŚćïij§

èğčåĘşæŰźæąĹ

`collections.Counter` çśžåřśæŸŕäÿŞéŮĺäÿžè£ŹçśżéŮőécŸèĂŇèő¿èőąçŽĎïijŇ åőČçŤŽèĞşæIJL'äÿĂäÿłæIJL'çŤĺçŽĎ `most_common()` æŰźæşŢçŻt'æŐěçżŹäžĘä¡äç■ŤæąĹãĂĆ

äÿžäžĘæijŤçd'žïijŇåĚĹåĄĞèő¿ä¡ăæIJL'äÿĂäÿłå■Ţèŕ■åĹŮèąĺåźűäÿŤæČşæL'¿åĞžåŞłäÿłå■Ţèŕ■åĞžçŐřé

```
words = [
    'look', 'into', 'my', 'eyes', 'look', 'into', 'my', 'eyes',
    'the', 'eyes', 'the', 'eyes', 'the', 'eyes', 'not', 'around',
↪'the',
    'eyes', "don't", 'look', 'around', 'the', 'eyes', 'look', 'into
↪',
    'my', 'eyes', "you're", 'under'
]
from collections import Counter
word_counts = Counter(words)
# åĞžçŐřécŚçŐĞæIJĂéńŸçŽĎ3äÿłå■Ţèŕ■
top_three = word_counts.most_common(3)
print(top_three)
# Outputs [('eyes', 8), ('the', 5), ('look', 4)]
```

èőÍèőž

ä¡IJäÿžè¿ŞåĚěïijŇ Counter åŕźèśąåŔŕäžěæŐěåŔŮäżżæĎŔçŽĎçŤśåŔŕåŞĹåÿŇ(hashable)åĚČçť'äæđ
åIJĺåžŢåśĆåőđçŐřäÿŁïijŇäÿĂäÿł Counter åŕźèśąåřśæŸŕäÿĂäÿłå■ŮåĚÿïijŇåřĘåĚČçť'äæŸäåřĎåĹřåőČåĞžç

```
>>> word_counts['not']
1
>>> word_counts['eyes']
8
>>>
```

åęĆæđIJä¡äæČşæL'ŃåŁĺåćđåŁäèőąæŢřïijŇåŔŕäžěçőĂå■ŢçŽĎçŤĺåŁäæşŢïijŽ

```
>>> morewords = ['why','are','you','not','looking','in','my','eyes']
>>> for word in morewords:
... word_counts[word] += 1
...
>>> word_counts['eyes']
9
>>>
```

æĹŰèĂĚä¡ääŔŕäžěä¡£çŤĺ update() æŰźæşŢïijŽ

```
>>> word_counts.update(morewords)
>>>
```

Counter åőđä¿ŃäÿĂäÿłéšIJäÿžäžžç§ěçŽĎçL'źæĂğæŸŕåőČäżňåŔŕäžěå¿ĹåőźæŸŞçŽĎèů§æŢřå■ęè£Řç

```
>>> a = Counter(words)
>>> b = Counter(morewords)
>>> a
Counter({'eyes': 8, 'the': 5, 'look': 4, 'into': 3, 'my': 3, 'around
↪': 2,
"you're": 1, "don't": 1, 'under': 1, 'not': 1})
```

```
>>> b
Counter({'eyes': 1, 'looking': 1, 'are': 1, 'in': 1, 'not': 1, 'you
 ↪': 1,
'my': 1, 'why': 1})
>>> # Combine counts
>>> c = a + b
>>> c
Counter({'eyes': 9, 'the': 5, 'look': 4, 'my': 4, 'into': 3, 'not':␣
 ↪2,
'around': 2, "you're": 1, "don't": 1, 'in': 1, 'why': 1,
'looking': 1, 'are': 1, 'under': 1, 'you': 1})
>>> # Subtract counts
>>> d = a - b
>>> d
Counter({'eyes': 7, 'the': 5, 'look': 4, 'into': 3, 'my': 2, 'around
 ↪': 2,
"you're": 1, "don't": 1, 'under': 1})
>>>
```

æŕńæŮăçŰŚéŮőïijŇ `Counter` åŕźèśąåIJĺåĞääźŐæL’ĂæIJL’éIJÄèęĄåĹűèąĺæĹŰèĂĚèőąæŢřæŢřæ■őçŽÌ
åIJĺèğčåĘşè£ŹçśžéŮőéćŸçŽĎæŮűåĂŹä¡ääžŤèŕřäijŸåĚĹéĂL’æŃl’åőČïijŇèĂŇäÿ■æŸŕæL’ŃåŁĺçŽĎåĹl’çŤĺå

1.13 éĂŽè£Ğæ§ŘäÿłåĕşéŤőå■ŮæŐŠåžŔäÿĂäÿłå■ŮåĔÿåĹŮèąĺ

éŮőécŸ

ä¡ăæIJL’äÿĂäÿłå■ŮåĔÿåĹŮèąĺïijŇä¡ăæČşæäźæ■őæ§Řäÿłæĺ̈Ůæ§ŘåĞääÿłå■ŮåĔÿå■ŮæőţæİěæŐŠåžŔè£

èğčåĘşæŰźæąĹ

éĂŽè£Ğä¡£çŤÍ `operator` æĺąåİŮçŽĎ `itemgetter`
åĞ¡æŢřïijŇåŔŕäžěéİđåÿÿåőźæŸŞçŽĎæŐŠåžŔè£Źæăůç ŽĎæŢřæ■őçżŞæđĎãĂĆ
åĄĞèő¿ä¡ääžŐæŢřæ■őåžŞäÿ■æčĂçť ćåĞžæİěç¡ŞçńŹäijŽåŚŸä£qæĄŕåĹŮèąĺïijŇåźűäÿŤäžěäÿŃåĹŮçŽĎæŢřa

```
rows = [
    {'fname': 'Brian', 'lname': 'Jones', 'uid': 1003},
    {'fname': 'David', 'lname': 'Beazley', 'uid': 1002},
    {'fname': 'John', 'lname': 'Cleese', 'uid': 1001},
    {'fname': 'Big', 'lname': 'Jones', 'uid': 1004}
]
```

æăźæ■őäżżæĎŔçŽĎå■ŮåĔÿå■ŮæőţæİěæŐŠåžŔè¿ŞåĚěçżŞæđIJèąŇæŸŕå¿Ĺåőźæ ŸŞåőđçŐrçŽĎïijŇäžč

```
from operator import itemgetter
rows_by_fname = sorted(rows, key=itemgetter('fname'))
rows_by_uid = sorted(rows, key=itemgetter('uid'))
print(rows_by_fname)
print(rows_by_uid)
```

äžčçăĄçŽĎè¿ŞåĞžåęĆäÿŃïijŽ

```
[{'fname': 'Big', 'uid': 1004, 'lname': 'Jones'},
{'fname': 'Brian', 'uid': 1003, 'lname': 'Jones'},
{'fname': 'David', 'uid': 1002, 'lname': 'Beazley'},
{'fname': 'John', 'uid': 1001, 'lname': 'Cleese'}]
[{'fname': 'John', 'uid': 1001, 'lname': 'Cleese'},
{'fname': 'David', 'uid': 1002, 'lname': 'Beazley'},
{'fname': 'Brian', 'uid': 1003, 'lname': 'Jones'},
{'fname': 'Big', 'uid': 1004, 'lname': 'Jones'}]
```

itemgetter() åĞ¡æŢřäź§æŤŕæŇĄåď'ŽäÿłkeysïijŇæŕŤåęĆäÿŃéÍćçŽĎäžčçăĄ

```
rows_by_lfname = sorted(rows, key=itemgetter('lname','fname'))
print(rows_by_lfname)
```

äijŽäžğçŤ§åęĆäÿŃçŽĎè¿ŞåĞžïijŽ

```
[{'fname': 'David', 'uid': 1002, 'lname': 'Beazley'},
{'fname': 'John', 'uid': 1001, 'lname': 'Cleese'},
{'fname': 'Big', 'uid': 1004, 'lname': 'Jones'},
{'fname': 'Brian', 'uid': 1003, 'lname': 'Jones'}]
```

èőÍèőž

åIJÍäÿŁéÍćä¿Ńå■Řäÿ■ïijŇ rows èćńäijăéĂŠçżŹæÓěåŔŮäÿĂäÿłåĚşéŤőå■ŮåŔĆæŢřçŽĎ sorted() åĘĚç¡őåĞ¡æŢřãĂĆ è£ŹäÿłåŔĆæŢřæŸŕ callable çśžåđŃïijŇåźűäÿŤäžŐ rows äÿ■æÓěåŔŮäÿĂäÿłå■ŢäÿĂåĚČçť'äïijŇçĎűåŘŐè£ŤåŻđèćńçŤÍæÍěæŐŠåžŔçŽĎåĂijãĂĆ itemgetter() åĞ¡æŢřářśæŸŕèť§èť'čåĹŻåžžè£Źäÿł callable åŕźèśąçŽĎãĂĆ

operator.itemgetter() åĞ¡æŢřæIJL'äÿĂäÿłèćń rows äÿ■çŽĎèőřå¡ŢçŤÍæÍěæ§ěæL'¿åĂijçŽĎçť'ćåijŢåŔĆæŢřãĂĆåŔŕäžěæŸŕäÿĂäÿłå■ŮåĚÿéŤőåŘ■çğřïijŇäÿĂäÿłæŢť'å¡ćåĂijæĹŰèĂĚäžža¡ŢèČ¡åď§äijååĚěäÿĂäÿłåŕźèśąçŽĎ __getitem__() æŰźæşŢçŽĎåĂijãĂĆ åęĆæđIJä¡ăäijååĚěåď'Žäÿłçť'ćåijŢåŔĆæŢřçżŹ itemgetter() ïijŇåőČçŤ§æĹŘçŽĎ callable åŕźèśąäijŽè£ŤåŻđäÿĂäÿłåŇĚåŘńæL'ĂæIJL'åĚČçť'ääĂijçŽĎåĚČçżĎïijŇ åźűäÿŤ sorted() åĞ¡æŢřäijŽæăźæ■őè£ŹäÿłåĚČçżĎäÿ■åĚČçť'ăéąžåžŘåŐżæŐŠåžŘãĂĆ ä¡Ęä¡ăæČşèęĄåŘŇæŮűåIJĺåĞăäÿłå■Ůæőţäÿ ŁéÍćè£ŽèąŇæŐŠåžŔ(æŕŤåęĆéĂŽè£ĞåğŞåŠŇåŘ■æÍěæŐŠåžŘïij

itemgetter() æIJL'æŮűåĂŹäź§åŘŕäžěçŤÍ lambda èąÍè¿¿åijŔäžčæŽ£ïijŇæŕŤåęĆïijŽ

```
rows_by_fname = sorted(rows, key=lambda r: r['fname'])
rows_by_lfname = sorted(rows, key=lambda r: (r['lname'],r['fname']))
```

è£Źçğ■æŰźæąĹäź§äÿ■éŤŹãĂĆä¡ĘæŸŕïijŇä¡£çŤÍ itemgetter() æŰźåijŔäijŽè£ŘèąŇçŽĎçÍ■å¿őå£ńçĆźãĂĆåŽăæ■ď'ïijŇåęĆæđIJä¡ăåŕźæĂğèČ¡èęĄæśĆæŕŤè¿ČéŃŸçŽĎèŕÍåřš itemgetter() æŰźåijŔãĂĆ

æIJÄåŘŐïijŇäÿ■èęĄå£ŸäžĘè£ŹèŁĆäÿ■åśŢçd'žçŽĎæŁĂæIJŕäź§åŘŇæăůéĂĆçŤĺäžŐ min() åŠŇ max() ç■L'åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> min(rows, key=itemgetter('uid'))
{'fname': 'John', 'lname': 'Cleese', 'uid': 1001}
>>> max(rows, key=itemgetter('uid'))
{'fname': 'Big', 'lname': 'Jones', 'uid': 1004}
>>>
```

1.14 æŐŠåžŔäÿ■æŤŕæŇĄåŐ§çŢ§æŕŤè¿ČçŽĎåŕźèśą

éŮőécŸ

ä¡ăæČşæŐŠåžŔçśżåđŇçŻÿåŘŇçŽĎåŕźèśąïijŇä¡ĘæŸŕäżŰäžňäÿ■æŤŕæŇĄåŐ§çŢ§çŽĎæŕŤè¿ČæŞ■ä¡IJã

èğčåĘşæŰźæąĹ

åĘĚç¡őçŽĎ sorted() åĞ¡æŢřæIJL'äÿĂäÿłåĚşéŤőå■ŮåŔĆæŢř key ïijŇåŔŕäżěäijăåĚěäÿĂäÿł callable åŕźèśąçżŹåőČïijŇ è£Źäÿłcallable åŕźèśąåŕźæŕŔäÿłäijăåĚěçŽĎåŕźèśąè£ŤåŻđäÿĂäÿłåĂijïijŇè£ŹäÿłåĂijäijŽèćń sorted çŤĺæİěæŐŠåžŔè£ŹäžŻåŕźèśąãĂĆ æŕŤåęĆïijŇåęĆæđIJä¡ăåIJĺåžŢçŤĺçĺŇåžŔéĞŇéİćæIJL'äÿĂäÿł User åőđä¿ŃåžŔåĹŮïijŇåźűäÿŤä¡ăåÿŇæIJŻéĂŽè£ĞäżŰäžňçŽĎ user_id åśđæĂğè£ŻèąŇæŐŠåžŔïijŇ ä¡ăåŔŕäżěæŔŘä¿ŻäÿĂäÿłäžě User åőđä¿Ńä¡IJäÿžè¿ŞåĚěåźűè¿ŞåĞžåŕźåžŤ user_id åĂijçŽĎ callable åŕźèśąãĂĆæŕŤåęĆïijŽ

```
class User:
    def __init__(self, user_id):
        self.user_id = user_id

    def __repr__(self):
        return 'User({})'.format(self.user_id)


def sort_notcompare():
    users = [User(23), User(3), User(99)]
    print(users)
    print(sorted(users, key=lambda u: u.user_id))
```

åŔęåd'ŰäÿĂçğ■æŰźåijŔæŸŕä¡£çŤĺ operator.attrgetter() æİěäžčæŻ£lambdaåĞ¡æŢřïijŽ

```
>>> from operator import attrgetter
>>> sorted(users, key=attrgetter('user_id'))
[User(3), User(23), User(99)]
>>>
```

èőÍèőž

éĂL'æŃl'ä¡£çŤÍlambdaåĞ¡æŢřæĹŰèĂĚæŸŕ attrgetter() åŔŕèČ¡åŔŰåĘşäžŐäÿłäžžåŰIJåě¡ãĂĆ ä¡ĘæŸŕïijŇ attrgetter() åĞ¡æŢřéĂŽåÿÿäijŽè£ŘèąŇçŽĎå£ńçĆźïijŇåźűäÿŤè£ŸèČ¡åŘŇæŮűåĚĄèőÿåd'Žäÿłå■Ůæőţè£ŻèąŇærŤè¿ČãĂ è£Źäÿłèůş operator.itemgetter() åĞ¡æŢřä¡IJçŤÍäžŐå■ŮåĚÿçśžåđŃå¿Ĺçśżäijij(åŔĆèĂČ1.13åřŔèŁĆ ä¿ŃåęĆïijŇåęĆæđIJ User åőđä¿Ńè£ŸæIJL'äÿĂäÿł first_name åŠŇ last_name åśđæĂğïijŇéĆčäźĹåŔŕäžěåŘŚäÿŃéÍćè£ŹæăůæŐŠåžŔïijŽ

```
by_name = sorted(users, key=attrgetter('last_name', 'first_name'))
```

åŘŇæăůéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijŇè£ŹäÿĂåřŔèŁĆçŤÍåĹřçŽĎæŁĂæIJŕåŘŇæăůéĂĆçŤÍäžŐåČŔ min() åŠŇ max() äźŃçśżçŽĎåĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> min(users, key=attrgetter('user_id'))
User(3)
>>> max(users, key=attrgetter('user_id'))
User(99)
>>>
```

1.15 éĂŽè£Ğæ§Řäÿłå■Ůæőţå řĘèőřå¡ŢåĹĘçżĎ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłå■ŮåĚÿæĹŰèĂĚåőđä¿ŃçŽĎåžŔåĹŮïijŇçĎűåŘŐä¡ăæČşæăźæ■őæ§Řäÿłçl'źåőŽçŽĎå■Ů date æÍěåĹĘçżĎè£■äžčèő£éŮőãĂĆ

èğčåĘşæŰźæąĹ

itertools.groupby() åĞ¡æŢřåŕźäžŐè£ŹæăůçŽĎæŢřæ■őåĹĘçżĎæŞ■ä¡IJéÍđåÿÿåőđçŤÍãĂĆ äÿžäžĘæijŤçd'žïijŇåĄĞèő¿ä¡ăåůšçżŔæIJL'äžĘäÿŃåĹŮçŽĎå■ŮåĚÿåĹŮèąÍïijŽ

```
rows = [
    {'address': '5412 N CLARK', 'date': '07/01/2012'},
    {'address': '5148 N CLARK', 'date': '07/04/2012'},
    {'address': '5800 E 58TH', 'date': '07/02/2012'},
    {'address': '2122 N CLARK', 'date': '07/03/2012'},
    {'address': '5645 N RAVENSWOOD', 'date': '07/02/2012'},
    {'address': '1060 W ADDISON', 'date': '07/02/2012'},
    {'address': '4801 N BROADWAY', 'date': '07/01/2012'},
    {'address': '1039 W GRANVILLE', 'date': '07/04/2012'},
]
```

çŐřåIJÍåĄĞèő¿ä¡ăæČşåIJÍæŃL'dateåĹĘçżĎåŘŐçŽĎæŢřæ■őåÍŮäÿŁè£ŻèąŇè£■äžčãĂĆäÿžäžĘè£ŹæăůåÍ date)æŐŠåžŔïijŃ çĎűåŘŐěřČçŤÍ itertools.groupby() åĞ¡æŢřïijŽ

```
from operator import itemgetter
from itertools import groupby

# Sort by the desired field first
rows.sort(key=itemgetter('date'))
# Iterate in groups
for date, items in groupby(rows, key=itemgetter('date')):
    print(date)
    for i in items:
        print('  ', i)
```

è£ŘèąŇçżŞæđIJïijŽ

```
07/01/2012
  {'date': '07/01/2012', 'address': '5412 N CLARK'}
  {'date': '07/01/2012', 'address': '4801 N BROADWAY'}
07/02/2012
  {'date': '07/02/2012', 'address': '5800 E 58TH'}
  {'date': '07/02/2012', 'address': '5645 N RAVENSWOOD'}
  {'date': '07/02/2012', 'address': '1060 W ADDISON'}
07/03/2012
  {'date': '07/03/2012', 'address': '2122 N CLARK'}
07/04/2012
  {'date': '07/04/2012', 'address': '5148 N CLARK'}
  {'date': '07/04/2012', 'address': '1039 W GRANVILLE'}
```

èőÍèőž

`groupby()` åĞ¡æŢřæL'ńæŔŘæŢt'äÿłåžŘåĹŮåźűäÿŤæ§ěæL'¿è£đçż■çŻÿåŘŇåĂij(æĹŰèĂĚæăźæ■őæŇ
åIJÍæŕŘæňąè£■äžčçŽĎæŮűåĂŹïijŇåőČäijŽè£ŤåŽđäÿĂäÿłåĂijåŠŇäÿĂäÿłè£■äžčåŹÍåŕźèśąïijŇ
è£ŹäÿłèŁ■äžčåŹÍåŕźèśąåŔŕäžěçŤ§æĹŘåĚČçt'ääĂijåĚÍéČÍç■L'äžŐäÿŁéÍćéĆčäÿłåĂijçŽĎçżĎäÿ■æL'ĂæIJL'åŕ

äÿĂäÿłéİđåÿÿéĞ■èęĄçŽĎåĞĘåđ'Ğæ■ěéłđ'æŸŕèęĄæăźæ■őæŇĞåőŽçŽĎå■ŮæőţåřĕæŢřæ■őæŐŠåžŔãĂ
åŽääÿž `groupby()` äžĚäżĚæčĂæ§ěè£đçż■çŽĎåĚČçt'ăïijŇåęĆæđIJäžŃåĚĹåźűæšąæIJL'æŐŠåžŔåőŇæĹŘç

åęĆæđIJä¡ăäżĚäżĚåŔłæŸŕæČşæăźæ■ő `date` å■ŮæőţåřĕæŢřæ■őåĹĘçżĎåĹřäÿĂäÿłåđ'ğçŽĎæŢřæ■őçżŞ
éĆčäźĹä¡ăæIJĂåě¡ä¡£çŤÍ `defaultdict()` æİěæđĎåżžäÿĂäÿłåđ'ŽåĂijå■ŮåĚÿïijŇåĚşäžŐåđ'ŽåĂijå■ŮåĚÿ

```
from collections import defaultdict
rows_by_date = defaultdict(list)
for row in rows:
    rows_by_date[row['date']].append(row)
```

è£ŹæăůçŽĎèŕİä¡ăåŔŕäžěå¿Ĺè¡żæİ¿çŽĎåřśèČ¡åŕźæŕŘäÿłæŇĞåőŽæŮěæIJ§èő£éŮőåŕźåžŤçŽĎèőřå¡ŢïijŽ

```
>>> for r in rows_by_date['07/01/2012']:
...     print(r)
...
{'date': '07/01/2012', 'address': '5412 N CLARK'}
```

```
{'date': '07/01/2012', 'address': '4801 N BROADWAY'}
>>>
```

åIJÍäÿŁéİćè£ŹäÿłäżÑå■ŘäÿïijŇæĹŚäżňæšąæIJL’å£ĚèęĄåĚĺářĘèőřå¡ŢæŐŠåžŔãĂĆåŻäæ■ď’ïijŇåęĆæđè£Źçğ■æŰźåijŔäijŽæŕŤåĚĺæŐŠåžŘçĎűåŘŐåĘ■éĂŽè£Ğ groupby() åĞ¡æŢřè£■äżčçŽĎæŰźåijŔè£ŘèąŇå¿Ůå£ńäÿĂäžŻãĂĆ

1.16 è£Ğæżď’åžŔåĹŮåĚČçť’ă

éŮőécŸ

ä¡ăæIJL’äÿĂäÿłæŢřæ■őåžŔåĹŮïijŇæČşåĹľçŤĺäÿĂäžŻèğĎåĹŹäżŐäÿ■æŔŘåŔŮåĞžéIJĂèęĄçŽĎåĂijæĺ

èğčåĘşæŰźæąĹ

æIJĂçőĂå■ŢçŽĎè£ĞæżďˇåžŔåĹŮåĚČçť’ăçŽĎæŰźæşŢářśæŸŕă¡£çŤĺåĹŮèąĺæŐĺářijãĂĆæŕŤåęĆïijŽ

```
>>> mylist = [1, 4, -5, 10, -7, 2, 3, -1]
>>> [n for n in mylist if n > 0]
[1, 4, 10, 2, 3]
>>> [n for n in mylist if n < 0]
[-5, -7, -1]
>>>
```

ä¡£çŤĺåĹŮèąĺæŐĺářijçŽĎäÿĂäÿłæ¡IJåIJÍçijžéŹůářśæŸŕåęĆæđIJè¿ŞåĚěéİđåÿÿåď’ğçŽĎæŮűåĂŹäijŽäžğçåęĆæđIJä¡ăářźåĘĚå■ŸæŕŤè¿ČæŢŔæĎ§ïijŇéĆčäźĹä¡ăåŔŕäžěä¡£çŤĺçŤ§æĹŘåŹĺèąĺè¿¿åijŔè£■äżčäžğçŤ§è£Ğ

```
>>> pos = (n for n in mylist if n > 0)
>>> pos
<generator object <genexpr> at 0x1006a0eb0>
>>> for x in pos:
... print(x)
...
1
4
10
2
3
>>>
```

æIJL’æŮűåĂŹïijŇè£Ğæżď’èğĎåĹŹæŕŤè¿Čåď’■æİĆïijŇäÿ■èČ¡çőĂå■ŢçŽĎåIJĺåĹŮèąĺæŐĺářijæĹŰèĂĚçæŕŤåęĆïijŇåĄĞèő¿è£Ğæżď’çŽĎæŮűåĂŹéIJĂèęĄåď’ĎçŘĘäÿĂäžŻåijĆåÿÿæĹŰèĂĚåĚűäżŰåď’■æİĆæČĚåĘçĎűåŘŐä¡£çŤĺåĘĚåžžçŽĎ filter() åĞ¡æŢřãĂĆçď’ža¿ŇåęĆäÿŇïijŽ

```
values = ['1', '2', '-3', '-', '4', 'N/A', '5']
def is_int(val):
    try:
```

```
        x = int(val)
        return True
    except ValueError:
        return False
ivals = list(filter(is_int, values))
print(ivals)
# Outputs ['1', '2', '-3', '4', '5']
```

filter() åĞ¡æŢřåĹŻåżžäžĘäÿĂäÿłè£■äžčåŹĺïijŇåŻäæ■ď åęĆæđIJä¡ăæČşå¿ŮåĹřäÿĂäÿłåĹŮèąĺçŽĎè
list() åŐżè¡ňæ■ćãĂĆ

èőĺèőž

åĹŮèąĺæŐĺåŕijåŠŇçŤ§æĹŘåŹĺèąĺè¿¿åijŔéĂŽåÿÿæČĚåĘţäÿŃæŸŕè£ĞæżďæŢřæ■őæIJĂçőĂå■ŢçŽĎæŰ
åĚűåőđåőČäžňè£ŸèČ¡åIJĺè£ĞæżďçŽĎæŮűåĂŹè¡ňæ■ćæŢřæ■õãĂĆærŤåęĆïijŽ

```
>>> mylist = [1, 4, -5, 10, -7, 2, 3, -1]
>>> import math
>>> [math.sqrt(n) for n in mylist if n > 0]
[1.0, 2.0, 3.1622776601683795, 1.4142135623730951, 1.
↪7320508075688772]
>>>
```

è£ĞæżďæŞ■ä¡IJçŽĎäÿĂäÿłåŔŸçğ■åřśæŸŕåřĘäÿ■çňęåŘĹæİąäžűçŽĎåĂijçŤĺæŰřçŽĎåĂijäžčæŻ£ïijŇèĂ
ærŤåęĆïijŇåIJĺäÿĂåĹŮæŢřæ■őäÿ■ä¡ăåŔŕèČ¡äÿ■äžĚæČşæL'¿åĹřæ■čæŢřïijŇèĂŇäÿŤè£ŸæČşåřĘäÿ■æŸŕæ■
éĂŽè£ĞåřĘè£Ğæżď æİąäžűæŤ¿åĹřæİąäžűèąĺè¿¿åijŔäÿ■åŐżïijŇåŕäžěå¿Ĺåőźæ ŸŞçŽĎèğčåĘşè£ŹäÿłéŮőécŸ

```
>>> clip_neg = [n if n > 0 else 0 for n in mylist]
>>> clip_neg
[1, 4, 0, 10, 0, 2, 3, 0]
>>> clip_pos = [n if n < 0 else 0 for n in mylist]
>>> clip_pos
[0, 0, -5, 0, -7, 0, 0, -1]
>>>
```

åŔęåď ŰäÿĂäÿłåĂijå¿ŮåĚşæşĺçŽĎè£Ğæżď åůěåĚůåřśæŸŕ itertools.
compress() ïijŇ åőČäžëäÿĂäÿł iterable åŕźèśąåŠŇäÿĂäÿłçŽÿåŕźåžŤçŽĎ
Boolean éĂĽæŃľåŹĺåžŔåĹŮä¡IJäÿžè¿ŞåĚěåŔĆæŢřãĂĆ çĎŮåŘŐè¿ŞåĞž
iterable åŕźèśąäÿ■åŕźåžŤéĂĽæŃľåŹĺäÿž True çŽĎåĚČçť ăãĂĆ
å¡Şä¡ăéIJĂèęĄçŤĺåŔęåď ŰäÿĂäÿłçŽÿåĚşèĄŢçŽĎåžŔåĹŮæİěè£Ğæżď æ§Räÿłåž ŔåĹŮçŽĎæŮűåĂŹïijŇè£Źä
ærŤåęĆïijŇåĄĞåęĆçŐřåIJĺä¡ăæIJL'äÿŃéİćäÿďåĹŮæŢřæ■őïijŽ

```
addresses = [
    '5412 N CLARK',
    '5148 N CLARK',
    '5800 E 58TH',
    '2122 N CLARK',
    '5645 N RAVENSWOOD',
    '1060 W ADDISON',
```

```
    '4801 N BROADWAY',
    '1039 W GRANVILLE',
]
counts = [ 0, 3, 10, 4, 1, 7, 6, 1]
```

çŐřåIJÍä¡ăæČşåřĘéĆčäžŻåŕźåžŤ `count` åĂijåď´gäžŐ5çŽĎåIJřåİĂåĚÍéČÍè¿ŞåĞžïijŇéĆčäźĹä¡ääŔŕäžěè£

```
>>> from itertools import compress
>>> more5 = [n > 5 for n in counts]
>>> more5
[False, False, True, False, False, True, True, False]
>>> list(compress(addresses, more5))
['5800 E 58TH', '1060 W ADDISON', '4801 N BROADWAY']
>>>
```

è£ŹéĞŇçŽĎåĚşéŤőçĆźåIJÍäžŐåĚĹåĹŻåžžäÿĂäÿł Boolean åžŔåĹŮïijŇæŇĞçď´žåŞłäžŻåĚČçť´ăçňęåŘĹæİąäżűãĂĆ çĎűåŘŐ `compress()` åĞ¡æŢřæăźæ■őè£ŹäÿłåžŔåĹŮåŐžéĂĽ´æŃľ´è¿ŞåĞžåŕźåžŤä¡■ç¡őäÿž `True` çŽĎåĚČçť´ãäĂĆ

åŠŇ `filter()` åĞ¡æŢřçśżäijijïijŇ `compress()` äź§æŸŕè£ŤåŻđçŽĎäÿĂäÿłè£■äżčåŹĺãĂĆåŻăæ■d´ïij éĆčäźĹä¡ăéIJĂèęĄä¡£çŤÍ `list()` æİěåřĘçżŞæđIJè¡ñæ■ćäÿžåĹŮèąĺçśżåđŃãĂĆ

1.17 äżŐå■ŮåĚÿäÿ■æŔŘåŔŰå■ŘéŻĘ

éŮőécŸ

ä¡ăæČşæđĎéĂääÿĂäÿłå■ŮåĚÿïijŇåőČæŸŕåŔęåď´ŰäÿĂäÿłå■ŮåĚÿçŽĎå■ŘéŻEãĂĆ

èğčåĘşæŰźæąĹ

æIJĂçőĂå■ŢçŽĎæŰźåijŔæŸŕä¡£çŤĺå■ŮåĚÿæŐĺåŕijãĂĆæŕŤåęĆïijŽ

```
prices = {
    'ACME': 45.23,
    'AAPL': 612.78,
    'IBM': 205.55,
    'HPQ': 37.20,
    'FB': 10.75
}
# Make a dictionary of all prices over 200
p1 = {key: value for key, value in prices.items() if value > 200}
# Make a dictionary of tech stocks
tech_names = {'AAPL', 'IBM', 'HPQ', 'MSFT'}
p2 = {key: value for key, value in prices.items() if key in tech_
↪names}
```

èőÍèőž

åď ğåď ŽæŢřæČĚåĘţäÿŃå■ŮåĚÿæŐÍåřijèČ¡åĄŽåĹřçŽĎïijŇéĂŽè£ĞåĹŻåžžäÿĂäÿłåĚČçżĎåžŔåĹŮçĎű
dict() åĞ¡æŢřäź§èČ¡åőđçŐřãĂĆæŕŤåęĆïijŽ

```
p1 = dict((key, value) for key, value in prices.items() if value >␣
↪200)
```

ä¡ĘæŸŕïijŇå■ŮåĚÿæŐÍåřijæŰźåijŔèąÍæĎŔæŻť æÿĚæŹřïijŇåźűäÿŤåőđéŹĚäÿŁäź§äijŽè£ŘèąŇçŽĎæŻť
(åIJÍè£ŹäÿŁä¿Ńå■ŘäÿïijŇåőđéŹĚæţŃèŕŢåĞäääŐæŕŤ dcit()
åĞ¡æŢřæŰźåijŔå£ńæŢť æŢť äÿĂåĂ■)ãĂĆ

æIJL'æŮűåĂŹåőŇæĹŘåŘŇäÿĂäžűäžŇäijŽæIJL'åď ŽçğæŰźåijŔãĂĆæŕŤåęĆïijŇçňňäžŇäÿłä¿Ńå■ŘçÍŃ

```
# Make a dictionary of tech stocks
tech_names = { 'AAPL', 'IBM', 'HPQ', 'MSFT' }
p2 = { key:prices[key] for key in prices.keys() & tech_names }
```

ä¡ĘæŸŕïijŇè£ŘèąŇæŮűéŮť æţŃèŕŢçżŞæđIJæŸ¿çď žè£Źçğ■æŰźæąĹåď ğæęĆæŕŤçňňäÿĂçğ■æŰźæąĹæ
åęĆæđIJåŕźçÍŃåžŔè£ŘèąŇæĂğèČ¡èęĄæśĆæŕŤè¿ČéńŸçŽĎèŕİïijŇéIJĂèęĄèŁśçĆźæŮűéŮť åŐżåĄŽèőąæŮűa
åĚşäžŐæŻť åď ŽèőąæŮűåŠŇæĂğèČ¡æţŃèŕŢïijŇåŔŕäžěåŔĆèĂČ14.13årŔèŁĆ

1.18 æŸäåřĎåŘ■çğřåĹřåžŔåĹŮåĚČçťă

éŮőécŸ

ä¡ăæIJL'äÿĂæőţéĂŽè£ĞäÿŃæăĞèő£éŮőåĹŮèąÍæĹŰèĂĚåĚČçżĎäÿ■åĚČçť'ăçŽĎäžčçăĄïijŇä¡ĘæŸŕè£Ź
äžŐæŸŕä¡ăæČşéĂŽè£ĞåŘ■çğřæİèèő£éŮőåĚČçť'ăãĂĆ

èğčåĘşæŰźæąĹ

collections.namedtuple() åĞ¡æŢřéĂŽè£Ğä¡£çŤĺäÿĂäÿłæŹőéĂŽçŽĎåĚČçżĎårźèśąæİěåÿőä¡ă
è£ŹäÿłåĞ¡æŢřåőđéŹĚäÿŁæŸŕäÿĂäÿłè£ŤåŽđPythonäÿ■æăĞåĞĘåĚČçżĎçśžåđŃå■ŘçśžçŽĎäÿĂäÿłåůěåŐĆæ
ä¡ăéIJĂèęĄäijăéĂŠäÿĂäÿłçśžåđŃåŘ■åŠŇä¡ăéIJĂèęĄçŽĎå■ŮæőţçżŹåőČïijŇçĎűåŘŐåőČårśäijŽè£ŤåŽđäÿĂ
äžčçăĄçď žä¿ŃïijŽ

```
>>> from collections import namedtuple
>>> Subscriber = namedtuple('Subscriber', ['addr', 'joined'])
>>> sub = Subscriber('jonesy@example.com', '2012-10-19')
>>> sub
Subscriber(addr='jonesy@example.com', joined='2012-10-19')
>>> sub.addr
'jonesy@example.com'
>>> sub.joined
'2012-10-19'
>>>
```

åř¡çőą `namedtuple` çŽĎåőđä¿ŃçIJŃèţůæİěåČŕäÿĂäÿłæŹőéĂŽçŽĎçśžåőđä¿ŃïijŇä¡ĘæŸŕåőČèů§åĚČ
ærŤåęĆïijŽ

```
>>> len(sub)
2
>>> addr, joined = sub
>>> addr
'jonesy@example.com'
>>> joined
'2012-10-19'
>>>
```

åŠ¡åŘ■åĚČçżĎçŽĎäÿĂäÿłäÿżèęĄçŤĺéĂŤæŸŕåřĘä¡ăçŽĎäżčçăĄäžŐäÿŃæăĞæŞ■ä¡IJäÿ■èğčèĎśåĞžæİě
åŻăæ■ď¨ïijŇåęĆædIJä¡ăäžŐæŢřæ■őåžŞèřČçŤĺäÿ■è£ŤåŻđäžĘäÿĂäÿłå¿Ĺåď¨ğçŽĎåĚČçżĎåĹÙèąĺïijŇéĂŽè£Ğ
å¡Şä¡ăåIJĺèąĺäÿ■æůżåŁäžĘæŰřçŽĎåĹÙçŽĎæÙűåĂŹä¡ăçŽĎäżčçăĄåŘŕèČ¡åŕšäijŽåĞžéŤŹäžĘãĆä¡ĘæŸŕå

äÿžäžĘèŕť¨æŸŐæÿĚæěŽïijŇäÿŃéİćæŸŕä¡£çŤĺæŹőéĂŽåĚČçżĎçŽĎäżčçăĄïijŽ

```
def compute_cost(records):
    total = 0.0
    for rec in records:
        total += rec[1] * rec[2]
    return total
```

äÿŃæăĞæŞ■ä¡IJéĂŽåÿÿäijŽèől¨äžčçăĄèąĺæĎŔäÿ■æÿĚæŹřijŇåźűäÿŤéİđåÿÿä¿ÌèţŰèőřå¡ŢçŽĎçżŞæđĎ
äÿŃéİćæŸŕä¡£çŤĺåŠ¡åŘ■åĚČçżĎçŽĎçL¨ĹæIJñïijŽ

```
from collections import namedtuple

Stock = namedtuple('Stock', ['name', 'shares', 'price'])
def compute_cost(records):
    total = 0.0
    for rec in records:
        s = Stock(*rec)
        total += s.shares * s.price
    return total
```

èőÍèőž

åŠ¡åŘ■åĚČçżĎåŕęäÿĂäÿłçŤĺéĂŤåřśæŸŕä¡IJäÿžå■ŮåĚÿçŽĎæŻ£äžčijŇåŻääÿžå■ŮåĚÿå■ŸåĆĺéIJĂèęĄ
åęĆædIJä¡ăéIJĂèęĄæđĎåżžäÿĂäÿłéİđåÿÿåď¨ğçŽĎåŇĚåŘńå■ŮåĚÿçŽĎæŢřæ■őçżŞæđĎïijŇéĆčäźĹä¡£çŤĺåŠ¡
ä¡ĘæŸŕéIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇäÿ■åČŔå■ŮåĚÿéĆčæăůïijŇäÿĂäÿłåŠ¡åŘ■åĚČçżĎæŸŕäÿ■åŘŕæŻť¨æŤźç

```
>>> s = Stock('ACME', 100, 123.45)
>>> s
Stock(name='ACME', shares=100, price=123.45)
>>> s.shares = 75
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
```

```
AttributeError: can't set attribute
>>>
```

åęĆædIJä¡ăçIJ§çŽĎéIJÄèęĄæŤźåŔŸåśđæĂğçŽĎåÄijïijŇéĆčäźĹåŔŕäżěä¡£çŤĺåŚ¡åŘ■åĚČçżĎåőđä¿ŃçŽ
_replace() æŰźæşŢïijŇ åőČäijŽåĹŻåžžäÿĂäÿłåĚĺæŰřçŽĎåŚ¡åŘ■åĚČçżĎåźűårĘåŕźåžŤçŽĎå■Ůæőţç
```
>>> s = s._replace(shares=75)
>>> s
Stock(name='ACME', shares=75, price=123.45)
>>>
```

_replace() æŰźæşŢè£ŸæIJL'äÿĂäÿłå¿ĹæIJL'çŤĺçŽĎçL'źæĂğåřśæŸŕå¡Şä¡ăçŽĎåŚ¡åŘ■åĚČçżĎæŇ
åőČæŸŕäÿĂäÿłéİđåÿÿæŰźä¿£çŽĎåąńåĚĚæŢřæ■őçŽĎæŰźæşŢãĂĆ
ä¡ååŔŕäżěåĚĹåĹŻåžžäÿĂäÿłåŇĚåŘńçijžçIJĄåÄijçŽĎåŐ§åđŃåĚČçżĎïijŇçĎűåŘŐä¡£çŤĺ
_replace() æŰźæşŢåĹŻåžžæŰřçŽĎåÄijècńæŻť'æŰrè£ĞçŽĎåőđä¿ŃãĂĆæŕŤåęĆïijŽ

```
from collections import namedtuple

Stock = namedtuple('Stock', ['name', 'shares', 'price', 'date',
↪'time'])

# Create a prototype instance
stock_prototype = Stock('', 0, 0.0, None, None)

# Function to convert a dictionary to a Stock
def dict_to_stock(s):
    return stock_prototype._replace(**s)
```

äÿŃéİćæŸŕåőČçŽĎä¡£çŤĺæŰźæşŢïijŽ

```
>>> a = {'name': 'ACME', 'shares': 100, 'price': 123.45}
>>> dict_to_stock(a)
Stock(name='ACME', shares=100, price=123.45, date=None, time=None)
>>> b = {'name': 'ACME', 'shares': 100, 'price': 123.45, 'date':
↪'12/17/2012'}
>>> dict_to_stock(b)
Stock(name='ACME', shares=100, price=123.45, date='12/17/2012',␣
↪time=None)
>>>
```

æIJĂåŘŐèęĄèŕť'çŽĎæŸŕïijŇåęĆædIJä¡ăçŽĎçŽőæăĞæŸŕåőŽäźL'äÿĂäÿłéIJĂèęĄæŻť'æŰřå¿Ĺåď'Žåőđä¿
è£ŹæŮűåĂŹä¡ăåžŤèŕěèĂČèŹŚåőŽäźL'äÿĂäÿłåŇĚåŘń `__slots__`
æŰźæşŢçŽĎçśż(åŔĆèĂČ8.4åřŔèŁĆ)ãĂĆ

1.19 è¡ňæ■ćåźűåŘŇæŮűèőąçőŮæŢřæ■ő

éŮőécŸ

ä¡ăéIJĂèęĄåIJÍæŢřæ■őåžŔåĹŮäÿŁæĽğèąŇèĄŽéŻĘåĞ¡æŢř(æŕŤåęĆ `sum()` , `min()` , `max()`)ïijŇ ä¡ĘæŸŕéęŰåĚĹä¡ăéIJĂèęĄåĚĹè¡ňæ■ćæĹŰèĂĚè£ĞæżďæŢřæ■ő

èğčåĘşæŰźæąĹ

äÿĂäÿłéİđåÿÿäijŸéŻĚçŽĎæŰźåijŔåŐżçżŞåŘĹæŢřæ■őèőąçőŮäÿŐè¡ňæ■ćåřśæŸŕä¡£çŤĺäÿĂäÿłçŤ§æĹŘæŕŤåęĆïijŇåęĆæđIJä¡ăæČşèőąçőŮåźşæŰźåŠŇïijŇåŔŕäžěåČŔäÿŃéİćè£ŹæăůåĄŽïijŽ

```
nums = [1, 2, 3, 4, 5]
s = sum(x * x for x in nums)
```

äÿŃéİćæŸŕæŻťåďŽçŽĎä¿Ńå■ŘïijŽ

```
# Determine if any .py files exist in a directory
import os
files = os.listdir('dirname')
if any(name.endswith('.py') for name in files):
    print('There be python!')
else:
    print('Sorry, no python.')
# Output a tuple as CSV
s = ('ACME', 50, 123.45)
print(','.join(str(x) for x in s))
# Data reduction across fields of a data structure
portfolio = [
    {'name':'GOOG', 'shares': 50},
    {'name':'YHOO', 'shares': 75},
    {'name':'AOL', 'shares': 20},
    {'name':'SCOX', 'shares': 65}
]
min_shares = min(s['shares'] for s in portfolio)
```

èőÍèőž

äÿŁéİćçŽĎçđ'žä¿ŃåŘŚä¡ăæijŤçđ'žäžĘå¡ŞçŤ§æĹŘåŹÍèąÍè¿¿åijŔä¡IJäÿžäÿĂäÿłå■ŢçŃňåŔĆæŢřäijăéĂŠç
æŕŤåęĆïijŇäÿŃéİćè£ŹäžŻèŕ■åŘěæŸŕç■Ľ'æŢĹçŽĎïijŽ

```
s = sum((x * x for x in nums)) #␣
↪æŸ¿çd'žçŽĎäijăéĂŠäÿĂäÿłçŤ§æĹŘåŹÍèąÍè¿¿åijŔårźèšą
s = sum(x * x for x in nums) #␣
↪æŻt'åŁääijŸéŻĚçŽĎåőđçŐřæŰźåijŔïijŇçIJĄçŢěäžĘąŃňåŔů
```

ä¡£çŤĺäÿĂäÿłçŤ§æĹŘåŹÍèąÍè¿¿åijŔä¡IJäÿžåŔĆæŢřäijŽæŕŤåĚĹåĹŻåżžäÿĂäÿłäÿť'æŮűåĹŮèąÍæŻť'åŁăéŕ
æŕŤåęĆïijŇåęĆæđIJä¡ääÿ■ä¡£çŤĺçŤ§æĹŘåŹÍèąÍè¿¿åijŔçŽĎèŕÏïijŇä¡ăåŔŕèČ¡äijŽèĂČèŹŚä¡£çŤĺäÿŃéİćçŽĎå

```
nums = [1, 2, 3, 4, 5]
s = sum([x * x for x in nums])
```

è£Źçğ■æŰźåijŔåŘŇæăůåŘŕäžěè¿¿åĹřæČşèęĄçŽĎæŢĹæđIJïijŇä¡ĘæŸŕåőČäijŽåď'ŽäÿĂäÿłæ■ěéłď'ïijŇå
åŕźäžŐåřŘåđŃåĹŮèąĺåŘŕèČ¡æšąäżĂäźĹåĚşçşżïijŇä¡ĘæŸŕåęĆæđIJåĚČçť'ăæŢřéĞŔéİđåÿÿåď'ğçŽĎæŮűåĂŹi
åőČäijŽåĹŻåžžäÿĂäÿłåůĺåď'ğçŽĎäżĚäżĚèćńä¡£çŤĺäÿĂæňąåřśèćńäÿćåijČçŽĎäÿt'æŮűæŢřæ■őçżŞæđĎãĂĆè

åIJĺä¡£çŤĺäÿĂäžŽèĄŽéŻĘåĞ¡æŢřæŕŤåęĆ `min()` åŠŇ `max()` çŽĎæŮűåĂŹä¡ăåŔŕèČ¡æŻt'åŁăåĂ¿åŘŚäžŐä¡£çŤĺçŤ§æĹŘåŹĺçL'ĹæIJňïijŇ
åőČäžňæŐěåŔŮçŽĎäÿĂäÿłkeyåĚşéŤőå■ŮåŔĆæŢřæĹŰèőÿåŕźä¡ăå¿ĹæIJL'åÿőåŁl'ãĂĆ
æŕŤåęĆïijŇåIJĺäÿŁéİćçŽĎèŕĄåĹÿä¿Ńå■Ŕäÿ■ïijŇä¡ăåŔŕèČ¡äijŽèĂČèŹŚäÿŃéİćçŽĎåőđçŐřçL'ĹæIJňïijŽ

```
# Original: Returns 20
min_shares = min(s['shares'] for s in portfolio)
# Alternative: Returns {'name': 'AOL', 'shares': 20}
min_shares = min(portfolio, key=lambda s: s['shares'])
```

1.20 åŘĹåźűåď'Žäÿłå■ŮåĚÿæĹŰæŸäåřĎ

éŮőécŸ

çŐřåIJĺæIJL'åď'Žäÿłå■ŮåĚÿæĹŰèĂĚæŸäåřĎïijŇä¡ăæČşåřĘåőČäžňäžŐéĂżè¿ŚäÿŁåŘĹåźűäÿžäÿĂäÿłå■
æŕŤåęĆæ§ěæL'¿åĂijæĹŰèĂĚæčĂæ§ěæ§ŘäžŹéŤőæŸŕåŘęå■ŸåIJĺãĂĆ

èğčåĘşæŰźæąĹ

åĄĞåęĆä¡ăæIJL'åęĆäÿŃäÿď'äÿłå■ŮåĚÿ:

```
a = {'x': 1, 'z': 3 }
b = {'y': 2, 'z': 4 }
```

çŐřåIJĺåĄĞèő¿ä¡ăå£Ěéążå IJĺäÿď'äÿłå■ŮåĚÿäÿ■æL'ğèąŇæ§ěæL'¿æŞ■ä¡IJ(æŕŤåęĆåĚĹäžŐ
`a` äÿ■æL'¿ïijŇåęĆæđIJæL'¿äÿ■åĹřåĘ■åIJĺ `b` äÿ■æL'¿)ãĂĆ
äÿĂäÿłéİđåÿÿçőĂå■ŢçŽĎèğčåĘşæŰźæąĹåřśæŸŕä¡£çŤĺ `collections` æĺąåİŮäÿ■çŽĎ
`ChainMap` çśžãĂĆæŕŤåęĆïijŽ

```
from collections import ChainMap
c = ChainMap(a,b)
print(c['x']) # Outputs 1 (from a)
print(c['y']) # Outputs 2 (from b)
print(c['z']) # Outputs 3 (from a)
```

èőĺèőž

äÿĂäÿł `ChainMap` æŐěåŔŮåď'Žäÿłå■ŮåĚÿåźűåřĘåőČäžňåIJĺéĂżè¿ŚäÿŁåŔŸäÿžäÿĂäÿłå■ŮåĚÿãĂĆ
çĎűåŘŐïijŇè£ŹäžŻå■ŮåĚÿåźűäÿ■æŸŕçIJ§çŽĎåŘĹåźűåIJĺäÿĂèţůäžĘïijŇ `ChainMap`

çśżåŔłæŸŕåIJĺåĘĚéČĺåĹŻåżžäžĘäÿĂäÿłåőźçžşè£ŹäžŻå■ŮåĚÿçŽĎåĹŮèąĺ
åźűéĞ■æŰřåőŽäźL’äžĘäÿĂäžŻåÿÿèğAçŽĎå■ŮåĚÿæŞ■ä¡IJæİěéĄ■åŐĘè£ŹäÿłåĹŮèąĺãĂĆåď’ğéČĺåĹĘå■ŮåĚ

```
>>> len(c)
3
>>> list(c.keys())
['x', 'y', 'z']
>>> list(c.values())
[1, 2, 3]
>>>
```

åęĆæđIJåĞžçŐřéĞ■åď’■éŤőïijŇéĆčäźĹçňňäÿĂæňąåĞžçŐřçŽĎæŸăářĎåĂijäijŽèćńè£ŤåŻđãĂĆ åŻăæ■ď’ïijŇä¿Ńå■ŘçĺŃåžŔäÿ■çŽĎ `c['z']` æĂżæŸŕäijŽè£ŤåŻđå■ŮåĚÿ `a` äÿ■åŕźåžŤçŽĎåĂijïijŇèĂŇäÿ■æŸŕ `b` äÿ■åŕźåžŤçŽĎåĂijãĂĆ

åŕźäžŐå■ŮåĚÿçŽĎæŻt’æŰřæĹŰåĹăéŹd’æŞ■ä¡IJæĂżæŸŕå¡śåŞ■çŽĎæŸŕåĹŮèąĺäÿ■çňňäÿĂäÿłå■ŮåĚÿă

```
>>> c['z'] = 10
>>> c['w'] = 40
>>> del c['x']
>>> a
{'w': 40, 'z': 10}
>>> del c['y']
Traceback (most recent call last):
...
KeyError: "Key not found in the first mapping: 'y'"
>>>
```

`ChainMap` åŕźäžŐçijŰçĺŃèŕ■èĺĂäÿ■çŽĎä¡IJçŤĺèŇČåŻt’åŔŸéĞŔ(æŕŤåęĆ `globals`, `locals` ç■Ľ)æŸŕéİđåÿÿæIJL’çŤĺçŽĎãĂĆ äžŃåőđäÿŁïijŇæIJL’äÿĂäžŻæŰźæşŢåŔŕäžëä¡£åőČåŔŸå¿ŮçőĂå

```
>>> values = ChainMap()
>>> values['x'] = 1
>>> # Add a new mapping
>>> values = values.new_child()
>>> values['x'] = 2
>>> # Add a new mapping
>>> values = values.new_child()
>>> values['x'] = 3
>>> values
ChainMap({'x': 3}, {'x': 2}, {'x': 1})
>>> values['x']
3
>>> # Discard last mapping
>>> values = values.parents
>>> values['x']
2
>>> # Discard last mapping
>>> values = values.parents
>>> values['x']
1
>>> values
```

```
ChainMap({'x': 1})
>>>
```

ä¡IJäÿž `ChainMap` çŽĎæŻ£äżčïijŇä¡ăåŔŕèČ¡äijŽèĂČèŹŚä¡£çŤÍ `update()` æŰźæşŢåřĘäÿď äÿłå■ŮåĚÿåŘĹåźűãĂĆæŕŤåęĆïijŽ

```
>>> a = {'x': 1, 'z': 3 }
>>> b = {'y': 2, 'z': 4 }
>>> merged = dict(b)
>>> merged.update(a)
>>> merged['x']
1
>>> merged['y']
2
>>> merged['z']
3
>>>
```

è£Źæăůäź§èČ¡èąŇå¿ŮéĂŽïijŇä¡ĘæŸŕåőČéIJÄèęĄä¡ăåĹŻåżžäÿĂäÿłáőŇåĚíäÿ■åŘŇçŽĎå■ŮåĚÿåŕźèśą(
åŘŇæŮűïijŇåęĆæđIJåŐ§å■ŮåĚÿåĄŽäžĘæŻť æŰŕïijŇè£Źçğ■æŤźåŔŸäÿ■äijŽåŔ■åžŤåĹŕæŰřçŽĎåŘĹåźűå■Ů

```
>>> a['x'] = 13
>>> merged['x']
1
```

`ChainMap` ä¡£çŤĺåŐ§æİěçŽĎå■ŮåĚÿïijŇáőČèĞłåůśäÿ■åĹŻåżžæŰřçŽĎå■ŮåĚÿãĂĆæL'ĂäžěáőČåźűäÿ

```
>>> a = {'x': 1, 'z': 3 }
>>> b = {'y': 2, 'z': 4 }
>>> merged = ChainMap(a, b)
>>> merged['x']
1
>>> a['x'] = 42
>>> merged['x'] # Notice change to merged dicts
42
>>>
```

çňňäžŇçńăïijŽå■ŮçňęäÿšåŠŇæŰĞæIJň

åĞääźŐæL'ĂæIJL'æIJL'çŤĺçŽĎçĺŇåžŔéČ¡äijŽæűL'åŔŁåĹřæ§ŘäžŻæŰĞæIJňåď'ĎçŘĘïijŇäÿ■çőąæŸŕèğ
è£ŹäÿĂçńăářĘéĞ■çĆźåĚşæşĺæŰĞæIJňçŽĎæŞ■ä¡IJåď'ĎçŘĘïijŇæŕŤåęĆæŔŘåŘŰå■ŮçňęäÿšïijŇæŘIJçť ćïijŇ
åď'ğéČĺåĹĘçŽĎéŮőécŸéČ¡èČ¡çőĂå■ŢçŽĎèřČçŤĺå■ŮçňęäÿšçŽĎåĘĚåžžæŰźæşŢåőŇæĹŘãĂĆ
ä¡ĘæŸŕïijŇäÿĂäžŻæŻť äÿžåď'■æİĆçŽĎæŞ■ä¡IJåŔŕèČ¡éIJÄèęĄæ■čåĹŹèąĺè¿¿åijŔæĹŰèĂĚåijžåď'ğçŽĎèğčæ
åźűäÿŤåIJĺæŞ■ä¡IJUnicodeæŮűåĂŹçćřåĹřçŽĎäÿĂäžŻæčŸæL'ŇçŽĎéŮőécŸåIJĺè£ŹéĞŇäź§äijŽèćńæŔŘåŔ

Contents:

2.1 ä¡£çŤĺåďŽäÿłçŢŇåőŽçňęåĹĘåĽšå■Ůçňęäÿš

éŮőécŸ

ä¡ăéIJĂèęĄåřĘäÿĂäÿłå■Ůçňęäÿšå[ĹĘåĽšäÿžåď'ŽäÿłåĽ■Ůæőţïij Ñä¡ĘæŸřåĹĘéŽŤçňę(è£ŸæIJL'åŚĺåŻť'çŽĹ

èğčåĘşæŰźæąĹ

`string` åŕźèśąçŽĎ `split()` æŰźæşŢåŔłéĂĆåžŤäžŐéİđåÿÿçőĂå■ŢçŽĎå■Ůçňęäÿšå[ĹĘåĽšæČĚå¡ćïij åőČåźűäÿ■åĚĄèőÿæIJL'åď'Žäÿłå[ĹĘéŽŤçňęæĹŰèĂĚæŸřå[ĹĘéŽŤçňęåŚĺåŻť'äÿ■çąőåőŽçŽĎçl'žæăijãĂĆ å¡Şä¡ăéIJĂèęĄæŻť'åŁăçĄţæť'żçŽĎåĹĞåĽ'šå■Ůçňęäÿšç ŽĎæŮűåĂŹïijÑæIJĂåě¡ä¡£çŤĺ `re.split()` æŰźæşŢïijŽ

```
>>> line = 'asdf fjdk; afed, fjek,asdf, foo'
>>> import re
>>> re.split(r'[;,\s]\s*', line)
['asdf', 'fjdk', 'afed', 'fjek', 'asdf', 'foo']
```

èőĺèőž

åĞ¡æŢř `re.split()` æŸŕéİđåÿÿåőđçŤĺçŽĎïijÑåŻääÿžåőČåĚĄèőÿä¡ääÿžåĹĘéŽŤçňęæŇĞåőŽåď'Žäÿł æŕŤåęĆïijÑåIJĺäÿŁéİćçŽĎä¿Ńå■Řäÿ■ïijÑåĹĘéŽŤçňęåŔŕäžěæŸŕéĂŮåŔůïijÑåĹĘåŔůæĹŰèĂĚæŸŕçl'žæăijïij åŔłèęĄè£ŹäÿłæĺąåijŔèćńæL'¿åĹřïijÑéĆčäźĹåŇźéĚ■çŽĎåĹĘéŽŤçňęäÿď'è¿źçŽĎåőđä¡ŞéČ¡äijŽèćńå¡ŞæĹŘæ è£ŤåŻđçżŞæđIJäÿžäÿĂäÿłå■ŮæőţåĹŮèąĺïijÑè£Źäÿłèů§ `str.split()` è£ŤåŻđåĂijçśžåđŃæŸŕäÿĂæăůçŽĎãĂĆ

å¡Şä¡ää¡£çŤĺ `re.split()` åĞ¡æŢřæŮűåĂŹïijÑéIJĂèęĄçL'źåĹńæşĺæĎŔçŽĎæŸŕæ■čåĹŹèąĺè¿¿åijŔäÿ åęĆæđIJä¡£çŤĺäžĘæ■ŢèŐůåĹĘçżĎïijÑéĆčäźĹèćńåŇźéĚ■çŽĎæŰĞæIJñäź§ářĘåĞžçŐřåIJĺçżŞæđIJåĹŮèąĺäÿ

```
>>> fields = re.split(r'(;|,|\s)\s*', line)
>>> fields
['asdf', ' ', 'fjdk', ';', 'afed', ',', 'fjek', ',', 'asdf', ',',
↪'foo']
>>>
```

èŐůåŔŰåĹĘåĽ'šå■Ůçňęåİĺæ§ŘäžŻæČĚåĘţäÿÑäź§æŸŕæIJL'çŤĺçŽĎãĂĆ æŕŤåęĆïijÑä¡ăåŔŕèČ¡æČşä£İçŢŹåĹĘåĽ'šå■ŮçňęäÿšïijÑçŤĺæİěåIJĺåŘŐéİćéĞ■æŰřæđĎéĂăäÿĂäÿłæŰřçŽĎè

```
>>> values = fields[::2]
>>> delimiters = fields[1::2] + ['']
>>> values
['asdf', 'fjdk', 'afed', 'fjek', 'asdf', 'foo']
>>> delimiters
[' ', ';', ',', ',', ',', '']
>>> # Reform the line using the same delimiters
>>> ''.join(v+d for v,d in zip(values, delimiters))
'asdf fjdk;afed,fjek,asdf,foo'
>>>
```

åęĆæđIJä¡ăäÿ■æČşä£ÌçŢŹåĹĘåL'šå■Ůçňęäÿšå Ĺřçżşæđ IJåĹŮèąĺäÿ■åŐżïijŇä¡Ęäż■çĎűéIJĂèęĄä¡£çŤĺåĺçąőä£Ìä¡ăçŽĎåĹEçżĎæŸŕéÌđæ■ŢèŐůåĹEçżĎïijŇå¡ćåęĆ `(?:...)` ãĂĆærŤåęĆïijŽ

```
>>> re.split(r'(?:,|;|\s)\s*', line)
['asdf', 'fjdk', 'afed', 'fjek', 'asdf', 'foo']
>>>
```

2.2 å■ŮçňęäÿšåijĂåď'æĹŰçżŞář¿åŇźéĚ■

éŮőécŸ

ä¡ăéIJĂèęĄéĂŽè£ĞæŇĞåőŽçŽĎæŰĞæIJňæĺąåijŔåŐżæčĂæ§ěå■ŮçňęäÿšçŽĎåijĂåď'æĹŰèĂĚçżŞář¿ŞSchemeç■L'ç■L'ãĂĆ

èğčåĘşæŰźæąĹ

æčĂæ§ěå■ŮçňęäÿšåijĂåď'æĹŰçżŞář¿çŽĎäÿĂäÿłçőĂå■ŢæŰźæşŢæŸŕä¡£çŤĺ `str.startswith()` æĹŰèĂĚæŸŕ `str.endswith()` æŰźæşŢãĂĆærŤåęĆïijŽ

```
>>> filename = 'spam.txt'
>>> filename.endswith('.txt')
True
>>> filename.startswith('file:')
False
>>> url = 'http://www.python.org'
>>> url.startswith('http:')
True
>>>
```

åęĆæđIJä¡ăæČşæčĂæ§ěåď'Žçğ■åŇźéĚ■åŔŕèČ¡ïijŇåŔłéIJĂèęĄåřĘæL'ĂæIJL'çŽĎåŇźéĚ■éąźæŤ¿åĚěåĹçĎűåŘŐäijăçżŹ `startswith()` æĹŰèĂĚ `endswith()` æŰźæşŢïijŽ

```
>>> import os
>>> filenames = os.listdir('.')
>>> filenames
[ 'Makefile', 'foo.c', 'bar.py', 'spam.c', 'spam.h' ]
>>> [name for name in filenames if name.endswith(('.c', '.h')) ]
['foo.c', 'spam.c', 'spam.h'
>>> any(name.endswith('.py') for name in filenames)
True
>>>
```

äÿŃéÌćæŸŕåŔęäÿĂäÿłä¿Ńå■ŘïijŽ

```
from urllib.request import urlopen

```

```
def read_data(name):
    if name.startswith(('http:', 'https:', 'ftp:')):
        return urlopen(name).read()
    else:
        with open(name) as f:
            return f.read()
```

åěĞæĂłçŽĎæŸŕïijŇè£ŹäÿłæŰźæşŢäÿ■å£Ěéążèę Ąè¿ŞåĚěäÿĂäÿłåĚČçżĎä¡IJäÿžåŔĆæŢřãĂĆ åęĆæđIJä¡ăæĄřåůğæIJL'äÿĂäÿł `list` æĹŰèĂĚ `set` çśżåđŃçŽĎéĂL'æŃl'éąźïijŇ èęĄçąőä£Ïäijăé ĂŠåŔĆæŢřåL'■åĚĹèřČçŤĺ `tuple()` åřĘåĚűè¡ñæ■ćäÿžåĚČçżĎçśżåđŃãĂĆæŕŤåęĆïijŽ

```
>>> choices = ['http:', 'ftp:']
>>> url = 'http://www.python.org'
>>> url.startswith(choices)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: startswith first arg must be str or a tuple of str, not␣
 ↪list
>>> url.startswith(tuple(choices))
True
>>>
```

èőÍèőž

`startswith()` åŠŇ `endswith()` æŰźæşŢæŔŘä¿ŻäžĘäÿĂäÿłéÍđåÿÿæŰźä¿£çŽĎæŰźåijŔåŐżåĄŽå çśżäijijçŽĎæŞ■ä¡IJäź§åŔŕäžěä¡£çŤĺåĹĞçL'ĞæÍěåőđçŐřïijŇä¡ĘæŸŕäžčçăĄçIJŃèţůæÍěæšąæIJL'éĆčäźĹäijŸé

```
>>> filename = 'spam.txt'
>>> filename[-4:] == '.txt'
True
>>> url = 'http://www.python.org'
>>> url[:5] == 'http:' or url[:6] == 'https:' or url[:4] == 'ftp:'
True
>>>
```

ä¡ăåŔŕäžěèČ¡è£ŸæČşä¡£çŤĺæ■čåĹŹèąĺè¿¿åijŔåŐżåőđçŐřïijŇæŕŤåęĆïijŽ

```
>>> import re
>>> url = 'http://www.python.org'
>>> re.match('http:|https:|ftp:', url)
<_sre.SRE_Match object at 0x101253098>
>>>
```

è£Źçğ■æŰźåijŔäź§èąŇå¿ŮéĂŽïijŇä¡ĘæŸŕåŕźäžŐçőĂå■ŢçŽĎåŇźéĚ■åőđåIJĺæŸŕæIJL'çĆźåŕŔæİŘåđ'ğ

æIJĂåŘŐæŔŘäÿĂäÿŃïijŇå¡ŞåŠŇåĚűäżŰæŞ■ä¡IJæŕŤåęĆæŹőéĂŽæŢřæ■őèĄŽåŘĹçŻÿçżŞåŘĹçŽĎæŮú `startswith()` åŠŇ `endswith()` æŰźæşŢæŸŕå¿Ĺäÿ■éŤŹçŽĎãĂĆ æŕŤåęĆïijŇäÿŃéİćè£Źäÿłèŕ■åŘèæčĂæ§ěæ§ŘäÿłæŰĞäžűåđ'źäÿ■æŸŕåŘęå■ŸåIJĺæŇĞåőŽçŽĎæŰĞäžűçśżåđ

```
if any(name.endswith(('.c', '.h')) for name in listdir(dirname)):
...
```

2.3 çŤÍShelléĂŽéĚ■çňęåŇźéĚ■å■Ůçňęäÿš

éŮőécŸ

ä¡ăæČşä¡£çŤÍ **Unix Shell** äÿ■åÿÿçŤÍçŽĎéĂŽéĚ■çňę(æŕŤåęĆ `*.py`,`Dat[0-9]*.csv` ç■L')åŐżåŇźéĚ■æŰĞæIJňå■Ůçňęäÿš

èğčåĘşæŰźæąĹ

`fnmatch` æĺąåİŮæŔŘä¿ŻäžĘäÿd'äÿłåĞ¡æŢřâĂŤâĂŤ `fnmatch()` åŠŇ `fnmatchcase()` ïijŇåŔŕäżěçŤÍæİěåőđçŐřè£Źæăůç˝ŽĎåŇźéĚ■ãĂĆçŤÍæşŢåęĆäÿŃïijŽ

```
>>> from fnmatch import fnmatch, fnmatchcase
>>> fnmatch('foo.txt', '*.txt')
True
>>> fnmatch('foo.txt', '?oo.txt')
True
>>> fnmatch('Dat45.csv', 'Dat[0-9]*')
True
>>> names = ['Dat1.csv', 'Dat2.csv', 'config.ini', 'foo.py']
>>> [name for name in names if fnmatch(name, 'Dat*.csv')]
['Dat1.csv', 'Dat2.csv']
>>>
```

`fnmatch()` åĞ¡æŢřä¡£çŤÍåžŢåśĆæŞ■ä¡IJçşżçż§çŽĎåd'ğåřŔåĘŹæŢŔæĎ§èğĎåĹŹ(äÿ■åŘńçŽĎçşżçż§

```
>>> # On OS X (Mac)
>>> fnmatch('foo.txt', '*.TXT')
False
>>> # On Windows
>>> fnmatch('foo.txt', '*.TXT')
True
>>>
```

åęĆædIJä¡ăåŕźè£ŹäÿłåŇžåĹńå¿ĹåIJĺæĎŔïijŇåŔŕäżěä¡£çŤÍ `fnmatchcase()` æİěäżčæŻ£ãĂĆåőČåőŇåĚĺä¡£çŤÍä¡ăçŽĎæĺąåijŔåd'ğåřŔåĘŹåŇźéĚ■ãĂĆæŕŤåęĆïijŽ

```
>>> fnmatchcase('foo.txt', '*.TXT')
False
>>>
```

è£Źäÿd'äÿłåĞ¡æŢřéĂŽåÿÿäijŽèćńå£¡çŢěçŽĎäÿĂäÿłçL'źæĂğæŸŕåIJĺåd'ĎçŘEéİđæŰĞäžűåŘ■çŽĎå■Ůç
æŕŤåęĆïijŇåĄĞèő¿ä¡ăæIJL'äÿĂäÿłèąÛéĄŞåIJřåİĂçŽĎåĹÛèąĺæŢřæ■őïijŽ

```
addresses = [
    '5412 N CLARK ST',
    '1060 W ADDISON ST',
    '1039 W GRANVILLE AVE',
    '2122 N CLARK ST',
    '4802 N BROADWAY',
]
```

ä¡ăåŔŕäžěåČŔè£ŹæăůåĘŹåĹŮèąĺæŐĺåŕijïijŽ

```
>>> from fnmatch import fnmatchcase
>>> [addr for addr in addresses if fnmatchcase(addr, '* ST')]
['5412 N CLARK ST', '1060 W ADDISON ST', '2122 N CLARK ST']
>>> [addr for addr in addresses if fnmatchcase(addr, '54[0-9][0-9]␣
↪*CLARK*')]
['5412 N CLARK ST']
>>>
```

èőÍèőž

fnmatch() åĞ¡æŢřåŇźéĚ■èČ¡åŁŻäżŇäžŐçőĂå■ŢçŽĎå■ŮçňęäÿšæŰźæşŢåŠŇåijžåď ğçŽĎæ■čåĹŹèą åęĆæđIJåIJÍæŢřæ■őåď ĎçŘĘæŞ■ä¡IJäÿ■åŔłéIJĂèęĄçőĂå■ŢçŽĎéĂŽéĚ■çňęářśèČ¡åőŇæĹŘçŽĎæŮűåĂŹïijŇ

åęĆæđIJä¡ăçŽĎäžčçăĄéIJĂèęĄåĄŽæŰĞäžűåŘ■çŽĎåŇźéĚ■ïijŇæIJĂåě¡ä¡£çŤÍ glob æĺąåİŮãĂĆåŔĆèĂČ5.13ářŔèŁĆãĂĆ

2.4 å■Ůçňęäÿšå ŇźéĚ■åŠŇæŘIJçť ć

éŮőécŸ

ä¡ăæČşåŇźéĚ■æĹŰèĂĚæŘIJçť ćçL´źåőŽæĺąåijŔçŽĎæŰĞæIJň

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşåŇźéĚ■çŽĎæŸŕå■Ůéİćĺ■ŮçňęäÿšïijŇéĆčäźĹä¡ăéĂŽåÿÿåŔłéIJĂèęĄèŘČçŤĺå§žæIJňå■Ů æŕŤåęĆ str.find() , str.endswith() , str.startswith() æĹŰèĂĚçśžäijijçŽĎæŰźæşŢïijŽ

```
>>> text = 'yeah, but no, but yeah, but no, but yeah'
>>> # Exact match
>>> text == 'yeah'
False
>>> # Match at start or end
>>> text.startswith('yeah')
True
>>> text.endswith('no')
```

```
False
>>> # Search for the location of the first occurrence
>>> text.find('no')
10
>>>
```

åŕźäžŐåď■æİĆçŽĎåŇźéĚ■éIJĂèęĄä¡£çŤĺæ■čåĹŹèąĺè¿¿åijŔåŠŇ re æĺąåİŮãĂĆ äÿžäžĘèğčéĞŁæ■čåĹŹèąĺè¿¿åijŔçŽĎå§žæIJňåŐ§çŘĘïijŇåĄĞèő¿ä¡ăæČşåŇźéĚ■æŢřå■Ůæăijåij ŔçŽĎæŮěæ 11/27/2012 ïijŇä¡ääŔŕäžěè£ŹæăůåĄŽïijŽ

```
>>> text1 = '11/27/2012'
>>> text2 = 'Nov 27, 2012'
>>>
>>> import re
>>> # Simple matching: \d+ means match one or more digits
>>> if re.match(r'\d+/\d+/\d+', text1):
... print('yes')
... else:
... print('no')
...
yes
>>> if re.match(r'\d+/\d+/\d+', text2):
... print('yes')
... else:
... print('no')
...
no
>>>
```

åęĆæđIJä¡ăæČşä¡£çŤĺåŘŇäÿĂäÿłæĺąåijŔåŐżåĄŽåď ŽæňąåŇźéĚ■ïijŇä¡ääžŤèŕěåĚĹåřĘæĺąåijŔå■Ůçňęä

```
>>> datepat = re.compile(r'\d+/\d+/\d+')
>>> if datepat.match(text1):
... print('yes')
... else:
... print('no')
...
yes
>>> if datepat.match(text2):
... print('yes')
... else:
... print('no')
...
no
>>>
```

match() æĂżæŸŕäžŐå■ŮçňęäÿšåijĂåğŇåŐżåŇźéĚ■ïijŇåęĆæđIJä¡ăæČşæ§ěæL'¿å■ŮçňęäÿšäżżæĎŔé ä¡£çŤĺ findall() æŰźæşŢåŐżäžčæŽ£ãĂĆæŕŤåęĆïijŽ

```
>>> text = 'Today is 11/27/2012. PyCon starts 3/13/2013.'
>>> datepat.findall(text)
['11/27/2012', '3/13/2013']
>>>
```

åIJÍåőŽäźL’æ■čåĹŹåijŔçŽĎæŮűåĂŹïijŇéĂŽåÿÿäijŽåĹľçŤÍæŃňåŔůåŐżæ■ŢèŐůåĹĘçżĎãĂĆæŕŤåęĆïij

```
>>> datepat = re.compile(r'(\d+)/(\d+)/(\d+)')
>>>
```

æ■ŢèŐůåĹĘçżĎåŔŕäżěä¡£å¿ŮåŘŐéİćçŽĎåd’ĎçŘĘæŻt’åŁăçőĂå■ŢïijŇåŻääÿžåŔŕäżěåĹĘåĹńářĘæŕŔäÿ

```
>>> m = datepat.match('11/27/2012')
>>> m
<_sre.SRE_Match object at 0x1005d2750>
>>> # Extract the contents of each group
>>> m.group(0)
'11/27/2012'
>>> m.group(1)
'11'
>>> m.group(2)
'27'
>>> m.group(3)
'2012'
>>> m.groups()
('11', '27', '2012')
>>> month, day, year = m.groups()
>>>
>>> # Find all matches (notice splitting into tuples)
>>> text
'Today is 11/27/2012. PyCon starts 3/13/2013.'
>>> datepat.findall(text)
[('11', '27', '2012'), ('3', '13', '2013')]
>>> for month, day, year in datepat.findall(text):
...     print('{}-{}-{}'.format(year, month, day))
...
2012-11-27
2013-3-13
>>>
```

`findall()` æŰźæşŢäijŽæŘIJçt’ćæŰĞæIJňåźűäžěåĹÙèąĺå¡ćåijŔè£ŤåŻđæL’ĂæIJL’çŽĎåŇźéĚ■ãĂĆ åęĆæđIJä¡ăæČşäžěè£■äžčæŰźåijŔè£ŤåŻđåŇźéĚ■ïijŇåŔŕäżěä¡£çŤÍ `finditer()` æŰźæşŢæİěäžčæŻ£ïijŇæŕŤåęĆïijŽ

```
>>> for m in datepat.finditer(text):
...     print(m.groups())
...
('11', '27', '2012')
('3', '13', '2013')
>>>
```

èőÍèőž

åĚşäžŐæ■čåĹŹèąĺè¿¿åijŔçŘĘèőžçŽĎæŢŹçÍŃåůšçżŔèűĚåĞžäžĘæIJňäźęçŽĎèŇČåŻťãĂĆ äÿ■è£ĞïijŇè£ŹäÿĂèŁĆéŸŘè£řäžĘä¡£çŤÍreæĺąåİŮè£ŻèąŇåŇźéĚ■åŠŇæŘIJçťćæŰĞæIJňçŽĎæIJĂå§žæIJňæ æäÿå£Čæ■ěéłd'åřśæŸŕåĚĹä¡£çŤÍ `re.compile()` çijŰèŕŚæ■čåĹŹèąĺè¿¿åijŔå■ŮçňęäÿšïijŇ çĎűåŘŐä¡£çŤÍ `match()`, `findall()` æĹŰèĂĚ `finditer()` ç■L'æŰźæşŢãĂĆ

å¡ŞåĘŹæ■čåĹŹåijŔå■Ůçňęäÿšç ŽĎæŮűåĂŹïijŇçŻÿåŕźæŹőéĄ■çŽĎåĄŽæşŢæŸŕä¡£çŤÍåŐ§ågŇå■Ůçňęä `r'(\d+)/(\d+)/(\d+)'` ãĂĆ è£Źçğ■å■Ůçňęäÿšår Ęäÿ■åŐżèğčæđŘåŔ■æŰIJæİäïijŇè£ŹåIJÍæ■čåĹŹèąĺè åęĆæđIJäÿ■è£ŹæăůåĄžçŽĎèŕİïijŇä¡ăå£Ěéążä¡£çŤÍäÿd'äÿłåŔ■æŰIJæİäïijŇçšżäijij `'(\\d+)/(\\d+)/(\\d+)'` ãĂĆ

éIJĂèęĄæşÍæĎŘçŽĎæŸŕ `match()` æŰźæşŢäžĚäžĚæčĂæ§ěå■Ůçňęäÿšç ŽĎåijĂågŇéČÍåĹĘãĂĆåőČçŽ

```
>>> m = datepat.match('11/27/2012abcdef')
>>> m
<_sre.SRE_Match object at 0x1005d27e8>
>>> m.group()
'11/27/2012'
>>>
```

åęĆæđIJä¡ăæČşçš¿çąőåŇźéĚ■ïijŇçąőä£İä¡ăçŽĎæ■čåĹŹèąĺè¿¿åijŔäžě$çżŞář¿ïijŇářšåČŔè£ŹäźĹè£Źæă

```
>>> datepat = re.compile(r'(\d+)/(\d+)/(\d+)$')
>>> datepat.match('11/27/2012abcdef')
>>> datepat.match('11/27/2012')
<_sre.SRE_Match object at 0x1005d2750>
>>>
```

æIJĂåŘŐïijŇåęĆæđIJä¡ääžĚäžĚæŸŕåĄŽäÿĂæňąçőĂå■ŢçŽĎæŰĞæIJňåŇźéĚ■/æŘIJçťćæŞ■ä¡IJçŽĎèŕİ `re` æĺąåİŮçžğåĹńçŽĎåĞ¡æŢřãĂĆærŤåęĆïijŽ

```
>>> re.findall(r'(\d+)/(\d+)/(\d+)', text)
[('11', '27', '2012'), ('3', '13', '2013')]
>>>
```

ä¡ĘæŸŕéIJĂèęĄæşÍæĎŘçŽĎæŸŕïijŇåęĆæđIJä¡äæL'ŞçőŮåĄŽåď'ğéĞŘçŽĎåŇźéĚ■åŠŇæŘIJçťćæŞ■ä¡IJ æĺąåİŮçžğåĹńçŽĎåĞ¡æŢřäijŽårĘæIJĂè£ŚçijŰèŕŚè£ĞçŽĎæĺąåijŘçijŞå■ŸètůæİěïijŇåŻăæ■d'åźűäÿ■äijŽæűĹ ä¡ĘæŸŕåęĆæđIJä¡£çŤÍécĎçijŰèŕŚæĺąåijŘçŽĎèŕİïijŇä¡ăårĘäijŽåĞŘårŚæ§ěæL'¿åŠŇäÿĂäžŽécİåď'ŰçŽĎåď'Ď

2.5 å■ŮçňęäÿšæŘIJçť ćåŠŇæŻ£æ■ć

éŮőécŸ

ä¡ăæČşåIJÍå■Ůçňęäÿšäÿ■æŘIJçť ćåŠŇåŇźéĚ■æŇĞåőŽçŽĎæŰĞæIJňæĺąåijŘ

èğčåĘşæŰźæąĹ

åŕźäžŐçőĂå■ŢçŽĎå■ŮéİćæĺąåijŔïijŇçŻťæŐëä¡£çŤĺ `str.repalce()` æŰźæşŢå■şåŔŕïijŇæŕŤåęĆïijŽ

```
>>> text = 'yeah, but no, but yeah, but no, but yeah'
>>> text.replace('yeah', 'yep')
'yep, but no, but yep, but no, but yep'
>>>
```

åŕźäžŐåď■æİĆçŽĎæĺąåijŔïijŇèŕůä¡£çŤĺ `re` æĺąåİŮäÿ■çŽĎ `sub()` åĞ¡æŢřãĂĆ äÿžäžĘèŕťæŸŐè£ŹäÿłïijŇåĄĞèő¿ä¡ăæČşåřĘå¡ćåijŔäÿž `11/27/2012` çŽĎæŮěæIJ§å■ŮçňęäÿšæŤźæĹŘ `2012-11-27` ãĂĆçđ'žä¿ŃåęĆäÿŇïijŽ

```
>>> text = 'Today is 11/27/2012. PyCon starts 3/13/2013.'
>>> import re
>>> re.sub(r'(\d+)/(\d+)/(\d+)', r'\3-\1-\2', text)
'Today is 2012-11-27. PyCon starts 2013-3-13.'
>>>
```

`sub()` åĞ¡æŢřäÿ■çŽĎçňňäÿĂäÿłåŔĆæŢřæŸŕèćńåŇźéĚ■çŽĎæĺąåijŔïijŇçňňäžŇäÿłåŔĆæŢřæŸŕæŻ£æ
`\3` æŇĞåŘŠåĽ■éİćæĺąåijŔçŽĎæ■ŢèŐůçżĎåŔůãĂĆ

åęĆæđIJä¡ăæĽŞçőŮçŤĺçŻÿåŘŇçŽĎæĺąåijŔåĄŽåď'Žæňąæ£æ■ćïijŇèĂČèŹŠåĚĹçijŰèŕŠåőČæİěæŔŘå

```
>>> import re
>>> datepat = re.compile(r'(\d+)/(\d+)/(\d+)')
>>> datepat.sub(r'\3-\1-\2', text)
'Today is 2012-11-27. PyCon starts 2013-3-13.'
>>>
```

åŕźäžŐæŻť'åŁååď■æİĆçŽĎæŻ£æ■ćïijŇåŔŕäžěäijăéĂŠäÿĂäÿłæŻ£æ■ćåŻđèřČåĞ¡æŢřæİěäžčæŻ£ïijŇæŕ

```
>>> from calendar import month_abbr
>>> def change_date(m):
... mon_name = month_abbr[int(m.group(1))]
... return '{} {} {}'.format(m.group(2), mon_name, m.group(3))
...
>>> datepat.sub(change_date, text)
'Today is 27 Nov 2012. PyCon starts 13 Mar 2013.'
>>>
```

äÿĂäÿłæŻ£æ■ćåŻđèřČåĞ¡æŢřçŽĎåŔĆæŢřæŸŕäÿĂäÿł `match` åŕźèśąïijŇäź§åřśæŸŕ `match()` æĹŰèĂĚ `find()` è£ŤåŻđçŽĎåŕźèśąãĂĆ ä¡£çŤĺ `group()` æŰźæşŢæİěæŔŘåŘŰçĽźåőŽçŽĎåŇźéĚ■éČĺåĹĘãĂĆåŻđèřČåĞ¡æŢřæIJĂåŘŐè£ŤåŻđæŻ£æ■ćå■Ůçňęäÿšã

åęĆæđIJéŹď'äžĘæŻ£æ■ćåŘŐçŽĎçżŞæđIJåď'ŰïijŇä¡ăè£ŸæČşç§ěéĄŞæIJĽåď'ŽåřŚæŻ£æ■ćåŔŚçŢ§äžĘ
`re.subn()` æİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
>>> newtext, n = datepat.subn(r'\3-\1-\2', text)
>>> newtext
```

```
'Today is 2012-11-27. PyCon starts 2013-3-13.'
>>> n
2
>>>
```

èőÍèőž

åĚşäžŐæ■čåĹŹèąÍè¿¿åijŔæŘIJçť ćåŠŇæŻ£æ■ćïijŇäÿŁéİćæijŤçď žçŽĎ sub() æŰźæşŢå§žæIJňåůšçżŔæűţçŻŰäžĘæĽ ĂæIJL ãĂĆ åĚűåőđæIJĂéŽ¿çŽĎéČĺåĹĘářśæŸŕçijŰåĘŹæ■čåĹŹèąÍè¿¿

2.6 å■Ůçňęäÿšå£¡çŢěåď ğářŔåĘŹçŽĎæŘIJçť ćæŻ£æ■ć

éŮőécŸ

ä¡ăéIJĂèęĄäžěå£¡çŢěåď ğářŔåĘŹçŽĎæŰźåijŔæŘIJçť ćäÿŐæŻ£æ■ćæŰĞæIJňå■Ůçňęäÿš

èğčåĘşæŰźæąĹ

äÿžäžĘåIJÍæŰĞæIJňæŞ■ä¡IJæŮűå£¡çŢěåď ğářŔåĘŹïijŇä¡ăéIJĂèęĄåIJÍä¡£çŤÍ re æĺąåİŮçŽĎæŮűåĂŹçżŹè£ŹäžŻæŞ■ä¡IJæŔŘä¿Ż re.IGNORECASE æăĞå£ŮåŔĆæŢřăĂĆæŕŤåęĆïijŽ

```
>>> text = 'UPPER PYTHON, lower python, Mixed Python'
>>> re.findall('python', text, flags=re.IGNORECASE)
['PYTHON', 'python', 'Python']
>>> re.sub('python', 'snake', text, flags=re.IGNORECASE)
'UPPER snake, lower snake, Mixed snake'
>>>
```

æIJĂåŘŐçŽĎéĆčäÿłä¿Ńå■ŘæŔ■çď žäžĘäÿĂäÿłářŔçijžéŹůïijŇæŻ£æ■ćå■Ůçňęäÿšåźűäÿ■äijŽèĞłåŁĺèů§äÿžäžĘä£őåď ■è£ŹäÿłïijŇä¡ăåŔŕèČ¡éIJĂèęĄäÿĂäÿłè¿ĚåŁľåĞ¡æŢřïijŇářšåČŔäÿŃéİćçŽĎè£ŹæăůïijŽ

```
def matchcase(word):
    def replace(m):
        text = m.group()
        if text.isupper():
            return word.upper()
        elif text.islower():
            return word.lower()
        elif text[0].isupper():
            return word.capitalize()
        else:
            return word
    return replace
```

äÿŃéİćæŸŕä¡£çŤĺäÿŁè£ŕåĞ¡æŢřçŽĎæŰźæşŢïijŽ

```
>>> re.sub('python', matchcase('snake'), text, flags=re.IGNORECASE)
'UPPER SNAKE, lower snake, Mixed Snake'
>>>
```

èґŚèĂĚæşĺïijŽ `matchcase('snake')` è£ŤåŻđäžĘäÿĂäÿłåŻđèřČåĞ¡æŢř(åŔĆæŢřå£ĚéążæŸŕ `match` åŕźèśą)ïijŇåL’■éİćäÿĂèŁĆæŔŘåĹřè£ĞïijŇ `sub()` åĞ¡æŢřéŹď’äžĘæŐěåŔŮæŻ£æ■ćå■Ůçňęäÿšåď’ŰïijŇè£ŸèČ¡æŐěåŔŮäÿĂäÿłåŻđèřČåĞ¡æŢřãĂĆ

èőĺèőž

åŕźäžŐäÿĂèĹňçŽĎå£¡çŢěåď’ğåřŔåĘŹçŽĎåŇźéĚ■æŞ■ä¡IJïijŇçőĂå■ŢçŽĎäijåéĂŠäÿĂäÿł `re.IGNORECASE` æăĞå£ŮåŔĆæŢřåřśåůšçżŔèűşåď’§äžĘãĂĆ ä¡ĘæŸŕéIJĂèęĄæşĺæĎŔçŽĎæŸŕïijŇè£ŹäÿłåŕźäžŐæ§ŘäžŻéIJĂèęĄåď’ğåřŔåĘŹè¡ñæ■ćçŽĎUnicodeåŇźéĚ■å åŔĆèĂČ2.10åřŔèŁĆäžĘèğčæŻt’åď’ŽçżĘèŁĆãĂĆ

2.7 æIJĂç§■åŇźéĚ■æĺąåijŔ

éŮőécŸ

ä¡ăæ■čåIJĺèŕŢçİĂçŤĺæ■čåĹŹèąĺè¿¿åijŔåŇźéĚ■æ§ŘäÿłæŰĞæIJñæĺąåijŔïijŇä¡ĘæŸŕåőČæL’¿åĹřçŽĎæŸ èĂŇä¡ăæČşä£őæŤźåőČåŔŸæĹŘæ§ěæL’¿æIJĂç§■çŽĎåŔŕèČ¡åŇźéĚ■ãĂĆ

èğčåĘşæŰźæąĹ

è£ŹäÿłéŮőécŸäÿĂèĹňåĞžçŐřåIJĺéIJĂèęĄåŇźéĚ■äÿĂåŕźåĹĘéŽŤçňęäźŃéŮt’çŽĎæŰĞæIJñçŽĎæŮűåĂ äÿžäžĘèŕt’æŸŐæÿĚæěŽïijŇèĂČèŹŚåęĆäÿŃçŽĎä¿Ňå■ŘïijŽ

```
>>> str_pat = re.compile(r'\"(.*)\"')
>>> text1 = 'Computer says "no."'
>>> str_pat.findall(text1)
['no.']
>>> text2 = 'Computer says "no." Phone says "yes."'
>>> str_pat.findall(text2)
['no." Phone says "yes.']
>>>
```

åIJĺè£ŹäÿłäĄ¿Ňå■ŘäÿïijŇæĺąåijŔ `r'\"(.*)\"'` çŽĎæĎŔåŻ¿æŸŕåŇźéĚ■èćńåŔŇåijŢåŔůåŇĚåŘńçŽĺ ä¡ĘæŸŕåIJĺæ■čåĹŹèąĺè¿¿åijŔäÿ■*æŞ■ä¡IJçňęæŸŕèt’łåľłçŽĎïijŇåŻăæ■d’åŇźéĚ■æŞ■ä¡IJäijŽæ§ěæL’¿æIJĂéŢ äžŐæŸŕåIJĺçňňäžŇäÿłäĄ¿Ňå■ŘäÿïæŘIJçt’ć `text2` çŽĎæŮűåĂŹè£ŤåŻđçżŞæđIJåźűäÿ■æŸŕæĹŚäžňæČşèęĄç

äÿžäžĘä£őæ■čè£ŹäÿłéŮőécŸïijŇåŔŕäžěåIJĺæĺąåijŔäÿ■çŽĎ*æŞ■ä¡IJçňęåŘŐéİćåŁääÿŁ?ä£őéěŕçňęïijŇå

```
>>> str_pat = re.compile(r'\"(.*?)\"')
>>> str_pat.findall(text2)
['no.', 'yes.']
>>>
```

è£Źæăůářšä¡£å¿ŮåŇźéĚ■åŔŸæĹŘéİđèť'łăl'łæĺąåijŔïijŇäžŐèĂŇå¿ŮåĹřæIJĂç§■çŽĎåŇźéĚ■ïijŇäź§ářšä

èőĺèőž

è£ŹäÿĂèŁĆåśŢçd'žäžĘåIJĺåĘŹåŇĚåŘńçĆź(.)å■Ůçňęç ŽĎæ■čåĹŹèąĺè¿¿åijŔçŽĎæŮűåĂŹéĄĞåĹřçŽĎäÿ
åIJĺäÿĂäÿłæĺąåijŔå■Ůçňęäÿšäÿ■ïijŇçĆź(.)åŇźéĚ■éŹd'äžĘæ■ćèąŇåd'ŰçŽĎäżżä¡Ţå■Ůçňęã ĂĆ
çĎűèĂŇïijŇåęĆæđIJä¡ăářĘçĆź(.)åŔůæŤ¿åIJĺåijĂåğŇäÿŐçżŞæİ§çňę(ærŤåęĆåijŢåŔů)äźŃéÚt'çŽĎæŮűåĂŹi
è£ŹæăůéĂŽåÿÿäijŽářijèĞt'å¿Ĺåd'Žäÿ■éÚt'çŽĎèćńåijĂåğŇäÿŐçżŞæİ§çňęåŇĚåŘńçŽĎæŰĞæIJňèćńå£¡çŢěæ
éĂŽè£ĞåIJÍ * æĹŰèĂĚ + è£ŹæăůçŽĎæŞ■ä¡IJçňęåŘŐéİćæůżåŁăäÿĂäÿł ?
åŔŕäžěåijżåĹűåŇźéĚ■çőŮæşŢæŤźæĹŘářżæL'¿æIJĂç§■çŽĎåŔŕèČ¡åŇźéĚ■ãĂĆ

2.8 åd'ŽèąŇåŇźéĚ■æĺąåijŔ

éŮőécŸ

ä¡ăæ■čåIJĺèŕŢçİĂä¡£çŤĺæ■čåĹŹèąĺè¿¿åijŔåŐżåŇźéĚ■äÿĂåd'ğåİŮçŽĎæŰĞæIJňïijŇèĂŇä¡ăéIJĂèęĄèů

èğčåĘşæŰźæąĹ

è£ŹäÿłéŮőécŸå¿ĹåĚÿåđŃçŽĎåĞžçŐřåIJĺå¡Şä¡ăçŤĺçĆź(.)åŐżåŇźéĚ■äżżæĎŔå■Ůçňęç ŽĎæŮűåĂŹïijŇæ
ærŤåęĆïijŇåĄĞèő¿ä¡ăæČşèŕŢçİĂåŐżåŇźéĚ■Cèŕ■èĺĂåĹĘåL'šçŽĎæşĺéĞŁïijŽ

```
>>> comment = re.compile(r'/\*(.*?)\*/')
>>> text1 = '/* this is a comment */'
>>> text2 = '''/* this is a
... multiline comment */
... '''
>>>
>>> comment.findall(text1)
[' this is a comment ']
>>> comment.findall(text2)
[]
>>>
```

äÿžäžĘä£őæ■čè£ŹäÿłéŮőécŸïijŇä¡ăåŔŕäžěä£őæŤźæĺąåijŔå■ŮçňęäÿšïijŇåćđåŁåářźæ■ćèąŇçŽĎæŤŕæŇ

```
>>> comment = re.compile(r'/\*((?:.|\n)*?)\*/')
>>> comment.findall(text2)
[' this is a\n multiline comment ']
>>>
```

åIJĺè£ŹäÿłæĺąåijŔäÿ■ïijŇ (?:.|\n) æŇĞåőŽäžĘäÿĂäÿłéİđæ■ŢèŐůçżĎ
(äź§ářšæŸŕåőČåőŽäźĽäžĘäÿĂäÿłäžĚäżĚçŤĺæĺěåĄŽåŇźéĚ■ïijŇèĂŇäÿ■èČ¡éĂŽè£Ğå■ŢçŃňæ■ŢèŐůæĹŰèĂ

èőÍèőž

re.compile() åĞ¡æŢřæŐěåŔŮäÿĂäÿłæăĞå£ŮåŔĆæŢřåŔń re.DOTALL ïijŇåIJÍè£ŹéĞŇéÍđåÿÿæIJL'çŤÍãĂĆ åőČåŔŕäžěèől'æ■čåĹŹèąÍè¿¿åijŔäÿ■çŽĎçĆź(.)åŇźéĚ■åŇĚæŃňæ■ćèąŇ

```
>>> comment = re.compile(r'/\*(.*?)\*/', re.DOTALL)
>>> comment.findall(text2)
[' this is a\n multiline comment ']
```

åŕźäžŐçőĂå■ŢçŽĎæČĚåĘţä¡£çŤÍ re.DOTALL æăĞèőřåŔĆæŢřåůěä¡IJçŽĎå¿Ĺåě¡ïijŇ ä¡ĘæŸŕåęĆæđIJæÍąåijŔéÍđåÿÿåď■æÍĆæĹŰèĂĚæŸŕäÿžäžĘæđĎéĂăå■Ůçňęäÿšäžď'çL'ŇèĂŇåřĘåď'Žäÿłæĺąåi è£ŹæŮűåĂŹä¡£çŤÍè£ŹäÿłæăĞèőřåŔĆæŢřåřšåŔŕèČ¡åĞžçŐřäÿĂäžŽéŮőécŸãĂĆ åęĆæđIJèől'ä¡ăéĂL'æŃl'çŽĎèŕÍïijŇæIJĂåě¡è£ŸæŸŕåőŽäźL'èĞłåůšçŽĎæ■čåĹŹèąÍè¿¿åijŔæÍąåijŔïijŇè£Źæăů

2.9 åřĘUnicodeæŰĞælJňæăĞåĞĘåŇŰ

éŮőécŸ

ä¡ăæ■čåIJÍåď'ĎçŘĘUnicodeå■ŮçňęäÿšïijŇéIJĂèęĄçąőä£ÍæL'ĂæIJL'å■ŮçňęäÿšåIJÍåžŢåśĆæIJL'çŻÿåŘŇ

èğčåĘşæŰźæąĹ

åIJÍUnicodeäÿ■ïijŇæ§ŘäžŻå■Ůçňęèč¡åď'§çŤÍåď'Žäÿłå ŘĹæşŢçŽĎçijŰçăĄèąÍçď'žãĂĆäÿžäžĘèŕť'æŸŐïij

```
>>> s1 = 'Spicy Jalape\u00f1o'
>>> s2 = 'Spicy Jalapen\u0303o'
>>> s1
'Spicy JalapeÃ±o'
>>> s2
'Spicy JalapeÃ±o'
>>> s1 == s2
False
>>> len(s1)
14
>>> len(s2)
15
>>>
```

è£ŹéĞŇçŽĎæŰĞæIJň"Spicy JalapeÃ±o"ä¡£çŤÍäžĘäÿď'çğ■å¡ćåijŔæÍěèąÍçď'žãĂĆ çňňäÿĂçğ■ä¡£çŤÍæŢt'ä¡Şå■Ůçňę"Ã±"(U+00F1)ïijŇçňňäžŇçğ■ä¡£çŤÍæŃL'äÿĄå■Ůæŕ■"n"åŘŐéÍćèů§äÿĂäÿł

åIJÍéIJĂèęĄæŕŤè¿Ča■ŮçňęäÿšçŽĎçÍŃåžŔäÿ■ä¡£çŤÍå■ŮçňęçŽĎåď'Žçğ■èąÍçď'žäijŽäžğçŤ§éŮőécŸãĂĆ äÿžäžĘä£őæ■čè£ŹäÿłéŮőécŸïijŇä¡ăåŔŕäžěä¡£çŤÍunicodedataæÍąåÍŮåĚĹåřĘæŰĞæIJňæăĞåĞĘåŇŰïijŽ

```
>>> import unicodedata
>>> t1 = unicodedata.normalize('NFC', s1)
>>> t2 = unicodedata.normalize('NFC', s2)
>>> t1 == t2
```

```
True
>>> print(ascii(t1))
'Spicy Jalape\xf1o'
>>> t3 = unicodedata.normalize('NFD', s1)
>>> t4 = unicodedata.normalize('NFD', s2)
>>> t3 == t4
True
>>> print(ascii(t3))
'Spicy Jalapen\u0303o'
>>>
```

normalize() çňňäÿĂäÿłåŔĆæŢřæŇĞåőŽå■ŮçňęäÿšæăĞåĞĘåŇŰçŽĎæŰźåijŔãĂĆ NFCèąĺçď'žå■ŮçňęåžŤèŕěæŸŕæŢt'ä¡ŞçżĎæĹŘ(æŕŤåęĆåŔŕèČ¡çŽĎèŕÍåřśä¡£çŤĺå■ŢäÿĂçijŰçăA)ïijŇèĂŇNFI

PythonåŘŇæăůæŤŕæŇĄæĽľ'åśŢçŽĎæăĞåĞĘåŇŰå¡ćåijŔNFKCåŠŇNFKDïijŇåőČäžňåIJÍåď'ĎçŘĘæ§Ř

```
>>> s = '\ufb01' # A single character
>>> s
'ïňĄ'
>>> unicodedata.normalize('NFD', s)
'ïňĄ'
# Notice how the combined letters are broken apart here
>>> unicodedata.normalize('NFKD', s)
'fi'
>>> unicodedata.normalize('NFKC', s)
'fi'
>>>
```

èőÍèőž

æăĞåĞĘåŇŰåŕźäžŐäżä¡ŢéIJĂèęĄäżěäÿĂèĞt'çŽĎæŰźåijŔåď'ĎçŘĘUnicodeæŰĞæIJňçŽĎçÍŇåžŔéČ¡å å¡Şåď'ĎçŘĘæÍèèĞłçŤÍæĹůè¿ŞåĚěçŽĎå■Ůçňęäÿšè ĂŇä¡ăå¿ĹéŽ¿åÖżæŐğåĹűçijŰçăĄçŽĎæŮűåĂŹårď'åĚűåę

åIJÍæÿĚçŘĘåŠŇè£Ğæżď'æŰĞæIJňçŽĎæŮűåĂŹå■ŮçňęçŽĎæăĞåĞĘåŇŰäź§æŸŕå¿ĹéĞ■èęĄçŽĎãĂĆ æŕŤåęĆïijŇåĄĞèő¿ä¡ăæČşæÿĚéŹď'æŐL'äÿĂäžŻæŰĞæIJňäÿŁéİćçŽĎåŔŸé§şçňęçŽĎæŮűåĂŹ(åŔŕèČ¡æŸŕäÿ

```
>>> t1 = unicodedata.normalize('NFD', s1)
>>> ''.join(c for c in t1 if not unicodedata.combining(c))
'Spicy Jalapeno'
>>>
```

æIJĂåŘŐäÿĂäÿłä¿Ńå■ŘåśŢçď'žäžĘ unicodedata æĺąåİŮçŽĎåŘęäÿĂäÿłéĞ■èęĄæŰźéİćïijŇäź§ářśæ combining() åĞ¡æŢřåŔŕäžěæţŃèŕŢäÿĂäÿłå■ŮçňęæŸŕåŘęäÿžåŠŇé§şå■Ůçňęã ĂĆ åIJÍè£ŹäÿłæĺąåİŮäÿ■è£ŸæIJL'åĚűäžŰåĞ¡æŢřçŤÍäžŐæ§ěæL'¿å■Ůçňęçśžå Ĺńïij ŇæţŃèŕŢæŸŕåŘęäÿžæŢřå■Ůå

UnicodeæŸ¿çĎűæŸŕäÿĂäÿłå¿Ĺåď'ğçŽĎäÿżéćŸãĂĆåęĆæđIJæČşæŻť'æůśåĚěçŽĎäžĘèğčåĚşäžŐæăĞåČ èŕůçIJŃèĂČ UnicodeåőŸç¡Śäÿ■åĚşäžŐè£ŹéČĺåĹęçŽĎèŕť'æŸŐ Ned BatchelderåIJÍ äžŰçŽĎç¡ŚçńŹ äÿŁåŕźPythonçŽĎUnicodeåď'ĎçŘĘéŮőécŸäź§æIJL'äÿĂäÿłå¿Ĺåě¡çŽĎäżŃçż■ãĂĆ

2.10 ålJÍæ■čåĹŹåijŔäÿ■ä¡£çŤÍUnicode

éŮőécŸ

ä¡ăæ■čåIJÍä¡£çŤÍæ■čåĹŹèąĺè¿¿åijŔåd'ĎçŘĘæŰĞæIJñïijÑä¡ĘæŸŕåĚşæşÍçŽĎæŸŕUnicodeå■Ůçňęåd'Ďç

èğčåĘşæŰźæąĹ

éžŸèőd'æČĚåĘţäÿŃ re æÍąåİŮåůšçżŔåŕźäÿĂäžŻUnicodeå■Ůçňęçśżæ IJL'äžĘå§žæIJñçŽĎæŤŕæŇĄãĂĆ æŕŤåęĆïijŇ \\d åůšçżŔåŇźéĚ■äżżæĎŘçŽĎunicodeæŢřå■Ůå■Ůçňęäžĕïijž

```
>>> import re
>>> num = re.compile('\d+')
>>> # ASCII digits
>>> num.match('123')
<_sre.SRE_Match object at 0x1007d9ed0>
>>> # Arabic digits
>>> num.match('\u0661\u0662\u0663')
<_sre.SRE_Match object at 0x101234030>
>>>
```

åęĆæđIJä¡ăæČşåIJÍæĺąåijŔäÿ■åŇĚåŘńæŇĞåőŽçŽĎUnicodeå■ŮçňęïijŇä¡ääŔŕäżěä¡£çŤÍUnicodeå■Ůçň \uFFF æĹŰèĂĚ \UFFFFFFFF)ãĂĆ æŕŤåęĆïijŇäÿŃéİćæŸŕäÿĂäÿłåŇźéĚ■åĞääÿłäÿ■åŘŇéŸ£æŃL'äijŕçijŰç

```
>>> arabic = re.compile('[\u0600-\u06ff\u0750-\u077f\u08a0-\u08ff]+
↪')
>>>
```

å¡ŞæL'ğèąŇåŇźéĚ■åŠŇæŘIJçt'ćæŞ■ä¡IJçŽĎæŰűåĂŹïijŇæIJĂåě¡æŸŕåĚĹæăĞåĞĘåŇŰåźűäÿŤæÿĚçŘĚ ä¡ĘæŸŕåŘŇæăůäź§åžŤèŕěæşÍæĎŔäÿĂäžŻçL'źæőŁæČĚåĘţïijŇærŤåęĆåIJÍå£¡çŢěåd'ğářŔåĘŹåŇźéĚ■åŠŇåd'

```
>>> pat = re.compile('stra\u00dfe', re.IGNORECASE)
>>> s = 'straÃ§e'
>>> pat.match(s) # Matches
<_sre.SRE_Match object at 0x10069d370>
>>> pat.match(s.upper()) # Doesn't match
>>> s.upper() # Case folds
'STRASSE'
>>>
```

èőÍèőž

æůůåŘĹä¡£çŤÍUnicodeåŠŇæ■čåĹŹèąĺè¿¿åijŔéĂŽåÿÿäijŽèőľ'ä¡ăæŁŞçŃĆãĂĆ åęĆæđIJä¡ăçIJ§çŽĎæL'ŞçőŮè£ŹæăůåĄŽçŽĎèŕĺïijŇæIJĂåě¡èĂČèŹŚäÿŇåőL'èčĚçňňäÿL'æŰźæ■čåĹŹåijŔåžŞ åőČäžňäijŽäÿžUnicodeçŽĎåd'ğářŔåĘŹè¡ňæ■ćåŠŇåĚűäżŰåd'ğéĞŔæIJL'èűčçL'źæĂğæŔŘä¿ŻåĚĺéİćçŽĎæŤŕ

2.11 åĹåéŹďå■Ůçňęäÿšäÿ■äÿ■élJĂèęĄçŽĎå■Ůçňę

éŮőécŸ

ä¡ăæČşåŐżæŐĽ'æŰĞæIJňå■Ůçňęäÿšåij Ăåď't'ïijŇçżŞář¿æĹŰèĂĚäÿ■éŮt'äÿ■æČşèęĄçŽĎå■Ůçňęïij Ňæŕ

èğčåĘşæŰźæąĹ

strip() æŰźæşŢèČ¡çŤĺäžŐåĹåéŹď'åijĂåğŃæĹŰçżŞář¿çŽĎå■Ůçňęã ĂĆ lstrip() åŠŇ rstrip() åĹĘąĹńäżŐåůęåŠŇäżŐåŔşæL'ğèąŇåĹåéŹď'æŞ■ä¡IJãĂĆ éżŸèŏď'æČĚåĘţäÿŃïijŇè£ŹäžŻæŰźæşŢäijŽåŐżéŹď'çl'žçŹ¡å■Ůçňęïij Ňä¡ĘæŸŕä¡ääź§åŔŕäżěæŇĞåőŽåĚűäżŰ

```
>>> # Whitespace stripping
>>> s = ' hello world \n'
>>> s.strip()
'hello world'
>>> s.lstrip()
'hello world \n'
>>> s.rstrip()
' hello world'
>>>
>>> # Character stripping
>>> t = '-----hello====='
>>> t.lstrip('-')
'hello====='
>>> t.strip('-=')
'hello'
>>>
```

èőÍèőž

è£ŹäžŻ strip() æŰźæşŢåIJÍèŕżåŔŰåŠŇæÿĚçŘĘæŢřæ■őäżěåď'ĞåŘŐçż■åď'ĎçŘĘçŽĎæŮűåĂŹæŸŕç æŕŤåęĆïijŇä¡ääŔŕäżěçŤÍåőČäžňæÍěåŐżæŐĽ'çl'žæăijïijŇåijŢåŔůåŠŇåőŇæĹŘåĚűäżŰäżżåŁąãĂĆ

ä¡ĘæŸŕéIJĂèęĄæşÍæĎŔçŽĎæŸŕåŐżéŹď'æŞ■ä¡IJäÿ■äijŽåŕźå■ŮçňęäÿšçŽĎäÿ■éŮt'çŽĎæŰĞæIJňäžğçŤ

```
>>> s = ' hello      world \n'
>>> s = s.strip()
>>> s
'hello      world'
>>>
```

åęĆædIJä¡ăæČşåď'ĎçŘĘäÿ■éŮt'çŽĎçl'žæăijïijŇéĆčäźĹä¡ăéIJĂèęĄæśĆåŁl'åĚűäžŰæŁĂæIJŕãĂĆæŕŤåę replace() æŰźæşŢæĹŰèĂĚæŸŕçŤÍæ■čåĹŹèąÍè¿¿åijŔæŻ£æ■ćãĂĆçď'žä¿ŇåęĆäÿŃïijŽ

```
>>> s.replace(' ', '')
'helloworld'
>>> import re
```

```
>>> re.sub('\s+', ' ', s)
'hello world'
>>>
```

éĂŽåÿÿæČĚåĘţäÿŃä¡ăæČşåřĘå■Ůçňęäÿš strip æŞ■ä¡IJåŠŇåĚűäżŰè£■äžčæŞ■ä¡IJçŻÿçżŞåŘĹïijŇæŕ
åęĆæđIJæŸŕè£ŹæăůçŽĎèŕÏïijŇéĆčäźĹçŤ§æĹŘåŹĺèąĺè¿¿åijŔåřśåŔŕäžěåď ğæŸ¿èžńæĽŃäžĘãĂĆæŕŤåęĆïijŽ

```
with open(filename) as f:
    lines = (line.strip() for line in f)
    for line in lines:
        print(line)
```

åIJĺè£ŹéĞŇïijŇèąĺè¿¿åijŔ lines = (line.strip() for line in f)
æĽğèąŇæŢřæ■őè¡ňæ■ćæŞ■ä¡IJãĂĆ è£Źçğ■æŰźåijŔéÍđåÿÿéńŸæŢĹïijŇåŽääÿžåőČäÿ■éIJÄèęĄécĎåĚĹèŕżå
åőČäžĚäżĚåŔłæŸŕåĹŻåžžäÿĂäÿłçŤ§æĹŘåŹĺïijŇåźűäÿŤæŕŔæňąè£ŤåŽđèąŇäźŃåĽ■äijŽåĚĹæĽğèąŇ
strip æŞ■ä¡IJãĂĆ

åŕźäžŐæŻť énŸéŸűçŽĎstripïijŇä¡ăåŔŕèČ¡éIJÄèęĄä¡£çŤÍ translate()
æŰźæşŢãĂĆèŕůåŔĆéŸĚäÿŃäÿÄèŁĆäžĘèğčæŻť åď ŽåĚşäžŐå■ŮçňęäÿšæÿĚçŘĘçŽĎåĘĚåőźãĂĆ

2.12 åőąæ§ěæÿĚçŘĘæŰĞæIJňå■Ůçňęäÿš

éŮőécŸ

äÿĂäžŻæŮăèĄŁçŽĎåźijçĺŽéżŠåőćåIJĺä¡ăçŽĎç¡ŚçńŹéąţéÍćèąĺå■Ţäÿ■è¿ŞåĚěæŰĞæIJň”pÃ¡tÄěÃűÃś”ïi

èğčåĘşæŰźæąĹ

æŰĞæIJňæÿĚçŘĘéŮőécŸäijŽæűĽ’åŔŁåĹřåŇĚæŃňæŰĞæIJňèğčæđŘäÿŐæŢřæ■őåď ĎçŘĘç■Ľ’äÿĂçşżå
åIJĺéÍđåÿÿçőĂå■ŢçŽĎæČĚå¡ćäÿŃïijŇä¡ăåŔŕèČ¡äijŽéĂĽ’æŃľä¡£çŤĺå■Ůçňęäÿšå Ğ¡æŢř(æŕŤåęĆ
str.upper() åŠŇ str.lower())åřĘæŰĞæIJňè¡ňäÿžæăĞåĞĘæäijåijŔãĂĆ ä¡£çŤÍ
str.replace() æĹŰèĂĚ re.sub() çŽĎçőĂå■ŢæŻ£æ■ćæŞ■ä¡IJèČ¡åĹăéŹđ’æĹŰèĂĚæŤźåŔŸæŇĞåőŽ
ä¡ăåŘŇæăůè£Ÿåŕźäžěä¡£çŤÍ2.9åřŔèŁĆçŽĎ unicodedata.normalize()
åĞ¡æŢřåřĘunicodeæŰĞæIJňæăĞåĞĘåŇŰãĂĆ

çĎűåŘŐïijŇæIJĽ’æŮűåĂŹä¡ăåŔŕèČ¡è£ŸæČşåIJĺæÿĚçŘĘæŞ■ä¡IJäÿŁæŻť è£ŻäÿĂæ■ěãĂĆæŕŤåęĆïijŇä¡
äÿžäžĘè£ŹæăůåĄŽïijŇä¡ăåŔŕäžěä¡£çŤÍçżŔåÿÿäijŽècńå£¡èğĘçŽĎ str.translate()
æŰźæşŢãĂĆ äÿžäžĘæijŤçď žïijŇåĄĞèő¿ä¡äçŐřåIJĺæIJĽäÿŃéÍćè£ŹäÿłåĞŇäźśçŽĎå■Ůçňęäÿšïijž

```
>>> s = 'pÃ¡tÄěÃűÃś\fis\tawesome\r\n'
>>> s
'pÃ¡tÄěÃűÃś\x0cis\tawesome\r\n'
>>>
```

çňňäÿĂæ■ěæŸŕæÿĚçŘĘçľžçŹ¡å■ŮçňęäĉäÿžäžĘè£ŹæăůåĄŽïijŇåĚĹåĹŻåžžäÿĂäÿłåřŔçŽĎè¡ňæ■ćèąĺ
translate() æŰźæşŢïijž

```
>>> remap = {
...     ord('\t') : ' ',
...     ord('\f') : ' ',
...     ord('\r') : None # Deleted
... }
>>> a = s.translate(remap)
>>> a
'pÃ¡tÄëÃűÃś is awesome\n'
>>>
```

æ■čåęĆä¡äçIJŃçŽĎéĆčæăůïijŇçľžçŹ¡å■Ůçňę \t åŠŇ \f åůšçżŔèćńéĞ■æŰřæŸăářĎåĹřäÿĂäÿłçľžæăijãĂĆåŻđè¡ęå■ŮçňęrçŻť'æŐěèćńåĹăéŹď'ãĂĆ

ä¡ăåŔŕäžěäžěè£Źäÿłèąĺæăijäÿžå§žçąÀè£ŻäÿĂæ■éæđĎåžžæŻť'åď'ğçŽĎèąĺæăijãĂĆæŕŤåęĆïijŇèőľ'æĹŚä

```
>>> import unicodedata
>>> import sys
>>> cmb_chrs = dict.fromkeys(c for c in range(sys.maxunicode)
...                          if unicodedata.combining(chr(c)))
...
>>> b = unicodedata.normalize('NFD', a)
>>> b
'pÃ¡tÄëÃűÃś is awesome\n'
>>> b.translate(cmb_chrs)
'python is awesome\n'
>>>
```

äÿŁéİćä¿Ńå■Řäÿ■ïijŇéĂŽè£Ğä¡£çŤĺ dict.fromkeys() æŰźæşŢæđĎéĂăäÿĂäÿłå■ŮåĚÿïijŇæŕŔäÿłUnicodeåŠŇé§şçňęä¡IJäÿžéŤőïijŇåŕźåžŤçŽĎåĂijåĚĺéČĺäÿž None ãĂĆ

çĎűåŘŐäj£çŤĺ unicodedata.normalize() åřĘåŐ§åğŃè¿ŞåĚěæăĞåĞĘåŇŰäÿžåĹĘèğčå¡ćåijŔå■Ů çĎűåŘŐåĘ■èřČçŤĺ translate åĞ¡æŢřåĹăéŹď'æĽ'ĂæIJL'éĞ■é§şçňęãĂĆ åŘŇæăůçŽĎæŁĂæIJŕäź§åŔŕäžěèćńçŤĺæİěåĹăéŹď'åĚűäžŰçśžåđŃçŽĎå■Ůçňę(æŕŤåęĆæŐğåĹűå■Ůçňęç■Ĺ')ã

ä¡IJäÿžåŘęäÿĂäÿłä¿Ńå■ŘïijŇè£ŹéĞŇæđĎéĂăäÿĂäÿłåřĘæĽ'ĂæIJL'UnicodeæŢřå■Ůå■ŮçňęæŸăářĎåĹi

```
>>> digitmap = { c: ord('0') + unicodedata.digit(chr(c))
...           for c in range(sys.maxunicode)
...           if unicodedata.category(chr(c)) == 'Nd' }
...
>>> len(digitmap)
460
>>> # Arabic digits
>>> x = '\u0661\u0662\u0663'
>>> x.translate(digitmap)
'123'
>>>
```

åŘęäÿĂçğ■æÿĚçŘĘæŰĞæIJňçŽĎæŁĂæIJŕæűL'åŔŁåĹřI/Oèğčçă Ąäÿ Őçij Ůçă Ąåğ¡æŢřãĂĆè£ŹéĞŇçŽĎ çĎűåŘŐåĘ■çżŞåŘĹ encode() æĹŰèĂĚ decode() æŞ■ä¡IJæİěæÿĚéŹď'æĹŰä£őæŤźåőČãĂĆæŕŤåęĆïijŽ

```
>>> a
'pÃ¡tÄëÃűÃś is awesome\n'
>>> b = unicodedata.normalize('NFD', a)
>>> b.encode('ascii', 'ignore').decode('ascii')
'python is awesome\n'
>>>
```

è£ŹéĞŇçŽĎæăĞåĞĘåŇŰæŞ■ä¡IJåřĘåŐ§æİěçŽĎæŰĞæIJňåĹĘèğčäÿžå■ŢçŃňçŽĎåŠŇé§şçňęãĂĆæŐěă
å¡ŞçĎűïijŇè£Źçğ■æŰźæşŢäžĔäžĚåŔłåIJĺæIJĂåŘŐçŽĎçŻőæăĞåřśæŸŕèŐůåŘŰåĹřæŰĞæIJňåŕźåžŤACSIIèą

èőĺèőž

æŰĞæIJňå■Ůçňęæÿ̆ĚçŘĘäÿĂäÿłæIJĂäÿżèęĄçŽĎéŮőécŸåžŤèřěæŸŕè£ŘèąŇçŽĎæĂğèČ¡ãĂĆäÿĂèĹňæ
åŕźäžŐçőĂå■ŢçŽĎæŻ£æ■ćæŞ■ä¡IJïijŇ `str.replace()` æŰźæşŢéĂŽåÿÿæŸŕæIJĂå£ńçŽĎïijŇçŤŽèĞşåIJ
æŕŤåęĆïijŇäÿžäžĘæÿĚçŘĘçl'žçŹ¡å■ŮçňęïijŇä¡ăåŘŕäžěè£ŹæăůåĄŽïijŽ

```
def clean_spaces(s):
    s = s.replace('\r', '')
    s = s.replace('\t', ' ')
    s = s.replace('\f', ' ')
    return s
```

åęĆædIJä¡ăåŐżæţŇèŕŢçŽĎèŕÍïijŇä¡ăåřśäijŽåŘŞçŐřè£Źçğ■æŰźåijŔäijŽærŤä¡£çŤÍ
`translate()` æĹŰèĂĚæ■čåĹŹèąÍè¿¿åijŔèęĄå£ńå¿Ĺåd'ŽãĂĆ

åŘęäÿĂæŰźéÍćïijŇåęĆædIJä¡ăéIJĂèęĄæL'ğèąŇäżżä¡Ţåd'■æİĆå■Ůçňęåŕźå■Ůçňę̆ŽĎéĞ■æŰřæŸăåřĎæ
`tanslate()` æŰźæşŢäijŽéİđåÿÿçŽĎå£ńãĂĆ

äžŐåd'ğçŽĎæŰźéİćæİěèőšïijŇåŕźäžŐä¡ăçŽĎåžŤçŤÍçÍŃåžŘæİěèŕť'æĂğèČ¡æŸŕä¡ääÿ■å¿Ůäÿ■åŐżèĞłåů
äÿ■åźÿçŽĎæŸŕïijŇæĹŚäžňäÿ■åŘŕèČ¡çżŹä¡ăåżžèőőäÿĂäÿłçL'źåőŽçŽĎæŁĂæIJŕïijŇä¡£åőČèČ¡åd'§éĂĆåžŤa
åŻäæ■d'åőđéŹĚæČĚåĘtäÿ■éIJĂèęĄä¡ăèĞłåůšåŐżåřİèŕŢäÿ■åŘŇçŽĎæŰźæşŢåźűèŕĎäijřåőČãĂĆ

åř¡çőąè£ŹäÿĂèŁĆéŻĘäÿ■èőĺèőžçŽĎæŸŕæŰĞæIJňïijŇä¡ĘæŸŕçśżäijijçŽĎæŁĂæIJŕäź§åŘŕäžěéĂĆçŤĺäž

2.13 å■Ůçňęäÿšåŕźé¡Ř

éŮőécŸ

ä¡ăæČşéĂŽè£Ğæ§Řçğ■åŕźé¡ŘæŰźåijŔæİěæäijåijŔåŇŰå■Ůçňęäÿš

èğčåĘşæŰźæąĹ

åŕźäžŐå§žæIJňçŽĎå■Ůçňęäÿšåŕźé¡ŘæŞ■ä¡IJïijŇåŘŕäžěä¡£çŤĺå■ŮçňęäÿšçŽĎ `ljust()`
, `rjust()` åŠŇ `center()` æŰźæşŢãĂĆærŤåęĆïijŽ

```
>>> text = 'Hello World'
>>> text.ljust(20)
```

```
'Hello World          '
>>> text.rjust(20)
'          Hello World'
>>> text.center(20)
'    Hello World     '
>>>
```

æL’ĂæIJL’è£ŹäžŻæŰźæşŢéČ¡èČ¡æŐěåŔŮäÿĂäÿłåŔŕéĂL’çŽĎåąńåĚĚå■Ůçňęã ĂĆæŕŤåęĆïijŽ

```
>>> text.rjust(20,'=')
'==========Hello World'
>>> text.center(20,'*')
'****Hello World*****'
>>>
```

åĞ¡æŢř format() åŘŇæăůåŔŕäžěçŤĺæİěå¿Ĺåőźæ ŸŞçŽĎåŕźé¡Řå■Ůçňęäÿšã ĂĆ ä¡ăèęĄåĄŽçŽĎåřśæŸŕä¡£çŤĺ <, > æĹŰèĂĚ ^ å■Ůçňęå ŘŐéİćçť ğèů§äÿĂäÿłæŇĞåőŽçŽĎåő¡åžęã ĂĆæŕŤåęĆ

```
>>> format(text, '>20')
'          Hello World'
>>> format(text, '<20')
'Hello World          '
>>> format(text, '^20')
'    Hello World     '
>>>
```

åęĆæđIJä¡ăæČşæŇĞåőŽäÿĂäÿłéİđçl’žæăijçŽĎåąńåĚĚå■Ůçňęïij ŇårĘåőČåĘŹåĹřåŕźé¡Řå■Ůçňęç ŽĎåL’■

```
>>> format(text, '=>20s')
'==========Hello World'
>>> format(text, '*^20s')
'****Hello World*****'
>>>
```

å¡Şæăijåij ŔåŇŰåd’Žäÿłå Ăijç ŽĎæŮűåĂŹïijŇè£ŹäžŻæăijåijŔäžčçăĄäź§åŔŕäžěèćńçŤĺåIJÍ format() æŰźæşŢäÿ■ãĂĆæŕŤåęĆïijŽ

```
>>> '{:>10s} {:>10s}'.format('Hello', 'World')
'     Hello      World'
>>>
```

format() åĞ¡æŢřçŽĎäÿĂäÿłåě¡åd’ĎæŸŕåőČäÿ■äžĚéĂĆçŤĺäžŐå■Ůçňęäÿšã ĂĆåőČåŔŕäžěçŤĺæİěæăij æŕŤåęĆïijŇä¡ååŔŕäžěçŤĺåőČæİěæăijåijŔåŇŰæŢřå■ŮïijŽ

```
>>> x = 1.2345
>>> format(x, '>10')
'    1.2345'
>>> format(x, '^10.2f')
'   1.23   '
>>>
```

èőÍèőž

åIJÍèĂĄçŽĎäžčçăĄäÿ■ïijŇä¡ăçżŔåÿÿäijŽçIJŃåĹřèćńçŤÍæİěæäijåijŔåŇŰæŰĞæIJňçŽĎ % æŞ■ä¡IJçňęãĂĆærŤåęĆïijŽ

```
>>> '%-20s' % text
'Hello World          '
>>> '%20s' % text
'          Hello World'
>>>
```

ä¡ĘæŸŕïijŇåIJÍæŰřçĽĹæIJňäžčçăĄäÿ■ïijŇä¡ăåžŤèŕěäijŸåĚĹéĂĽ'æŃľ format() åĞ¡æŢřæĹŰèĂĚæŰźæşŢãĂĆ format() èęĄærŤ % æŞ■ä¡IJçňęçŽĎåŁ§èČ¡æŻť'äÿžåijžåď'ğãĂĆ åźűäÿŤ format() äź§ærŤä¡£çŤÍ ljust() , rjust() æĹŰ center() æŰźæşŢæŻť'éĂŽçŤÍïijŇ åŻääÿžåőČåŔŕäžěçŤÍæİěæäijåijŔåŇŰäżżæĎŔåŕźèśąïijŇèĂŇäÿ■äžĚäżĚæŸŕå■Ůçňęäÿšã ĂĆ

åęĆæđIJæČşèęĄåőŇåĚÍäžĘèğč format() åĞ¡æŢřçŽĎæIJL'çŤÍçL'źæĂğïijŇ èŕůåŔĆèĂČ åIJÍçž£PythonæŰĞæąč

2.14 åŘĹåźűæŃijæŐěå■Ůçňęäÿš

éŮőécŸ

ä¡ăæČşåřĘåĞääÿłåřŔçŽĎå■Ůçňęäÿšå ŘĹåźűäÿžäÿĂäÿłåď'ğçŽĎå■Ůçňęäÿš

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşèęĄåŘĹåźűçŽĎå■Ůçňęäÿšæ ŸŕåIJÍäÿĂäÿłåžŔåĹŮæĹŰèĂĚ iterable äÿ■ïijŇéĆčäźĹæIJĂå£nçŽĎæŰźåijŔåřśæŸŕä¡£çŤÍ join() æŰźæşŢãĂĆærŤåęĆïijŽ

```
>>> parts = ['Is', 'Chicago', 'Not', 'Chicago?']
>>> ' '.join(parts)
'Is Chicago Not Chicago?'
>>> ','.join(parts)
'Is,Chicago,Not,Chicago?'
>>> ''.join(parts)
'IsChicagoNotChicago?'
>>>
```

åĹİçIJŃètůæİěïijŇè£Źçğ■èŕ■æşŢçIJŃäÿŁåŐżäijŽærŤè¿ČæĂłïijŇä¡ĘæŸŕ join() èćńæŇĞåőŽäÿžå■Ůçňęäÿšç ŽĎäÿĂäÿłæŰźæşŢãĂĆ è£Źæăůå ĄŽçŽĎéČĺåĹĘåŐ§åŻăæŸŕä¡ăæČşåŐżè£đæŐěçŽĎåŕźèśąåŔŕèČ¡æİěèĞłåŘĎçğ■äÿ■åŘŇçŽĎæŢŕæ■ åęĆæđIJåIJÍæĽĂæIJL'è£ŹäžŽåŕźèśąäÿŁéČ¡åőŽäźĽäÿĂäÿł join() æŰźæşŢæŸŐæŸ¿æŸŕåĘŮä¡ŹçŽĎãĂĆ åŻăæ■d'ä¡ăåŘłéIJĂèęĄæŇĞåőŽä¡ăæČşèęĄçŽĎåĹĘåĽšå■Ůçňęäÿšåź join() æŰźæşŢåŐżåřĘæŰĞæIJňçĽĞæőţçżĎåŘĹètůæİěãĂĆ

åęĆæđIJä¡ăäżĚäżĚåŘłæŸŕåŘĹåźűåřŚæŢřåĞääÿłå■ŮçňęäÿšïijŇä¡£çŤĺåŁăåŔů(+)éĂŽåÿÿåůšçżŔèűşåď'§

```
>>> a = 'Is Chicago'
>>> b = 'Not Chicago?'
>>> a + ' ' + b
'Is Chicago Not Chicago?'
>>>
```

åŁääŔů(+)æŞ■ä¡IJçňęåIJĺä¡IJäÿžäÿĂäžŻåď■æİĆå■ŮçňęäÿšæäijåijŔåŇŰçŽĎæŻ£äžčæŰźæąĹçŽĎæŮűå

```
>>> print('{} {}'.format(a,b))
Is Chicago Not Chicago?
>>> print(a + ' ' + b)
Is Chicago Not Chicago?
>>>
```

åęĆæđIJä¡ăæČşåIJĺæžŘçăĄäÿ■årĘäÿd'äÿłå■ŮéİĆå■Ůçňęäÿšå ŘĹåźűètůæİěïijŇä¡ääŔłéIJÄèęĄçőÄå■ŢçŽ

```
>>> a = 'Hello' 'World'
>>> a
'HelloWorld'
>>>
```

èőĺèőž

å■Ůçňęäÿšå ŘĹåźűåŔŕèČ¡çIJŃäÿŁåŐżåźűäÿ■éIJÄèęĄçŤĺäÿĂæŢt'èŁĆæİěèőĺèőžãĂĆ
ä¡ĘæŸŕäÿ■åžŤèŕêårŘçIJŃè£ŹäÿłéŮőécŸïijŇçĺŃåžŔåŚŸéĂŽåÿÿåIJĺå■ŮçňęäÿšæäijåijŔåŇŰçŽĎæŮűåĂŹåŻ

æIJĂéĞ■èęĄçŽĎéIJÄèęĄåijŢètůæşĺæĎŘçŽĎæŸŕïijŇå¡ŞæĹŚäżňä¡£çŤĺåŁääŔů(+)æŞ■ä¡IJçňęåŐżè£đæ
åŻääÿžåŁääŔůè£đæŐëäijŽåijŢètůåĘĚå■Ÿåď■åĹűäžëåŘŁåđČåIJ¿åŻđæŤűæŞ■ä¡IJãĂĆ
çL'źåĹńçŽĎïijŇä¡ăæřÿè£IJéČ¡äÿ■åžŤåČŘäÿŃéİcè£ŹæăůåĘŹå■Ůçňęäÿšè£đæŐëäžčçăĄïijŽ

```
s = ''
for p in parts:
    s += p
```

è£Źçğ■åĘŹæşŢäijŽæŕŤä¡£çŤĺ `join()` æŰźæşŢè£ŘèąŇçŽĎèęĄæĚćäÿĂäžŻïijŇåŻääÿžæŕŘäÿĂæňąæL'
ä¡ăæIJĂåě¡æŸŕåĚĹæŤűéŻĘæL'ĂæIJL'çŽĎå■Ůçňęäÿšçl'ĞæőţçĎűåŘŐåĘ■ářĘåőČäžňè£đæŐëèţůæİěãĂĆ

äÿĂäÿłçŻÿåŕźæŕŤè¿ČèĄłæŸŐçŽĎæŁĂåůğæŸŕåĹl'çŤĺçŤ§æĹŘåŹĺèąĺè¿¿åijŔ(åŔĆèĂČ1.19ářŘèŁĆ)è¡ňa

```
>>> data = ['ACME', 50, 91.1]
>>> ','.join(str(d) for d in data)
'ACME,50,91.1'
>>>
```

åŘŇæăůè£Ÿå¿ŮæşĺæĎŘäÿ■å£ĚèęĄçŽĎå■Ůçňęäÿšè£đæŐëæŞ■ä¡IJãĂĆæIJL'æŮűåĂŹçĺŃåžŔåŚŸåIJĺæ

```
print(a + ':' + b + ':' + c) # Ugly
print(':'.join([a, b, c])) # Still ugly
print(a, b, c, sep=':') # Better
```

å¡Şæůůå ŘĹä¡£çŤÍI/OæŞ■ä¡IJåŠŇå■Ůçňęäÿšè£đæŐěæŞ■ä¡IJçŽĎæŮűůåĂŹïijŇæIJL’æŮűůåĂŹéIJÄèęĄäżŤ
ærŤåęĆïijŇèĂČèŹŚäÿŃéİćçŽĎäÿď’çńŕäžčçăĄçL’Ğæőţïijž

```
# Version 1 (string concatenation)
f.write(chunk1 + chunk2)

# Version 2 (separate I/O operations)
f.write(chunk1)
f.write(chunk2)
```

åęĆæđIJäÿď’äÿłå■Ůçňęäÿšå¿ĹářŕïijŇéĆčäźĹçňňäÿĂäÿłçL’ĹæIJňæĂğèČ¡äijŽæŻť’åě¡äžŻïijŇåŻääÿžI/Oç
åŔęåď’Űäÿ ĂæŰźéİćïijŇåęĆæđIJäÿď’äÿłå■Ůçňęäÿšå¿Ĺåď’ğïijŇéĆčäźĹçňňäžŇäÿłçL’ĹæIJňåŔŕèČ¡äijŽæŻť’åŁä
åŻääÿžåőČéĄ£åĚ■äžĘąĹŻåžžäÿĂäÿłå¿Ĺåď’ğçŽĎäÿť’æŮűçżŞæđIJåźűäÿŤèęĄåď’■åĹűåď’ğéĞŔçŽĎåĘĚå■Ÿå
è£ŸæŸŕéĆčåŔěèŕÍïijŇæIJL’æŮűåĂŹæŸŕéIJÄèęĄæäźæ■őä¡äçŽĎåžŤçŤÍçÍŃåžŔçL’źçĆźæİěåĘşåőŽåžŤèŕěä¡£

æIJĂåŘŐèřĹäÿĂäÿŃïijŇåęĆæđIJä¡ääĞĘåď’ĞçijŮåĘŹæđĎåžžåď’ğéĞŔåřŔå■Ůçňęäÿšç ŽĎè¿ŞåĞžäžčçăA
ä¡ăæIJĂåě¡èĂČèŹŚäÿŃä¡£çŤÍçŤ§æĹŘåŹĺåĞ¡æŢřïijŇåĹl’çŤÍyieldèŕ■åŔěäžğçŤ§è¿ŞåĞžçL’Ğæőţã ĆæŕŤåęĆ

```
def sample():
    yield 'Is'
    yield 'Chicago'
    yield 'Not'
    yield 'Chicago?'
```

è£ŹçğæŰźæşŢäÿĂäÿłæIJL’èűčçŽĎæŰźéİćæŸŕåőČåźűæšąæIJL’åŕźè¿ŞåĞžçL’Ğæőţå Ľřåž Ţèę ĄæĂŐæăů
ä¿ŃåęĆïijŇä¡ääŔŕäżěçőĂå■ŢçŽĎä¡£çŤÍ `join()` æŰźæşŢårĘè£ŹäžŻçL’Ğæőţå ŘĹåźűèţůæİěïijž

```
text = ''.join(sample())
```

æĹŰèĂĚä¡ääź§åŔŕäżěårĘå■Ůçňęäÿšç L’Ğæőţé Ğ■åőŽåŘŠåĽřI/Oïijž

```
for part in sample():
    f.write(part)
```

åĘ■æĹŰèĂĚä¡äè£ŸåŔŕäżěåĘŹåĞžäÿĂäžŻçżŞåŘĹI/OæŞ■ä¡IJçŽĎæůůåŘĹæŰźæąĽïijž

```
def combine(source, maxsize):
    parts = []
    size = 0
    for part in source:
        parts.append(part)
        size += len(part)
        if size > maxsize:
            yield ''.join(parts)
            parts = []
            size = 0
        yield ''.join(parts)

# çżŞåŘĹæŰĞäżűæŞ■ä¡IJ
with open('filename', 'w') as f:
    for part in combine(sample(), 32768):
        f.write(part)
```

è£ŹéĞŇçŽĎåĚşéŤőçĆźåIJÍäžŐåŐ§ågŃçŽĎçŤ§æĹŘåŹÍåĞ¡æŢřåźűäÿ■éIJĂèęAç§ěéĄŞä¡£çŤÍçżĘèŁĆïij

2.15 å■Ůçňęäÿšäÿ■æŔŠåĚěåŔŸéĞŔ

éŮőécŸ

ä¡ăæČşåĹŻåžžäÿĂäÿłåĘĚåţŇåŔŸéĞŔçŽĎå■ŮçňęäÿšïijŇåŔŸéĞŔèćńåőČçŽĎåĂijæĽĂèąÍçď žçŽĎå■Ů

èğčåĘşæŰźæąĹ

PythonåźűæšąæIJL'åŕźåIJÍå■Ůçňęäÿšäÿ■çőĂå■ŢæŻ£æ■ćåŔŸéĞŔåĂijæŔŘä¿ŻçŻť'æŐěçŽĎæŤŕæŇĄãĂ
ä¡ĘæŸŕéĂŽè£Ğä¡£çŤÍå■Ůçňęäÿšç ŽĎ `format()` æŰźæşŢæİěèğčåĘşè£ŹäÿłéŮőécŸãĂĆærŤåęĆïijŽ

```
>>> s = '{name} has {n} messages.'
>>> s.format(name='Guido', n=37)
'Guido has 37 messages.'
>>>
```

æĹŰèĂĚïijŇåęĆæđIJèęĄèćńæŻ£æ■çŽĎåŔŸéĞŔèČ¡åIJÍåŔŸéĞŔå§§äÿ■æL'¿åĹřïijŇ
éĆčäźĹä¡ääŘŕäžěçżŞåŘĹä¡£çŤÍ `format_map()` åŠŇ `vars()`
ãĂĆåřśåČŘäÿŃéİćè£ŹæăůïijŽ

```
>>> name = 'Guido'
>>> n = 37
>>> s.format_map(vars())
'Guido has 37 messages.'
>>>
```

`vars()` è£ŸæIJL'äÿĂäÿłæIJL'æĎŔæĂİçŽĎçL'źæĂğåřśæŸŕåőČäź§éĂĆçŤÍäžŐåŕźèśąåőđä¿ŃãĂĆærŤå

```
>>> class Info:
...     def __init__(self, name, n):
...         self.name = name
...         self.n = n
...
>>> a = Info('Guido',37)
>>> s.format_map(vars(a))
'Guido has 37 messages.'
>>>
```

`format` åŠŇ `format_map()` çŽĎäÿĂäÿłçijžéŹůåřśæŸŕåőČäžňåźűäÿ■èČ¡å¿Ĺåě¡çŽĎåď ĎçŘĘåŔŸéĞ

```
>>> s.format(name='Guido')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'n'
>>>
```

äÿĂçğ■éĄ£åĚ■è£Źçğ■éŤŹèŕŕçŽĎæŰźæşŢæŸŕåŔęåď Ůåő ŽäźĽäÿĂäÿłåŘńæIJĽ __missing__() æŰźæşŢçŽĎå■ŮåĚÿåŕźèśąïijŇåřšåČŔäÿŃéİćè£Źæăůïijž

```
class safesub(dict):
"""éŸšæ■ćkeyæL'¿äÿ■åĹř"""
def __missing__(self, key):
    return '{' + key + '}'
```

çŐřåIJĺä¡ăåŔŕäżěåĹľçŤĺè£ŹäÿłçśżåŇĚèčĚè¿ŞåĚěåŘŐäijăéĂŠçžŹ format_map() ïijž

```
>>> del n # Make sure n is undefined
>>> s.format_map(safesub(vars()))
'Guido has {n} messages.'
>>>
```

åęĆæđIJä¡ăåŔŠçŐřèĞłåůśåIJĺäżčçăĄäÿ■éćŠçźĄçŽĎæL'ğèąŇè£ŹäžŻæ■ěéłď'ïijŇä¡ăåŔŕäżěåřĘåŔŸéĞŔa

```
import sys

def sub(text):
    return text.format_map(safesub(sys._getframe(1).f_locals))
```

çŐřåIJĺä¡ăåŔŕäżěåČŔäÿŃéİćè£ŹæăůåĘŹäžĘïijž

```
>>> name = 'Guido'
>>> n = 37
>>> print(sub('Hello {name}'))
Hello Guido
>>> print(sub('You have {n} messages.'))
You have 37 messages.
>>> print(sub('Your favorite color is {color}'))
Your favorite color is {color}
>>>
```

èőĺèőž

åď Žåźť äżěæİěçŤśäžŐPythonçijžäźŔåŕźåŔŸéĞŔæŻ£æ■ćçŽĎåĘĚç¡őæŤŕæŇĄèĂŇåŕijèĞť äžĘåŘĎçğ■äÿ ä¡IJäÿžæIJňèŁĆäÿ■åśŢçď žçŽĎäÿĂäÿłåŔŕèČ¡çŽĎèğčåĘşæŰźæąĹïijŇä¡ăåŔŕäżěæIJĽæŮűåĂŹäijŽçIJŃåĹřåČ

```
>>> name = 'Guido'
>>> n = 37
>>> '%(name) has %(n) messages.' % vars()
'Guido has 37 messages.'
>>>
```

ä¡ăåŔŕèČ¡è£ŸäijŽçIJŃåĹřå■Ůçňęäÿšæĺąæİ£çŽĎä¡£çŤĺïijž

```
>>> import string
>>> s = string.Template('$name has $n messages.')
>>> s.substitute(vars())
'Guido has 37 messages.'
>>>
```

çĎűèĂŇïijŇ `format()` åŠŇ `format_map()` çŻÿæŕŤè¿ČäÿŁéİćè£ŹäžŻæŰźæąĹèĂŇåůšæŻť åŁăåĚĹ ä¡£çŢĺ `format()` æŰźæşŢè£ŸæIJL'äÿĂäÿłåě¡åď'ĎåŕśæŸŕä¡ăåŔŕäżěèŐůå¿ŮåŕźåŋŮçňęäÿšæăijåijŔåŇŰçŽĎ èĂŇè£ŹäžŻçL'źæĂğæŸŕä¡£çŢĺåČŔæĺąæİ£åŋŮçňęäÿšäźŃçśżçŽĎæŰźæąĹäÿŋåŔŕèČ¡èŐůå¿ŮçŽĎãĂĆ

æIJňæIJžè£ŸéČĺåĹĘäżŃçż■äžĘäÿĂäžŻénŸçžğçL'źæĂğãĂĆæŸăåřĎæĹŰèĂĚå■ŮåĚÿçśžäÿ■éšIJäÿžäžž `__missing__()` æŰźæşŢåŔŕäžěèől'ä¡ăåőŽäźL'åęĆä¡Ţåď'ĎçŘĘçijžåď'śçŽĎåĂijãĂĆ åIJĺ `SafeSub` çśżäÿ■ïijŇè£ŹäÿłæŰźæşŢèćńåőŽäźL'äÿžåŕźçijžåď'śçŽĎåĂijè£ŤåŻđäÿĂäÿłå■ää¡■çňęãĂĆ ä¡ăåŔŕäžěåŔŚçŐřçijžåď'śçŽĎåĂijäijŽåĞžçŐřåIJĺçżŞæđIJå■Ůçňęäÿšäÿ■(åIJĺèřČèŕŢçŽĎæŮűåĂŹåŔŕèČ¡å¿Ĺa `KeyError` åijĆåÿÿãĂĆ

`sub()` åĞ¡æŢřä¡£çŢĺ `sys._getframe(1)` è£ŤåŻđèřČçŢĺèĂĚçŽĎæăĹåÿğãĂĆåŔŕäžěäžŐäÿ■èő£éŮ `f_locals` æİèèŐůå¿ŮåśĂéČĺåŔŸéĞŔãĂĆ æŕńæŮăçŰŚéŮőçżİåď'ğéČĺåĹĘæČĚåĘţäÿŇåIJĺäžčçăĄäÿ■åŐżç ä¡ĘæŸŕïijŇåŕźäžŐåČŔå■ŮçňęäÿšæŻ£æ■ćåůěåĚůåĞ¡æŢřèĂŇèĺĂåőČæŸŕéIđåÿÿæIJL'çŢĺçŽĎãĂĆ åŔęåď'ŰïijŇåĂijå¿ŮæşĺæĎŔçŽĎæŸŕ `f_locals` æŸŕäÿĂäÿłåď'■åĹűèřČçŢĺåĞ¡æŢřçŽĎæIJňåIJŕåŔŸéĞŔçŽ åř¡çőąä¡ăåŔŕäžěæŤźåŔŸ `f_locals` çŽĎåĘĚåőźïijŇä¡ĘæŸŕè£ŹäÿłâŁőæŤźåŕźäžŐåŘŐéİćçŽĎåŔŸéĞŔèő£é æĽĂäżěïijŇèŹ¡èŕť'èő£éŮőäÿĂäÿłæăĹåÿğçIJŃäÿŁåŐżå¿ĹéĆłæĄűïijŇä¡ĘæŸŕåŕźåőČçŽĎäżżä¡ŢæŞ■ä¡IJäÿ■ä

2.16 äžěæŇĞåőŽåĹŮåő¡æăijåijŔåŇŰå■Ůçňęäÿš

éŮőécŸ

ä¡ăæIJL'äÿĂäžŻéŢ£å■ŮçňęäÿšïijŇæČşäžěæŇĞåőŽçŽĎåĹŮåő¡åřĘåőČäżňéĞ■æŰřæăijåijŔåŇŰãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŢĺ `textwrap` æĺąåİŮæİěæăijåijŔåŇŰå■ŮçňęäÿšçŽĎè¿ŞåĞžãĂĆærŤåęĆïijŇåĄĞåęĆä¡ăæIJL'äÿŇå

```
s = "Look into my eyes, look into my eyes, the eyes, the eyes, \
the eyes, not around the eyes, don't look around the eyes, \
look into my eyes, you're under."
```

äÿŃéİćæijŤçd'žä¡£çŢĺ `textwrap` æăijåijŔåŇŰå■ŮçňęäÿšçŽĎåd'Žçğ■æŰźåijŕïijŽ

```
>>> import textwrap
>>> print(textwrap.fill(s, 70))
Look into my eyes, look into my eyes, the eyes, the eyes, the eyes,
not around the eyes, don't look around the eyes, look into my eyes,
you're under.

>>> print(textwrap.fill(s, 40))
Look into my eyes, look into my eyes,
the eyes, the eyes, the eyes, not around
```

```
the eyes, don't look around the eyes,
look into my eyes, you're under.

>>> print(textwrap.fill(s, 40, initial_indent='    '))
    Look into my eyes, look into my
eyes, the eyes, the eyes, the eyes, not
around the eyes, don't look around the
eyes, look into my eyes, you're under.

>>> print(textwrap.fill(s, 40, subsequent_indent='    '))
Look into my eyes, look into my eyes,
    the eyes, the eyes, the eyes, not
    around the eyes, don't look around
    the eyes, look into my eyes, you're
    under.
```

èőÍèőž

textwrap æÍąåİŮåŕźäžŐå■Ůçňęäÿšæ L’Şå■řæŸŕéİđåÿÿæIJL’çŤÍçŽĎïijŇçL’źåĹńæŸŕå¡Şä¡ăåÿŇæIJŻè¿Ş ä¡ăåŔŕäžěä¡£çŤÍ os.get_terminal_size() æŰźæşŢæİèèŐůåŔŰçżĹçńŕçŽĎåd’ğářŔářžåŕÿãĂĆærŤåęĆ

```
>>> import os
>>> os.get_terminal_size().columns
80
>>>
```

fill() æŰźæşŢæŐěåŔŮäÿĂäžŻåĚűäžŰåŔŕéĂL’åŔĆæŢřæİěæŐğåĹűtabïijŇèŕ■åŔěçżŞář¿ç■L’ãĂĆ åŔĆéŸĚ textwrap.TextWrapperæŰĞæąč èŐůåŔŰæŻt’åd’ŽåĘĚåőźãĂĆ

2.17 åIJĺå■Ůçňęäÿšäÿ■åd’ĎçŘĘhtmlåŠŇxml

éŮőécŸ

ä¡ăæČşåřĘHTMLæĹŰèĂĚXMLåőđä¡ŞåęĆ &entity; æĹŰ &#code; æŻ£æ■ćäÿžåŕźåžŢçŽĎæŰĞæIJñãĂĆ åĘ■èĂĚïijŇä¡ăéIJĂèęĄè¡ňæ■ćæŰĞæIJñäÿ■çL’źåőŽçŽĎå■Ůçňę(æŕŤåę >, æĹŰ &)ãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşæŻ£æ■ćæŰĞæIJňå■Ůçňęäÿšäÿ■çŽĎ ‘<’ æĹŰèĂĚ ‘>’ ïijŇä¡£çŤÍ html.escape() åĞ¡æŢřåŔŕäžěå¿ĹåőźæŸŞçŽĎåőŇæĹŘãĂĆærŤåęĆïijŽ

```
>>> s = 'Elements are written as "<tag>text</tag>".'
>>> import html
>>> print(s)
Elements are written as "<tag>text</tag>".
```

```
>>> print(html.escape(s))
Elements are written as &quot;&lt;tag&gt;text&lt;/tag&gt;&quot;.

>>> # Disable escaping of quotes
>>> print(html.escape(s, quote=False))
Elements are written as "&lt;tag&gt;text&lt;/tag&gt;".
>>>
```

åęĆæđIJä¡ăæ■čåIJĺåď ĎçŘĘçŽĎæŸŕASCIIæŰĞæIJňïijŇåźűäÿŤæČşåřĘéİđASCIIæŰĞæIJňåŕźåžŤçŽĎç
åŔŕäžěçżŹæ§ŘäžŻI/OåĞ¡æŢřäijăéĂŠåŔĆæŢř `errors='xmlcharrefreplace'`
æİěè¿¿åĹřè£ŹäÿłçŻőãĂĆæŕŤåęĆïijŽ

```
>>> s = 'Spicy JalapeÃśo'
>>> s.encode('ascii', errors='xmlcharrefreplace')
b'Spicy Jalape&#241;o'
>>>
```

äÿžäžĘæŻ£æ■ćæŰĞæIJňäÿ■çŽĎçijŰçăĄåőđä¡ŞïijŇä¡ăéIJĂèęĄä¡£çŤĺåŔęåď ŰäÿĂçğ■æŰźæşŢãĂĆ
åęĆæđIJä¡ăæ■čåIJĺåď ĎçŘĘHTMLæĹŰèĂĚXMLæŰĞæIJňïijŇèŕŢçİĂåĚĹä¡£çŤĺäÿĂäÿłåŘĹéĂĆçŽĎHTML
éĂŽåÿÿæČĚåĘţäÿŃïijŇè£ŹäžŻåůěåĚůäijŽèĞłåŁĺæŻ£æ■ćè£ŹäžŻçijŰçăĄåĂijïijŇä¡ăæŮăéIJĂæŃĚå£ČãĂĆ

æIJL'æŮűåĂŹïijŇåęĆæđIJä¡ăæŐěæŤűåĹřäžĘäÿĂäžŻåŘńæIJL'çijŰçăĄåĂijçŽĎåŐ§ågŃæŰĞæIJňïijŇél
éĂŽåÿÿä¡ăåŘłéIJĂèęĄä¡£çŤÍHTMLæĹŰèĂĚXMLèğčæđŘåŹÍçŽĎäÿĂäžŻçŻÿåĚşåůěåĚůåĞ¡æŢř/æŰźæşŢå■

```
>>> s = 'Spicy &quot;Jalape&#241;o&quot.'
>>> from html.parser import HTMLParser
>>> p = HTMLParser()
>>> p.unescape(s)
'Spicy "JalapeÃśo".'
>>>
>>> t = 'The prompt is &gt;&gt;&gt;'
>>> from xml.sax.saxutils import unescape
>>> unescape(t)
'The prompt is >>>'
>>>
```

èőÍèőž

åIJÍçŤ§æĹŘHTMLæĹŰèĂĚXMLæŰĞæIJňçŽĎæŮűåĂŹïijŇåęĆæđIJæ■čçąőçŽĎè¡ñæ■ćçL'źæőŁæăĞèć
çL'źåĹńæŸŕå¡Şä¡ăä¡£çŤÍ `print()` åĞ¡æŢřæĹŰèĂĚåĚűäžŰå■ŮçňęäÿšæăijåijŘåŇŰæİěäžğçŤ§è¿ŞåĞžçŽĎa
ä¡£çŤĺåČŔ `html.escape()` çŽĎåůěåĚůåĞ¡æŢřåŕäžěå¿Ĺåőźæ ŸŞçŽĎèğčåĘşè£ŹçśźéŰőécŸãĂĆ

åęĆæđIJä¡ăæČşäžěåĚűäžŰæŰźåijŔåď ĎçŘĘæŰĞæIJňïijŇè£ŸæIJL'äÿĂäžŻåĚűäžŰçŽĎåůěåĚůåĞ¡æŢřa
`xml.sax.saxutils.unescape()` åŔŕäžěåÿőåŁľä¡ăãĂĆ
çĎűěĂŇïijŇä¡ăåžŤèŕěåĚĹèŕČçăŤæÿĚæěŽæĂŐæăůä¡£çŤĺäÿĂäÿłåŘĹéĂĆçŽĎèğčæđŘåŹĺãĂĆ
æŕŤåęĆïijŇåęĆæđIJä¡ăåIJĺåď ĎçŘĘHTMLæĹŰXMLæŰĞæIJňïijŇ
ä¡£çŤĺæ§ŘäÿłèğčæđŘæĺąåİŮæŕŤåęĆ `html.parse` æĹŰ `xml.etree.ElementTree`
åůšçżŔåÿőä¡ăèĞłåŁĺåď ĎçŘĘäžĘçŻÿåĚşçŽĎæŻ£æ■ćçżĘèŁĆãĂĆ

2.18 å■Ůçňęäÿšäžd'çL'ŇèğčæđŘ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłå■ŮçňęäÿšïijŇæČşäžŐåůęèĞşåŔşåřĘåĚűèğčæđŘäÿžäÿĂäÿłäžd'çL'ŇæţĄãĂĆ

èğčåĘşæŰźæąĹ

åĄĞåęĆä¡ăæIJL'äÿŃéÍćè£ŹæăůäÿĂäÿłæŰĞæIJňå■ŮçňęäÿšïijŽ

```
text = 'foo = 23 + 42 * 10'
```

äÿžäžĘäžd'çL'ŇåŇŰå■ŮçňęäÿšïijŇä¡ääÿ■äžĚéIJÄèęĄåŇźéĚ■æĺąåijŔïijŇè£Ÿå¿ŮæŇĞåőŽæĺąåijŔçŽĎçærŤåęĆïijŇä¡ääŘŕèČ¡æČşářĘå■ŮçňęäÿšåČŔäÿŃéÍćè£ŹæăůèĄñæ■ćäÿžäžŔåĹŮářźïijŽ

```
tokens = [('NAME', 'foo'), ('EQ','='), ('NUM', '23'), ('PLUS','+'),
    ('NUM', '42'), ('TIMES', '*'), ('NUM', '10')]
```

äÿžäžĘæL'ğèąŇè£ŹæăůçŽĎåĹĞåĹĘïijŇçňňäÿĂæ■ěářśæŸŕåČŔäÿŃéÍćè£ŹæăůåĹl'çŤĺåŠ¡åŘ■æ■ŢèŐůçż

```
import re
NAME = r'(?P<NAME>[a-zA-Z_][a-zA-Z_0-9]*)'
NUM = r'(?P<NUM>\d+)'
PLUS = r'(?P<PLUS>\+)'
TIMES = r'(?P<TIMES>\*)'
EQ = r'(?P<EQ>=)'
WS = r'(?P<WS>\s+)'

master_pat = re.compile('|'.join([NAME, NUM, PLUS, TIMES, EQ, WS]))
```

åIJÍäÿŁéÍćçŽĎæĺąåijŔäÿ■ïijŇ `?P<TOKENNAME>` çŤĺäžŐçżŹäÿĂäÿłæĺąåijŔåŚ¡åŘ■ïijŇä¿ŻåŘŐéÍćä¡£çŤ

äÿŇäÿĂæ■ěïijŇäÿžäžĘäžd'çL'ŇåŇŰïijŇä¡£çŤĺæĺąåijŔåŕźèśąå¿ĹåřŚèćńäžžç§ěéĄŞçŽĎ `scanner()` æŰźæşŢãĂĆ è£ŹäÿłæŰźæşŢäijŽåĹŻåžžäÿĂäÿł `scanner` åŕźèśąïijŇ åIJÍè£ŹäÿłåŕźèśąäÿŁäÿ■æŰ■çŽĎèřČçŤÍ `match()` æŰźæşŢäijŽäÿĂæ■ěæ■ěçŽĎæL'ńæŔŘçŻőæăĞæŰŰĞæIJňïijŇærŔæ■ěäÿĂäÿłåŇźéĚ■ãĂĆ äÿŃéÍćæŸŕæijŤçd'žäÿĂäÿł `scanner` åŕźèśąåęĆä¡Ţåůěä¡IJçŽĎäžd'äžŠåijŔä¿Ńå■ŘïijŽ

```
>>> scanner = master_pat.scanner('foo = 42')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('NAME', 'foo')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('WS', ' ')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
```

```
>>> _.lastgroup, _.group()
('EQ', '=')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('WS', ' ')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('NUM', '42')
>>> scanner.match()
>>>
```

åőđéŹĚä¡£çŤÍè£Źçğ■æŁĂæIJŕçŽĎæŮűåĂŹïijŇåŔŕäžěå¿Ĺåőźæ ŸŞçŽĎåČŔäÿŃéÍćè£ŹæăůårĘäÿŁè£ŕäž

```
def generate_tokens(pat, text):
Token = namedtuple('Token', ['type', 'value'])
scanner = pat.scanner(text)
for m in iter(scanner.match, None):
    yield Token(m.lastgroup, m.group())

# Example use
for tok in generate_tokens(master_pat, 'foo = 42'):
    print(tok)
# Produces output
# Token(type='NAME', value='foo')
# Token(type='WS', value=' ')
# Token(type='EQ', value='=')
# Token(type='WS', value=' ')
# Token(type='NUM', value='42')
```

åęĆæđIJä¡ăæČşè£ĞæżđʼäżđʼçĽŇæţĄïijŇä¡ăåŔŕäžěåőŽäźĽæŻtʼådʼŽçŽĎçŤ§æĹŘåŹĺåĞ¡æŢřæĹŰèĂĚä¡ æŕŤåęĆïijŇäÿŃéÍćæijŤçđʼžæĂŐæăůè£ĞæżđʼæĽĂæIJĽçŽĎçlʼžçŹ¡äżđʼçĽŇïijŽ

```
tokens = (tok for tok in generate_tokens(master_pat, text)
            if tok.type != 'WS')
for tok in tokens:
    print(tok)
```

èőÍèőž

éĂŽåÿÿæÍěèőšäžđʼçĽŇåŇŰæŸŕå¿ĹåđʼŽéńŸçžğæŰĞæIJňèğčæđŘäÿŐåđʼĎçŘĘçŽĎçňňäÿĂæ■ẽãĂĆ äÿžäžĘä¡£çŤÍäÿŁéÍćçŽĎæĽńæŔŘæŰźæşŢïijŇä¡ăéIJĂèęĄèőřä¡Ŕè£ŹéĞŇäÿĂäžŻéĞ■èęĄçŽĎåĞäçĆźãĂĆ çňňäÿĂçĆźåřśæŸŕä¡ăå£Ěéążçąõèőđʼä¡ää¡£çŤÍæ■čåĹŹèąÍè¿¿åijŔæŇĞåőŽäžĘæĽĂæIJĽè¿ŞåĚěäÿ■åŔŕèČ¡åĞ åęĆæđIJæIJĽäżżä¡Ţäÿ■åŔŕåŇźéĚ■çŽĎæŰĞæIJňåĞžçŐřäžĘïijŇæĽńæŔŘåřśäijŽçŻtʼæŐěåĄIJæ■ćãĂĆè£Źä

äžđʼçĽŇçŽĎéąžåžŔäź§æŸŕæIJĽå¡śåŞ■çŽĎãĂĆ `re` æĺąåİŮäijŽæŇĽʼçĚğæŇĞåőŽåě¡çŽĎéąžåžŔåŐżåĄ åŻäæ■đʼïijŇåęĆæđIJäÿĂäÿłæĺąåijŔæĄřåě¡æŸŕåŔęäÿĂäÿłæŻtʼéŢ£æĺąåijŔçŽĎå■Řå■ŮçňęäÿšïijŇéĆčäźĹä¡ăé

```
LT = r'(?P<LT><)'
LE = r'(?P<LE><=)'
EQ = r'(?P<EQ>=)'

master_pat = re.compile('|'.join([LE, LT, EQ])) # Correct
# master_pat = re.compile('|'.join([LT, LE, EQ])) # Incorrect
```

çñňäžŇäýłæĺąåijŕæŸŕéŤŹçŽĎïijŇåŻääýžåőČäijŽåřĘæŰĞæIJň<=åŇźéĚ■äýžäžd'çL'ŇLTçť ğèů§çİĂEQi

æIJĂåŘŐïijŇä¡ăéIJĂèęĄçŢŹæĎŔäýŃå■Řå■Ůçňęäýšå¡ćåijŔçŽĎæĺąåijŔãĂĆæŕŤåęĆïijŇåĄĞèő¿ä¡ăæIJI

```
PRINT = r'(?P<PRINT>print)'
NAME = r'(?P<NAME>[a-zA-Z_][a-zA-Z_0-9]*)'

master_pat = re.compile('|'.join([PRINT, NAME]))

for tok in generate_tokens(master_pat, 'printer'):
    print(tok)

# Outputs :
# Token(type='PRINT', value='print')
# Token(type='NAME', value='er')
```

åĚşäžŐæŻť'énŸéŸűçŽĎäžd'çL'ŇåŇŰæŁĂæIJŕïijŇä¡ăåŔŕèČ¡éIJĂèęĄæ§ěçIJŃ PyPars-
ing æĹÚèĂĚ PLY åŇĚãĂĆ äýĂäýłèřČçŤÍPLYçŽĎä¿Ńå■ŘåIJĺäýŃäýĂèŁĆäijŽæIJL'æijŤçd'žãĂĆ

2.19 åőđçŐřäýĂäýłçőĂå■ŢçŽĎéĂŠå¡ŠäýŃéŹ■åĹĘæđŘåŹÍ

éŮőéćŸ

ä¡ăæČşæăźæ■őäýĂçżĎèŕ■æşŢèğĎåĹŹèğčæđŘæŰĞæIJňåźűæL'ğèąŇåŚ¡äžd'ïijŇæĹŰèĂĚæđĎéĂăäýĂäj
åęĆæđIJèŕ■æşŢéİđåÿÿçőĂå■ŢïijŇä¡ăåŔŕäžěèĞłåůśåĘŹè£ŹäýłèğčæđŘåŹĺïijŇèĂŇäÿ■æŸŕä¡£çŤĺäýĂäžŻæąĘ

èğčåĘşæŰźæąĹ

åIJÍè£ŹäýłéŮőécŸäÿ■ïijŇæĹŚäžňéŽĘäÿ■èőĺèőžæăźæ■őçL'źæőŁèŕ■æşŢåŐżèğčæđŘæŰĞæIJňçŽĎéŮőé
äÿžäžĘè£ŹæăůåĄŽïijŇä¡ăéęŰåĚĹèęĄäžěBNFæĹŰèĂĚEBNFå¡ćåijŔæŇĞåőŽäÿĂäýłæăĞåĞĘèŕ■æşŢãĂĆ
æŕŤåęĆïijŇäÿĂäýłçőĂå■ŢæŢřå■ęèąĺè¿¿åijŔèŕ■æşŢåŔŕèČ¡åČŔäýŃéİćè£Źæăůïijž

```
expr ::= expr + term
    |    expr - term
    |    term

term ::= term * factor
    |    term / factor
    |    factor
```

```
factor ::= ( expr )
    |   NUM
```

æĹŰèĂĚïijÑäžěEBNFå¡ćåijŔïijŽ

```
expr ::= term { (+|-) term }*

term ::= factor { (*|/) factor }*

factor ::= ( expr )
    |   NUM
```

åIJÍEBNFäÿ■ïijÑèćńåŇĚåŘńåIJÍ { . . . }* äÿ■çŽĎèğĎåĹŹæŸŕåŘŕéĂL'çŽĎãĂĆ*äžčèąĺ0æňąæĹŰåď'Žæ

çŐřåIJÍïijÑåęĆæđIJä¡ăåŕźBNFçŽĎåůěä¡IJæIJžåĹűè£Ÿäÿ■æŸŕå¿ĹæŸŐçŹ¡çŽĎèŕİïijÑåřśæŁŁåőČå¡ŞåĄ
äÿĂèĹňæİěëőšïijÑèğčæđŘçŽĎåŐ§çŘĘåřśæŸŕä¡ääĹl'çŤÍBNFåőŇæĹŘåď'Žäÿ łæŻ£æ■ćåŠŇæL'l'åśŢäžěåŇźéĚ
äÿžäžĘæijŤçď'žïijÑåĄĞèő¿ä¡ăæ■čåIJÍèğčæđŘå¡ćåęĆ 3 + 4 * 5 çŽĎèąĺè¿¿åijŔãĂĆ
è£Źäÿłèąĺè¿¿åijŔåĚĹèęĄéĂŽè£Ğä¡£çŤÍ2.18èŁĆäÿ■äžŃçż■çŽĎæŁĂæIJŕåĹĘèğčäÿžäÿĂçżĎäžď'çL'ŇæţĄãĂ
çżŞæđIJåŘŕèČ¡æŸŕåČŔäÿŇåĹŮè£ŹæăůçŽĎäžď'çL'ŇåžŘåĹŮïijŽ

```
NUM + NUM * NUM
```

åIJÍæ■d'å§žçąĂäÿŁïijÑ èğčæđŘåŁĺä¡IJäijŽèŕŢçİĂåŐžéĂŽè£ĞæŻ£æ■ćæŞ■ä¡IJåŇźéĚ■èŕ■æşŢåĹřè¿ŞåĚ

```
expr
expr ::= term { (+|-) term }*
expr ::= factor { (*|/) factor }* { (+|-) term }*
expr ::= NUM { (*|/) factor }* { (+|-) term }*
expr ::= NUM { (+|-) term }*
expr ::= NUM + term { (+|-) term }*
expr ::= NUM + factor { (*|/) factor }* { (+|-) term }*
expr ::= NUM + NUM { (*|/) factor}* { (+|-) term }*
expr ::= NUM + NUM * factor { (*|/) factor }* { (+|-) term }*
expr ::= NUM + NUM * NUM { (*|/) factor }* { (+|-) term }*
expr ::= NUM + NUM * NUM { (+|-) term }*
expr ::= NUM + NUM * NUM
```

äÿŇéİćæL'ĂæIJL'çŽĎèğčæđŘæ■ěéłď'åŘŕèČ¡éIJĂèęĄèŁśçĆźæŮűéŮt'åijĎæŸŐçŹ¡ïijÑä¡ĘæŸŕåőČäžňåĆ
çňňäÿĂäÿłè¿ŞåĚěäžď'çL'ŇæŸŕfNUMïijÑåŽăæ■d'æŻ£æ■ćéęŰåĚĹäijŽåŇźéĚ■éĆčäÿłéČĺåĹĘãĂĆ
äÿĂæŮęåŇźéĚ■æĹŘåŁ§ïijÑåřśäijŽè£ŽåĚěäÿŇäÿĂäÿłäžď'çL'Ň+ïijÑäžěæ■d'çśźæŐĺãĂĆ
å¡ŞåůšçżŕçąőåőŽäÿ■èČ¡åŇźéĚ■äÿŇäÿĂäÿłäžď'çL'ŇçŽĎæŮűåĂŹïijÑåŘşè¿źçŽĎéČĺåĹĘ(æŕŤåęĆ
{ (*/) factor }*)åřśäijŽèćńæÿĚçŘĘæŐL'ãĂĆ åIJĺäÿĂäÿłæĹŘåŁ§çŽĎèğčæđŘäÿ■ïijÑæŢt'äÿłåŘşè¿źé

æIJL'äžĘåL'■éİćçŽĎç§ěèŕĘèČŇæŹŕïijÑäÿŇéİćæĹŚäžňäÿ¿äÿĂäÿłçőĂå■Ţçď'žä¿ŇæİěåśŢçď'žåęĆä¡Ţæđĺ

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
"""
Topic: äÿŃéŹ■èğčæđŘåŹĺ
Desc :
"""
```

```
import re
import collections

# Token specification
NUM = r'(?P<NUM>\d+)'
PLUS = r'(?P<PLUS>\+)'
MINUS = r'(?P<MINUS>-)'
TIMES = r'(?P<TIMES>\*)'
DIVIDE = r'(?P<DIVIDE>/)'
LPAREN = r'(?P<LPAREN>\()'
RPAREN = r'(?P<RPAREN>\))'
WS = r'(?P<WS>\s+)'

master_pat = re.compile('|'.join([NUM, PLUS, MINUS, TIMES,
                                  DIVIDE, LPAREN, RPAREN, WS]))
# Tokenizer
Token = collections.namedtuple('Token', ['type', 'value'])


def generate_tokens(text):
    scanner = master_pat.scanner(text)
    for m in iter(scanner.match, None):
        tok = Token(m.lastgroup, m.group())
        if tok.type != 'WS':
            yield tok


# Parser
class ExpressionEvaluator:
    '''
    Implementation of a recursive descent parser. Each method
    implements a single grammar rule. Use the ._accept() method
    to test and accept the current lookahead token. Use the ._
↪expect()
    method to exactly match and discard the next token on on the␣
↪input
    (or raise a SyntaxError if it doesn't match).
    '''

    def parse(self, text):
        self.tokens = generate_tokens(text)
        self.tok = None  # Last symbol consumed
        self.nexttok = None  # Next symbol tokenized
        self._advance()  # Load first lookahead token
        return self.expr()

    def _advance(self):
        'Advance one token ahead'
        self.tok, self.nexttok = self.nexttok, next(self.tokens,␣
↪None)
```

```
    def _accept(self, toktype):
        'Test and consume the next token if it matches toktype'
        if self.nexttok and self.nexttok.type == toktype:
            self._advance()
            return True
        else:
            return False

    def _expect(self, toktype):
        'Consume next token if it matches toktype or raise␣
↪SyntaxError'
        if not self._accept(toktype):
            raise SyntaxError('Expected ' + toktype)

    # Grammar rules follow
    def expr(self):
        "expression ::= term { ('+'|'-') term }*"
        exprval = self.term()
        while self._accept('PLUS') or self._accept('MINUS'):
            op = self.tok.type
            right = self.term()
            if op == 'PLUS':
                exprval += right
            elif op == 'MINUS':
                exprval -= right
        return exprval

    def term(self):
        "term ::= factor { ('*'|'/') factor }*"
        termval = self.factor()
        while self._accept('TIMES') or self._accept('DIVIDE'):
            op = self.tok.type
            right = self.factor()
            if op == 'TIMES':
                termval *= right
            elif op == 'DIVIDE':
                termval /= right
        return termval

    def factor(self):
        "factor ::= NUM | ( expr )"
        if self._accept('NUM'):
            return int(self.tok.value)
        elif self._accept('LPAREN'):
            exprval = self.expr()
            self._expect('RPAREN')
            return exprval
        else:
            raise SyntaxError('Expected NUMBER or LPAREN')
```

```
def descent_parser():
    e = ExpressionEvaluator()
    print(e.parse('2'))
    print(e.parse('2 + 3'))
    print(e.parse('2 + 3 * 4'))
    print(e.parse('2 + (3 + 4) * 5'))
    # print(e.parse('2 + (3 + * 4)'))
    # Traceback (most recent call last):
    #   File "<stdin>", line 1, in <module>
    #   File "exprparse.py", line 40, in parse
    #   return self.expr()
    #   File "exprparse.py", line 67, in expr
    #   right = self.term()
    #   File "exprparse.py", line 77, in term
    #   termval = self.factor()
    #   File "exprparse.py", line 93, in factor
    #   exprval = self.expr()
    #   File "exprparse.py", line 67, in expr
    #   right = self.term()
    #   File "exprparse.py", line 77, in term
    #   termval = self.factor()
    #   File "exprparse.py", line 97, in factor
    #   raise SyntaxError("Expected NUMBER or LPAREN")
    #   SyntaxError: Expected NUMBER or LPAREN


if __name__ == '__main__':
    descent_parser()
```

èőÍèőž

æŰĞæIJňèğčæđŘæŸŕäÿĂäÿłå¿Ĺåď ğçŽĎäÿžécŸïijŇ äÿĂèĹňäijŽå■ăçŤĺå■ęçŤ§å■ęäźăçijŰèŕŚèŕ¿çÍŃæŮ
åęĆæđIJä¡ăåIJÍæL’¿åŕżåĚşäžŐèŕ■æşŢïijŇèğčæđŘçőŮæşŢç■L’çŻÿåĚşçŽĎèČŇæŹŕç§ěèŕEçŽĎèŕÍïijŇä¡ăåžŤèŕ
å¿ĹæŸ¿çĎűïijŇåĚşäžŐè£ŹæŰźéÍćçŽĎåĘĚåőźåď łåď ŽïijŇäÿ■åŘŕèČ¡åIJÍè£ŹéĞŇåĚĺéČĺĺśŢåijĂãĂĆ

åř¡çőąåęĆæ■d’ïijŇçijŰåĘŹäÿĂäÿłéĂŠå¡ŠäÿŃéŹ■èğčæđŘåŹÍçŽĎæŢť ä¡ŞæĂÌèůŕæŸŕæŕŤè¿ČçőĂå■ŢçŽ
åijĂåğŃçŽĎæŮűůåĂŹïijŇä¡ăåĚĹèŐůå¿ŮæL’ĂæIJL’çŽĎèŕ■æşŢèğĎåĹŹïijŇçĎűåŘŐářĘåĚűè¡ňæ■ćäÿžäÿĂäÿ
åŽăæ■d’åęĆæđIJä¡ăçŽĎèŕ■æşŢçśżäijijè£Źæăůïijž

```
expr ::= term { ('+'|'-') term }*

term ::= factor { ('*'|'/') factor }*

factor ::= '(' expr ')'
    | NUM
```

ä¡ăåžŤèŕěéęŰåĚĹářĘåőČäżňè¡ňæ■ćæĹŘäÿĂçžĎåČŔäÿŃéÍćè£ŹæăůçŽĎæŰźæşŢïijž

```
class ExpressionEvaluator:
    ...
    def expr(self):
    ...
    def term(self):
    ...
    def factor(self):
    ...
```

æŕŘäÿłæŰźæşŢèęĄåőŇæĹŘçŽĎäżżåŁąå¿ĹçőĂå■Ţ - åőČå£ĚéążäżŐåůęèĞşåŘşéĄ■åŐĘèŕ■æşŢèğĎåĹŹäżŐæ§ŘçğĭæĎŘäźĽäÿŁèőšïijŇæŰźæşŢçŽĎçŻőçŽĎåřśæŸŕèęĄäźĹåď'ĎçŘĘåőŇèŕ■æşŢèğĎåĹŹïijŇèęĄäźĹäÿžäžĘè£ŹæăůåĄŽïijŇéIJĂéĞĞçŤĺäÿŇéĺćçŽĎè£ŹäžŻåőđçŐřæŰźæşŢïijŽ

- åęĆæđIJèğĎåĹŹäÿ■çŽĎäÿŇäÿłçňęåŘůæŸŕåŘęåď'ŰäÿĂäÿłèŕ■æşŢèğĎåĹŹçŽĎåŘ■å■Ů(æŕŤåęĆtermæè£ŹåřśæŸŕèŕěçőŮæşŢäÿ■"äÿŇéŹ■"çŽĎçŤśæİě - æŐğåĹűäÿŇéŹ■åĹřåŘęäÿĂäÿłèŕ■æşŢèğĎåĹŹäÿ■åŐæIJL'æŮűåĂŹèğĎåĹŹäijŽèřČçŤĺåůšçżŘæL'ğèąŇçŽĎæŰźæşŢ(æŕŤåęĆïijŇåIJĺ
  `factor ::= '('expr ')'` äÿ■åŕźexprçŽĎèřČçŤĺ)ãĂĆ
  è£ŹåřśæŸŕçőŮæşŢäÿ■"éĂŠå¡Š"çŽĎçŤśæİěãĂĆ

- åęĆæđIJèğĎåĹŹäÿ■äÿŇäÿĂäÿłçňęåŘůæŸŕäÿłçL'źæőŁçňęåŘů(æŕŤåęĆ()ïijŇä¡åå¿Ůæ§ěæL'¿äÿŇäÿĂäÿåęĆæđIJäÿ■åŇźéĚ■ïijŇåřšäžğçŤ§äÿĂäÿłèŕ■æşŢéŤŹèŕŕãĂĆè£ŹäÿĂèŁĆäÿ■çŽĎ
  `_expect()` æŰźæşŢåřśæŸŕçŤĺæİěåĄŽè£ŹäÿĂæ■ěçŽĎãĂĆ

- åęĆæđIJèğĎåĹŹäÿ■äÿŇäÿĂäÿłçňęåŘůäÿžäÿĂäžŻåŘŕèČ¡çŽĎéĂL'æŇl'éąź(æŕŤåęĆ + æĹŰ -)ïijŇ ä¡ăå£ĚéążåŕźæŕŘäÿĂçğ■åŘŕèČ¡æČĚåĘţæčĂæ§ěäÿŇäÿĂäÿłäžd'çL'ŇïijŇåŘłæIJL'å¡Şåőčå
  è£Źäź§æŸŕæIJñèŁĆçd'žä¿Ńäÿ■ `_accept()` æŰźæşŢçŽĎçŽőçŽĎãĂĆ
  åőČçŻÿå¡ŞäžŐ_expect()æŰźæşŢçŽĎåijśåŇŰçL'ĹæIJñïijŇåŻääÿžåęĆæđIJäÿĂäÿłåŇźéĚ■æL'¿åĹřäžĘåä¡ĘæŸŕåęĆæđIJæšąæL'¿åĹřïijŇåőČäÿ■äijŽäžğçŤ§éŤŹèŕŕèĂŇæŸŕåŻđæżŽ(åĚĄèőÿåŘŐçż■çŽĎæčĂæ

- åŕźäžŐæIJL'éĞ■åď'■éČĺåĹĘçŽĎèğĎåĹŹ(æŕŤåęĆåIJĺèğĎåĹŹèąĺè¿¿åijŔ `::= term {('+'|'-') term }*` äÿ■)ïijŇ éĞ■åď'■åŁĺä¡IJéĂŽè£ĞäÿĂäÿłwhileå¿łçŐŕæİěåőđçŐřãĂĆ
  å¿łçŐŕäÿżä¡ŞäijŽæŤűéŻĘæĹŰåď'ĎçŘĘæL'ĂæIJL'çŽĎéĞ■åď'■åĚČçť'äçŻť'åĹřæšąæIJL'åĚűäžŰåĚČçť'ă

- äÿĂæŮęæŢť'äÿłèŕ■æşŢèğĎåĹŹåď'ĎçŘĘåőŇæĹŘïijŇæŕŘäÿłæŰźæşŢäijŽè£ŤåŻđæ§Řçğ■çżŞæđIJçżŹèè£ŹåřśæŸŕåIJĺèğčæđŘè£ĞçÍŇäÿ■åĂijæŸŕæĂŐæăůçť'ŕåŁäçŽĎåŐ§çŘĘãĂĆ
  æŕŤåęĆïijŇåIJĺèąĺè¿¿åijŔæśĆåĂijçÍŇäžŘäÿ■ïijŇè£ŤåŻđåĂijäžčèąĺèąĺè¿¿åijŔèğčæđŘåŘŐçŽĎéČĺåĹĘçæIJĂåŘŐæL'ĂæIJL'åĂijäijŽåIJĺæIJĂéąűåśĆçŽĎèŕ■æşŢèğĎåĹŹæŰźæşŢäÿ■åŘĹäźűèţůæİěãĂĆ

åř¡çőąåŘŚä¡ăæijŢçď'žçŽĎæŸŕäÿĂäÿłçőĂå■ŢçŽĎä¿Ńå■ŘïijŇéĂŠå¡ŠäÿŇéŹ■èğčæđŘåŹĺåŕźäžèçŤĺæİěåæŕŤåęĆïijŇPythonèŕ■èĺĂæIJñèžńåřśæŸŕéĂŽè£ĞäÿĂäÿłéĂŠå¡ŠäÿŇéŹ■èğčæđŘåŹĺåŐžèğčéĞŁçŽĎãĂĆ
åęĆæđIJä¡ăåŕźæ■ď'æĎ§åĚť'ëűčïijŇä¡ăåŘŕäžěéĂŽè£Ğæ§ěçIJŃPythonæžŘçăĄæŰĞäžűGrammar/GrammaræİçIJŃåőŇä¡ăäijŽåŘŠçŐřïijŇéĂŽè£ĞæL'ŃåŁĺæŰźåijŘåŐžåőđçŐřäÿĂäÿłèğčæđŘåŹĺåĚűåőđäijŽæIJL'å¿Ĺåď'Žç

åĚűäÿ■äÿĂäÿłåśĂéŹŘåřśæŸŕåőČäžňäÿ■èČ¡èćńçŤĺäžŐåŇĚåŘńäżżä¡Ţåůę éĂŠå¡ŠçŽĎèŕ■æşŢèğĎåĹŹäÿ

```
items ::= items ',' item
    | item
```

äÿžäžĘè£ŹæăůåĄŽïijŇä¡ăåŘŕèČ¡äijŽåČŘäÿŇéİćè£ŹæăůäĄ£çŤĺ `items()` æŰźæşŢïijŽ

```
def items(self):
    itemsval = self.items()
```

```
    if itemsval and self._accept(','):
        itemsval.append(self.item())
    else:
        itemsval = [ self.item() ]
```

åŤŕäÿĂçŽĎéŮőéćŸæŸŕè£ŹäÿłæŰźæşŢæăźæIJňäÿ■èČ¡åůěä¡IJïijŇäžŃåőđäÿŁïijŇåőČäijŽäžğçŤ§äÿĂäÿ

åĚşäžŐèŕ■æşŢèğĎåĹŹæIJňèžńä¡ăåŔŕèČ¡äź§äijŽçćřåĹřäÿĂäžŻæčŸæĽ'ŃçŽĎéŮőéćŸãĂĆ æŕŤåęĆïijŇä¡ăåŔŕèČ¡æČşç§ěéĄŞäÿŃéİćè£ŹäÿłçőĂå■ŢæĽ'ijèŕ■æşŢæŸŕåŘęèąĺè£řå¿Ůå¡ŞïijŽ

```
expr ::= factor { ('+'|'-'|'*'|'/') factor }*

factor ::= '(' expression ')'
    | NUM
```

è£Źäÿłèŕ■æşŢçIJŃäÿŁåŐżæšąåŢěéŮőéćŸïijŇä¡ĘæŸŕåőČå■t'äÿ■èČ¡år§èğL'åĹřæăĞåĞĘåŻŻåĹŹè£ŘçőŮ æŕŤåęĆïijŇèąĺè¿¿åijŔ "3 + 4 * 5" äijŽå¿ŮåĹř35èĂŇäÿ■æŸŕæIJ§æIJŻçŽĎ23. åĹĘåijĂä¡£çŤĺ"expr"åŠŇ"term"èğĎåĹŹåŔŕäžěèőľ'åőČæ■čçąőçŽĎåůěä¡IJãĂĆ

åŕźäžŐåd'■æİĆçŽĎèŕ■æşŢïijŇä¡ăæIJĂåě¡æŸŕéĂL'æŃľæ§ŘäÿłèğčæđŘåůěåĚůæŕŤåęĆPyParsingæĹŰèĂ äÿŃéİćæŸŕä¡£çŤĺPLYæİěéĞ■åĘŹèąĺè¿¿åijŔæśĆåĂijçĺŃåžŔçŽĎäžčçăĄïijŽ

```
from ply.lex import lex
from ply.yacc import yacc

# Token list
tokens = [ 'NUM', 'PLUS', 'MINUS', 'TIMES', 'DIVIDE', 'LPAREN',
↪'RPAREN' ]
# Ignored characters
t_ignore = ' \t\n'
# Token specifications (as regexs)
t_PLUS = r'\+'
t_MINUS = r'-'
t_TIMES = r'\*'
t_DIVIDE = r'/'
t_LPAREN = r'\('
t_RPAREN = r'\)'

# Token processing functions
def t_NUM(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Error handler
def t_error(t):
    print('Bad character: {!r}'.format(t.value[0]))
    t.skip(1)

# Build the lexer
lexer = lex()
```

```
# Grammar rules and handler functions
def p_expr(p):
    '''
    expr : expr PLUS term
         | expr MINUS term
    '''
    if p[2] == '+':
        p[0] = p[1] + p[3]
    elif p[2] == '-':
        p[0] = p[1] - p[3]


def p_expr_term(p):
    '''
    expr : term
    '''
    p[0] = p[1]


def p_term(p):
    '''
    term : term TIMES factor
    | term DIVIDE factor
    '''
    if p[2] == '*':
        p[0] = p[1] * p[3]
    elif p[2] == '/':
        p[0] = p[1] / p[3]

def p_term_factor(p):
    '''
    term : factor
    '''
    p[0] = p[1]

def p_factor(p):
    '''
    factor : NUM
    '''
    p[0] = p[1]

def p_factor_group(p):
    '''
    factor : LPAREN expr RPAREN
    '''
    p[0] = p[2]

def p_error(p):
    print('Syntax error')
```

```
parser = yacc()
```

è£ŹäÿłçĺŃåžŔäÿ■ïijŇæĽĂæIJL'äžčçăĄéČ¡ä¡■äžŐäÿĂäÿłæŕŤè¿ČéńŸçŽĎåśĆæňąãĂĆä¡ääŔłéIJĂèęĄäÿ
èĂŇåőđéŹĚçŽĎè£ŘèąŇè§čæđŘåŹĺïijŇæŐěåŔŮäžď'çĽŇç■Ľ'ç■Ľ'åžŢåśĆåŁĺä¡IJåůšçżŔèćńåžŞåĞ¡æŢřåőđçČ

äÿŃéİćæŸŕäÿĂäÿłæĂŐæăůä¡£çŤĺå¿ŮåĹřçŽĎè§čæđŘåŕźèśąçŽĎä¿Ńå■ŘïijŽ

```
>>> parser.parse('2')
2
>>> parser.parse('2+3')
5
>>> parser.parse('2+(3+4)*5')
37
>>>
```

åęĆæđIJä¡ăæČşåIJĺä¡ăçŽĎçijŰçĺŃè£ĞçĺŃäÿ■æİěçĆźæŇŚæĹŸåŠŇåĹžæ£ĂïijŇçijŰåĘŹè§čæđŘåŹĺåŠŇ
åĘ■æňąïijŇäÿĂæIJňçijŰèřŚåŹĺçŽĎäźęçś■äijŽåŇĚåŘńå¿Łåď'ŽåžŢåśĆçŽĎçŘĘèőžç§ěèřĘãĂĆäÿ■è£Ğå¿Łåď'
PythonèĞłåůšçŽĎastæĺąåİŮäź§åĂijå¿ŮåŐžçIJŃäÿĂäÿŃãĂĆ

2.20 å■ŮèŁĆå■Ůçňęäÿšäÿ ŁçŽĎå■ŮçňęäÿšæŞ■ä¡IJ

éŮőécŸ

ä¡ăæČşåIJĺå■ŮèŁĆå■Ůçňęäÿšäÿ ŁæĽğèąŇæŹőéĂŽçŽĎæŰĞæIJňæŞ■ä¡IJ(æŕŤåęĆçğżéŹď'ïijŇæŘIJçť'ćå

èğčåĘşæŰźæąĹ

å■ŮèŁĆå■ŮçňęäÿšåŘŇæăůäź§æŤŕæŇĄåď'ğéČĺåĹĘåŠŇæŰĞæIJňå■Ůçňęäÿšäÿ ĂæăůçŽĎåĘĚç¡őæŞ■ä¡

```
>>> data = b'Hello World'
>>> data[0:5]
b'Hello'
>>> data.startswith(b'Hello')
True
>>> data.split()
[b'Hello', b'World']
>>> data.replace(b'Hello', b'Hello Cruel')
b'Hello Cruel World'
>>>
```

è£ŹäžŻæŞ■ä¡IJåŘŇæăůäź§éĂĆçŤĺäžŐå■ŮèŁĆæŢřçżĎãĂĆæŕŤåęĆïijŽ

```
>>> data = bytearray(b'Hello World')
>>> data[0:5]
bytearray(b'Hello')
>>> data.startswith(b'Hello')
True
```

```
>>> data.split()
[bytearray(b'Hello'), bytearray(b'World')]
>>> data.replace(b'Hello', b'Hello Cruel')
bytearray(b'Hello Cruel World')
>>>
```

ä¡ăåŔŕäžěä¡£çŤĺæ■čåĹŹèąĺè¿¿åijŔåŇźéĚ■å■ŮèŁĆå■ŮçňęäÿšïijŇä¡ĘæŸŕæ■čåĹŹèąĺè¿¿åijŔæIJňèžńå£Ě

```
>>>
>>> data = b'FOO:BAR,SPAM'
>>> import re
>>> re.split('[:,]',data)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "/usr/local/lib/python3.3/re.py", line 191, in split
return _compile(pattern, flags).split(string, maxsplit)
TypeError: can't use a string pattern on a bytes-like object
>>> re.split(b'[:,]',data) # Notice: pattern as bytes
[b'FOO', b'BAR', b'SPAM']
>>>
```

èőÍèőž

åď ğåď ŽæŢřæČĚåĘţäÿŃïijŇåIJÍæŰĞæIJňå■Ůçňęäÿšäÿ ŁçŽĎæŞ■ä¡IJåİĞåŔŕçŤĺäžŐå■ŮèŁĆå■Ůçňęäÿš
çĎűèĂŇïijŇè£ŹéĞŇäź§æIJL'äÿĂäžŹéIJĂèęĄæşÍæĎŔçŽĎäÿ■åŘŇçĆźãĂĆéęŰåĚĹïijŇå■ŮèŁĆå■Ůçňęäÿšç

```
>>> a = 'Hello World' # Text string
>>> a[0]
'H'
>>> a[1]
'e'
>>> b = b'Hello World' # Byte string
>>> b[0]
72
>>> b[1]
101
>>>
```

è£Źçğ■èŕ■äźL'äÿŁçŽĎåŇžåĹńäijŽåŕźäžŐåď ĎçŘĘéİćåŘŠå■ŮèŁĆçŽĎå■Ůçňęæ Ţřæ■őæIJL'å¡śåŞ■ãĂĆ

çňňäžŇçĆźïijŇå■ŮèŁĆå■Ůçňęäÿšäÿ■äijŽæŔŘä¿Żäÿ Ääÿłç¿ŐèğĆçŽĎå■Ůçňęäÿšèąĺçď žïijŇäź§äÿ■èČ¡å

```
>>> s = b'Hello World'
>>> print(s)
b'Hello World' # Observe b'...'
>>> print(s.decode('ascii'))
Hello World
>>>
```

çśžäijijçŽĎïijŇäź§äÿ■å■ŸåIJĺäżżä¡ŢéĂĆçŤĺäžŐå■ŮèŁĆå■ŮçňęäÿšçŽĎæäijåijŔåŇŰæŞ■ä¡IJïijŽ

```
>>> b'%10s %10d %10.2f' % (b'ACME', 100, 490.1)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for %: 'bytes' and 'tuple'
>>> b'{} {} {}'.format(b'ACME', 100, 490.1)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: 'bytes' object has no attribute 'format'
>>>
```

åęĆæđIJä¡ăæČşæăijåijŔåŇŰå■ŮèŁĆå■ŮçňęäÿšïijŇä¡ăå¿ŮåĚĹä¡£çŤĺæăĞåĞĘçŽĎæŰĞæIJňå■Ůçňęäÿš

```
>>> '{:10s} {:10d} {:10.2f}'.format('ACME', 100, 490.1).encode(
↪'ascii')
b'ACME 100 490.10'
>>>
```

æIJĂåŘŐéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijŇä¡£çŤĺå■ŮèŁĆå■ŮçňęäÿšåŔŕèČ¡äijŽæŤźåŔŸäÿĂäžŻæŞ■ä¡IJçŽĹ
æŕŤåęĆïijŇåęĆæđIJä¡ăä¡£çŤĺäÿĂäÿłçijŰçăĄäÿžå■ŮèŁĆçŽĎæŰĞäžűåŘ■ïijŇèĂŇäÿ■æŸŕäÿĂäÿłæŹőéĂŽçŽ

```
>>> # Write a UTF-8 filename
>>> with open('jalape\xf1o.txt', 'w') as f:
...     f.write('spicy')
...
>>> # Get a directory listing
>>> import os
>>> os.listdir('.') # Text string (names are decoded)
['jalapeÃśo.txt']
>>> os.listdir(b'.') # Byte string (names left as bytes)
[b'jalapen\xcc\x83o.txt']
>>>
```

æşlæĎŔä¿Ńå■Řäÿ■çŽĎæIJĂåŘŐéČĺåĹĘçżŹçŻőå¡ŢåŘ■äijăéĂŠäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšæŸŕæĂŐæăů
åIJĺçŻőå¡Ţäÿ■çŽĎæŰĞäžűåŘ■åŇĚåŘńåŐ§ågŃçŽĎUTF-8çijŰçăĄãĂĆ
åŔĆèĂČ5.15åřŔèŁĆèŐůåŔŰæŻť'åď'ŽæŰĞäžűåŘ■çŻÿåĚşçŽĎåĘĚåőźãĂĆ

æIJĂåŘŐæŔŘäÿĂçĆźïijŇäÿĂäžŻçĺŃåžŔåŚŸäÿžäžĘæŔŘå■ĞçĺŃåžŔæĽ'ğèąŇçŽĎéĂ§åžęäijŽåĂ¿åŘŚäž
åř¡çőąæŞ■ä¡IJå■ŮèŁĆå■ŮçňęäÿšçąőåőđäijŽæŕŤæŰĞæIJňæŻť'åŁäéńŸæŢĹ(åŽääÿžåď'ĎçŘĘæŰĞæIJňåŽžæI
è£Źæăůå ĄŽéĂŽåÿÿäijŽåŕijèĞť'éİđåÿÿæİĆäźśçŽĎäžčçăĄãĂĆä¡ääijŽçżŔåÿÿåŔŚçŐřå■ŮèŁĆå■Ůçňęäÿšåźűäÿ
åźűäÿŤä¡äè£Ÿå¿ŮæĽ'ŃåŁĺåď'ĎçŘĘæĽ'ĂæIJĽçŽĎçijŰçăĄ/èğčçăĄæŞ■ä¡IJãĂĆ
åİęçŹ¡èőšïijŇåęĆæđIJä¡ăåIJĺåď'ĎçŘĘæŰĞæIJňçŽĎèŕÏïijŇářśçŻť'æŐěåIJĺçĺŃåžŔäÿ■ä¡£çŤĺæŹőéĂŽçŽĎæŰĞ

çňňäÿĽ'çńăïijŽæŢřå■ŮæŮěæIJ§åŠŇæŮűéŮť

åIJĺPythonäÿ■æĽ'ğèąŇæŢť'æŢřåŠŇæţőçĆźæŢřçŽĎæŢřå■ęè£ŘçőŮæŮűå¿ĹçőĂå■ŢçŽĎãĂĆ
år¡çőąåęĆæ■ď'ïijŇåęĆæđIJä¡ăéIJĂèęĄæĽ'ğèąŇåĹĘæŢřãĂĄæŢřçżĎæĹŰèĂĚæŸŕæŮěæIJ§åŠŇæŮűéŮť'çŽĎ
æIJňçńăéŻĘäÿ■èőĺèőžçŽĎářśæŸŕè£ŹäžŻäÿžéćŸãĂĆ

Contents:

3.1 æŢřå■ÚçŽĎåŻŻèĹ■äžŤåĚě

éŮőécŸ

ä¡ăæČşåŕźæţőçĆźæŢřæĽğèąŇæŇĞåőŽçš¿åžęçŽĎèĹ■åĚěè£ŘçőŮãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐçőĂå■ŢçŽĎèĹ■åĚěè£ŘçőŮïijŇä¡£çŤĺåĘĚç¡őçŽĎ `round(value, ndigits)` åĞ¡æŢřå■şåŔŕãĂĆærŤåęĆïijŽ

```
>>> round(1.23, 1)
1.2
>>> round(1.27, 1)
1.3
>>> round(-1.27, 1)
-1.3
>>> round(1.25361,3)
1.254
>>>
```

å¡ŞäÿĂäÿłåĂijåĹŽåě¡åIJÍäÿď äÿłè¿źçŢŇçŽĎäÿ■éŮť çŽĎæŮűåĂŹïijŇ `round` åĞ¡æŢřè£ŤåŻđçżåőČæIJĂè£ŚçŽĎåĄűæŢřãĂĆ äźş§åřśæŸŕèřťïijŇåŕź1.5æĹŮèĂĚ2.5çŽĎèĹ■åĚěè£ŘçőŮéČ¡

äijăçżŹ `round()` åĞ¡æŢřçŽĎ `ndigits` åŔĆæŢřåŔŕäžěæŸŕèť §æŢřïijŇè£Źçğ■æČĚåĘţäÿŃïijŇèĹ■åĚěè£ŘçőŮäijŽä¡IJçŤĺåIJÍå■Ąä¡■ãĂĄçŹ¿ä¡■ãĂĄå■Čä¡■ç■L'äÿŁéİćãĂĆærŤåęĆïijŽ

```
>>> a = 1627731
>>> round(a, -1)
1627730
>>> round(a, -2)
1627700
>>> round(a, -3)
1628000
>>>
```

èőÍèőž

äÿ■èęĄåřĘèĹ■åĚěåŠŇæăijåijŔåŇŰè¿ŞåĞžæŘđæůûæůĘäžĘãĂĆ åęĆæđIJä¡ăçŽĎçŽőçŽĎåŘłæŸŕçőĂå■ŢçŽĎè¿ŞåĞžäÿĂåőŽåő¡åžęçŽĎæŢřïijŇä¡ăäÿ■éIJĂèęĄä¡£çŤÍ `round()` åĞ¡æŢřãĂĆ èĂŇäžĚäžĚåŘłéIJĂèęĄåIJÍæăijåijŔåŇŰçŽĎæŮűåĂŹæŇĞåőŽçš¿åžęå■şåŔŕãĂĆærŤ

```
>>> x = 1.23456
>>> format(x, '0.2f')
'1.23'
>>> format(x, '0.3f')
'1.235'
>>> 'value is {:0.3f}'.format(x)
```

```
'value is 1.235'
>>>
```

åŘŇæăůïijŇäÿ■èęĄèŕŢçİĂåŐżèĹ■åĚěæţőçĆźåĂijæİě”ä£őæ■č”èąĺéİćäÿŁçIJŃèţůæİěæ■čçąőçŽĎéŮőéć

```
>>> a = 2.1
>>> b = 4.2
>>> c = a + b
>>> c
6.300000000000001
>>> c = round(c, 2) # "Fix" result (???)
>>> c
6.3
>>>
```

åŕźäžŐåď˘ğåď’ŽæŢřä¡£çŤĺåĹřæţőçĆźçŽĎçĺŃåžŔïijŇæšąæIJL’å£ĚèęĄäź§äÿ■æŐĺè■Řè£ŹæăůåĄŽãĂĆ
år¡çőąåIJĺèőąçőŮçŽĎæŮűůåĂŹäijŽæIJL’äÿĂçĆźçĆźåřŔçŽĎèŕřåůőïijŇä¡ĘæŸŕè£ŹäžŽåřŔçŽĎèŕřåůőæŸŕèČ¡è
åęĆæđIJäÿ■èČ¡åĚĄèőÿè£Źæăůç˘ĎåřŔèŕřåůő(æŕŤåęĆæűĽ’åŘŁåĹřéĞŚèđ■éćĘå§§)ïijŇéĆčäźĹåřśå¿ŮèĂČèŹ
`decimal` æĺąåİŮäžĘïijŇäÿŃäÿĂèŁĆæĹŚäžňäijŽèŕęçżĘèőĺèőžãĂĆ

3.2 æĽ’ğèąŇçš¿çąőçŽĎæţőçĆźæŢřè£ŘçőŮ

éŮőécŸ

ä¡ăéIJĂèęĄåŕźæţőçĆźæŢřæĽ’ğèąŇçš¿çąőçŽĎèőąçőŮæŞ■ä¡IJïijŇåźűäÿŤäÿ■åÿŇæIJŻæIJL’äżżä¡ŢåřŔèŕŕ

èğčåĘşæŰźæąĹ

æţőçĆźæŢřçŽĎäÿĂäÿłæŹőéĄ■éŮőécŸæŸŕåőČäžňåźűäÿ■èČ¡çš¿çąőçŽĎèąĺçď˘žå■Ąè£ŻåĹűæŢřãĂĆ
åźűäÿŤïijŇå■şä¡£æŸŕæIJĂçőĂå■ŢçŽĎæŢřå■ęè£ŘçőŮäź§äijŽäžğçŤ§åřŔçŽĎèŕřåůőïijŇæŕŤåęĆïijŽ

```
>>> a = 4.2
>>> b = 2.1
>>> a + b
6.300000000000001
>>> (a + b) == 6.3
False
>>>
```

è£ŹäžŻéŤŹèŕŕæŸŕçŤśåžŢåśĆCPUåŠŇIEEE 754æăĞåĞĘéĂŽè£ĞèĞłåůśçŽĎæţőçĆźå■Ţä¡■åŐżæĽ’ğèąŇ
çŤśäžŐPythonçŽĎæţőçĆźæŢřæ■őçśżåđŃä¡£çŤĺåžŢåśĆèąĺçď˘žå■ŸåĆĺæŢřæ■őïijŇåŽăæ■d’ä¡ăæšąåŁđæşŢåŐ

åęĆæđIJä¡ăæČşæŻt’åŁăçš¿çąő(åźűèČ¡åőźå£■äÿĂåőŽçŽĎæĂğèČ¡æ■§èĂŮ)ïijŇä¡ăåŔŕäžěä¡£çŤĺ
`decimal` æĺąåİŮïijŽ

```
>>> from decimal import Decimal
>>> a = Decimal('4.2')
>>> b = Decimal('2.1')
```

```
>>> a + b
Decimal('6.3')
>>> print(a + b)
6.3
>>>  (a + b) == Decimal('6.3')
True
```

åĹİçIJŃètůæİěïijŇäÿŁéIćçŽĎäžčçăĄåě¡åČŕæIJL’çĆźåěĞæĂłïijŇæŕŤåęĆæĹŚäžňçŤĺå■Ůçňęäÿšæİěèąĺço
çĎűěĂŇïijŇ `Decimal` åŕźèśąäijŽåČŕæŹőéĂŽætőçĆźæŢřäÿĂæăůçŽĎåůěä¡IJ(æŤŕæŇĄæĽĂæIJL’çŽĎåÿÿçT
åęĆæđIJä¡ăæĽŞå■řáőČäžňæĹŰèĂĚåIJĺå■ŮçňęäÿšæăijåijŔåŇŰåĞ¡æŢřäÿ■ä¡£çŤĺáőČäžňïijŇçIJŃètůæİěèů§a

`decimal` æĺąåİŮçŽĎäÿĂäÿłäÿżèęĄçL’źå¿ĄæŸŕåĚĄèőÿä¡ăæŐğåĹűèőąçőŮçŽĎæŕŔäÿĂæŰźéIćïijŇåŇ
äÿžäžĘè£ŹæăůåĄŽïijŇä¡ăåĚĹå¿ŮåĹŻåžžäÿĂäÿłæIJňåIJřäÿŁäÿŃæŰĞåźűæŻť æŤźáőČçŽĎèő¿ç¡őïijŇæŕŤåęC

```
>>> from decimal import localcontext
>>> a = Decimal('1.3')
>>> b = Decimal('1.7')
>>> print(a / b)
0.76470588235294117647058823 53
>>> with localcontext() as ctx:
...     ctx.prec = 3
...     print(a / b)
...
0.765
>>> with localcontext() as ctx:
...     ctx.prec = 50
...     print(a / b)
...
0.76470588235294117647058823529411764705882352941176
>>>
```

èőÍèőž

`decimal` æĺąåİŮåőđçŐřäžĘIBMçŽĎ”éĂŽçŤĺářŔæŢřè£ŘçőŮègĎèŇČ”ãĂĆäÿ■çŤĺèŕť’ïijŇæIJL’å¿Ĺåď’Ž

PythonæŰřæL’ŃäijŽåĂ¿åŘŚäžŐä¡£çŤĺ `decimal` æĺąåİŮæİěåď’ĎçŘĘætőçĆźæŢřçŽĎçš¿çąőè£ŘçőŮãĂ
çĎűěĂŇïijŇåĚĹçŘĘèğčä¡ăçŽĎåžŤçŤĺçÍŃåžŔçŻőçŽĎæŸŕéİđåÿÿéĞ■èęĄçŽĎãĂĆ
åęĆæđIJä¡ăæŸŕåIJĺåĄŽçğŚå■ęèőąçőŮæĹŰåůěçÍŃéćĘå§§çŽĎèőąçőŮãĂĄçŤţèĎŚçżŸåŻ¿ïijŇæĹŰèĂĚæŸŕç
éĆčäźĹä¡£çŤĺæŹőéĂŽçŽĎætőçĆźçšžåđŇæŸŕæŕŤè¿ČæŹőéĄ■çŽĎåĄŽæşŢãĂĆ
åĚűäÿ■äÿĂäÿłåŐ§åŻăæŸŕïijŇåIJĺçIJ§åőđäÿŰçŢŃäÿ■å¿ĹářŚäijŽèęĄæśĆçš¿çąőåĽřæŹőéĂŽætőçĆźæŢřèČ¡æ
åŽăæ■ď’ïijŇèőąçőŮè£ĞçÍŃäÿ■çŽĎéĆčäźĹäÿĂçĆźçĆźçŽĎèŕŕåůőæŸŕèćńåĚĄèőÿçŽĎãĂĆ
çňňäžŇçĆźářśæŸŕïijŇåŐ§çŤ§çŽĎætőçĆźæŢřèőąçőŮèęĄå£ńçŽĎåď’Ž-
æIJL’æŮűåĂŹä¡ăåIJĺæL’ğèąŇåď’ğéĞŔè£ŘçőŮçŽĎæŮűåĂŹéĂ§åžęäź§æŸŕéİđåÿÿéĞ■èęĄçŽĎãĂĆ

å■şä¿£åęĆæ■ď’ïijŇä¡ăå■ť’äÿ■èČ¡åőŇåĚĺå£¡çŢèèŕŕåůőãĂĆæŢřå■ęåőűèŁśäžĘåď’ğéĞŔæŮűéŮť’åŐżçăŤçĺ
ä¡ääź§å¿ŮæşĺæĎŔäÿŃåĞŔæşŢåĹăéŹd’äžěåŔŁåď’ğæŢřåŠŇářŔæŤřçŽĎåŁăåĹĘè£ŘçőŮæL’ĂåÿęæİěçŽĎå¡šå

```
>>> nums = [1.23e+18, 1, -1.23e+18]
>>> sum(nums) # Notice how 1 disappears
```

```
0.0
>>>
```

äÿŁéÍćçŽĎéŤŹèŕŕåŔŕäžěåĹľçŤÍ `math.fsum()` æĽĂæŔŘä¿ŻçŽĎæŻť'çš¿çąőèőąçőŮèČ¡åŁŻæÍěèğčåĘ

```
>>> import math
>>> math.fsum(nums)
1.0
>>>
```

çĎűèĂŇïijŇåŕźäžŐåĚűäżŰçŽĎçőŮæşŢïijŇä¡ääžŤèŕěäżŤçżEçăŤçľ'űåőČåźűçŘĘèğčåőČçŽĎèŕŕåůőäžğçŤ

æĂżçŽĎæÍěèŕť'ïijŇ `decimal` æÍąåİŮäÿżèęĄçŤĺåIJÍæűĽ'åŔŁåĹřéĞŚèđ■çŽĎécĘå§§ãĂĆ
åIJÍè£Źçśżç|ŃåžŘäÿ■ïijŇåŞłæĂŢæŸŕäÿĂçĆźåřŔåřŘçŽĎèŕŕåůőåIJÍèőąçőŮè£ĞçÍŃäÿ■èŤŞåžűéČ¡æŸŕäÿ■åĚA
åŻăæ■d'ïijŇ `decimal` æÍąåİŮäÿżèğčåEşè£ŹçśżéŮőécŸæŔŘä¿ŻäžĘæŰźæşŢãĂĆ
å¡ŞPythonåŠŇæŢřæ■őåžŞæĽŞäžd'éĄŞçŽĎæŮűåĂŹäź§éĂŽåÿÿäijŽéĄĞåĹř `Decimal`
åŕźèśąïijŇåźűäÿŤïijŇéĂŽåÿÿäź§æŸŕåIJĺåd'ĎçŘĘéĞŚèđ■æŢřæ■őçŽĎæŮűåĂŹãĂĆ

3.3 æŢřå■ŮçŽĎæăijåijŔåŇŰè¿ŞåĞž

éŮőécŸ

ä¡äéIJĂèęĄåřĘæŢřå■ŮæăijåijŕåŇŰåŘŐè¿ŞåĞžïijŇåźűæŐğåĹűæŢřå■ŮçŽĎä¡■æŢřãĂĄåŕźé¡ŘãĂĄå■Č

èğčåĘşæŰźæąĹ

æăijåijŔåŇŰè¿ŞåĞžå■ŢäÿłæŢřå■ŮçŽĎæŮűåĂŹïijŇåŔŕäžěä¡£çŤĺåĘĚç¡őçŽĎ
`format()` åĞ¡æŢřïijŇærŤåęĆïijŽ

```
>>> x = 1234.56789

>>> # Two decimal places of accuracy
>>> format(x, '0.2f')
'1234.57'

>>> # Right justified in 10 chars, one-digit accuracy
>>> format(x, '>10.1f')
'    1234.6'

>>> # Left justified
>>> format(x, '<10.1f')
'1234.6    '

>>> # Centered
>>> format(x, '^10.1f')
'  1234.6  '

>>> # Inclusion of thousands separator
```

```
>>> format(x, ',')
'1,234.56789'
>>> format(x, '0,.1f')
'1,234.6'
>>>
```

åęĆæđIJä¡ăæČşä¡£çŤÍæŇĞæŢřèőřæşŢïijŇářĘfæŤźæĹŘeæĹŰèĂĚE(åŔŰåĘşäžŐæŇĞæŢřè¿ŞåĞžçŽĎåd

```
>>> format(x, 'e')
'1.234568e+03'
>>> format(x, '0.2E')
'1.23E+03'
>>>
```

åŘŇæŮűæŇĞåőŽåő¡åžęåŠŇçš¿åžęçŽĎäÿĂèĹňå¡ćåijŔæŸý
`'[<>^]?width[,]?(.digits)?'` ïijŇ åĚűäÿ■ `width` åŠŇ `digits` äÿžæŢťæŢřïijŇïij§äžčèąĺåŔŕéĂL'éČĺåĹĘãĂĆ åŘŇæăůçŽĎæăijåijŔäź§èćńçŤĺåIJĺå■Ůçňęäÿšç ŽĎ `format()` æŰźæşŢäÿ■ãĂĆæŕŤåęĆïijŽ

```
>>> 'The value is {:0,.2f}'.format(x)
'The value is 1,234.57'
>>>
```

èőĺèőž

æŢřå■Ůæăijåij ŔåŇŰè¿ŞåĞžéĂŽåÿÿæŸŕæŕŤè¿ČçőĂå■ŢçŽĎãĂĆäÿŁéĺćæijŤçď žçŽĎæŁĂæIJŕåŘŇæŮű `decimal` æĺąåİŮäÿ■çŽĎ `Decimal` æŢřå■Ůåŕźèśąã ĂĆ

å¡ŞæŇĞåőŽæŢřå■ŮçŽĎä¡■æŢřåŘőïijŇçżŞæđIJåĂijäijŽæăźæ■ő `round()` åĞ¡æŢřåŘŇæăůçŽĎèğĎåĹŹè£ŻèąŇåŻŻèĹ■äžŤåĚěåŘőè£ŤåŻđãĂĆæŕŤåęĆïijŽ

```
>>> x
1234.56789
>>> format(x, '0.1f')
'1234.6'
>>> format(-x, '0.1f')
'-1234.6'
>>>
```

åŇĚåŘńå■Čä¡■çňęçŽĎæăijåijŔåŇŰèů§æIJňåIJřåŇŰæšąæIJL'åĚşçşžãĂĆ åęĆæđIJä¡ăéIJĂèęĄæăźæ■őåIJřåŇžæÍèæŸ¿çď žå■Čä¡■çňęïijŇä¡ăéIJĂèęĄèĞłåůšåŐżèřČæ§ëäÿŃ `locale` æĺąåİŮäÿ■çŽĎåĞ¡æŢřäžĘãĂĆ ä¡ååŘŇæăůäź§åŔŕäžëä¡£çŤĺå■Ůçňęäÿšç ŽĎ `translate()` æŰźæşŢæÍëäžď æ■ćå■Čä¡■çňęãĂĆæŕŤåęĆïijŽ

```
>>> swap_separators = { ord('.'):',', ord(','):'.' }
>>> format(x, ',').translate(swap_separators)
'1.234,56789'
>>>
```

åIJÍå¿Ĺåď ŽPythonäžčçăĄäÿ■äijŽçIJŃåĹřä¡£çŤÍ%æİěæäijåijŔåŇŰæŢřå■ŮçŽĎïijŇæŕŤåęĆïijŽ

```
>>> '%0.2f' % x
'1234.57'
>>> '%10.1f' % x
'    1234.6'
>>> '%-10.1f' % x
'1234.6    '
>>>
```

è£Źçğ■æäijåijŔåŇŰæŰźæşŢäź§æŸŕåŔŕèąŇçŽĎïijŇäÿ■è£ĞæŕŤæŻť åŁăåĚĹè£ŻçŽĎ `format()` èęĄåůőäÿĂçĆźãĂĆ æŕŤåęĆïijŇåIJÍä¡£çŤÍ%æŞ■ä¡IJçňęæäijåijŔåŇŰæŢřå■ŮçŽĎæŮűåĂŹïijŇäÿ

3.4 äžŇåĚńå■ĄåĚ■è£ŻåĹűæŢť æŢř

éŮőécŸ

ä¡ăéIJĂèęĄè¡ňæ■ćæĹŰèĂĚè¿ŞåĞžä¡£çŤÍäžŇè£ŻåĹűïijŇåĚńè£ŻåĹűæĹŰå■ĄåĚ■è£ŻåĹűèąÍçď žçŽĎæŢ

èğčåĘşæŰźæąĹ

äÿžäžĘářĘæŢť æŢřè¡ňæ■ćäÿžäžŇè£ŻåĹűãĂĄåĚńè£ŻåĹűæĹŰå■ĄåĚ■è£ŻåĹűçŽĎæŰĞæIJňäÿšïijŇ åŔŕäžěåĹĘåĹńä¡£çŤÍ `bin()` , `oct()` æĹŰ `hex()` åĞ¡æŢřïijŽ

```
>>> x = 1234
>>> bin(x)
'0b10011010010'
>>> oct(x)
'0o2322'
>>> hex(x)
'0x4d2'
>>>
```

åŔęåď ŰïijŇåęĆæđIJä¡ääÿ■æČşè¿ŞåĞž `0b` , `0o` æĹŰèĂĚ `0x` çŽĎåL■çijĂçŽĎèŕÍïijŇåŔŕäžěä¡£çŤÍ `format()` åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> format(x, 'b')
'10011010010'
>>> format(x, 'o')
'2322'
>>> format(x, 'x')
'4d2'
>>>
```

æŢť æŢřæŸŕæIJL ç ňęåŔůçŽĎïijŇæL Ăäžěåę ĆæđIJä¡ăåIJĺåď ĎçŘĘèť §æŢřçŽĎèŕÍïijŇè¿ŞåĞžçżŞæđIJäij

```
>>> x = -1234
>>> format(x, 'b')
```

```
'-10011010010'
>>> format(x, 'x')
'-4d2'
>>>
```

åęĆæđIJä¡ăæČşäžğçŤ§äÿĂäÿłæŮăçňęåŔůåĂijïijŇä¡ăéIJĂèęĄåćđåŁääÿĂäÿłæŇĞçđ'žæIJĂåđ'ğä¡■éŢ£åž

```
>>> x = -1234
>>> format(2**32 + x, 'b')
'11111111111111111111011001011110'
>>> format(2**32 + x, 'x')
'fffffb2e'
>>>
```

äÿžäžĘäżěäÿ■åŘŇçŽĎè£ŻåĹűè¡ňæ■ćæŢt'æŢřå■ŮçňęäÿšïijŇçőĂå■ŢçŽĎä¡£çŤĺåÿęæIJL'è£ŻåĹűçŽĎ int() åĞ¡æŢřå■şåŔŕïijŽ

```
>>> int('4d2', 16)
1234
>>> int('10011010010', 2)
1234
>>>
```

èőÍèőž

åđ'ğåđ'ŽæŢřæČĚåĘţäÿŃåđ'ĎçŘĘäžŇè£ŻåĹűãĂĄåĚńè£ŻåĹűåŠŇå■ĄåĚ■è£ŻåĹűæŢt'æŢřæŸřå¿ĹçőĂå
åŘłèęĄèőřä¡Ŕè£ŹäžŻè¡ňæ■ćåśđäžŐæŢt'æŢřåŠŇåĚűåŕźåžŤçŽĎæŰĞæIJňèąÍçđ'žäźŇéŮt'çŽĎè¡ňæ■cå■şåŔŕã

æIJĂåŘŐïijŇä¡£çŤĺåĚńè£ŻåĹűçŽĎçÍŃåžŘåŚŸæIJL'äÿĂçĆźéIJĂèęĄæşÍæĎŔäÿŇãĂĆ
PythonæŇĞåőŽåĚńè£ŻåĹűæŢřçŽĎèŕ■æşŢèů§åĚűäžŰèŕ■èÍĂçÍ■æIJL'äÿ■åŘŇãĂĆæŕŤåęĆïijŇåęĆæđIJä¡ăåČ

```
>>> import os
>>> os.chmod('script.py', 0755)
    File "<stdin>", line 1
        os.chmod('script.py', 0755)
                              ^
SyntaxError: invalid token
>>>
```

éIJĂçąőä£ÍåĚńè£ŻåĹűæŢřçŽĎåL'■çijĂæŸŕ 0o ïijŇårśåČŔäÿŇéÍćè£Źæăůïijž

```
>>> os.chmod('script.py', 0o755)
>>>
```

3.5 å■ŮèŁĆåĹřåđ'ğæŢt'æŢřçŽĎæL'ŞåŇĚäÿŐèğčåŇĚ

éŮőécŸ

ä¡ăæIJL’äÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšáźűæČşåřĘåőČèğčåŐŃæĹŘäÿĂäÿłæŢt’æŢřãĂĆæĹŰèĂĚïijŇä¡ăéIJĂ

èğčåĘşæŰźæąĹ

åĄĞèő¿ä¡ăçŽĎçÍŃåžŔéIJĂèęĄåd’ĎçŘĘäÿĂäÿłæŃěæIJL’128ä¡■éŢ£çŽĎ16äÿłåĚČçt’ăçŽĎå■ŮèŁĆå■Ůç

```
data = b'\x00\x124V\x00x\x90\xab\x00\xcd\xef\x01\x00#\x004'
```

äÿžäžĘåřĘbytesèğčæđŘäÿžæŢt’æŢřïijŇä¡£çŤÍ `int.from_bytes()` æŰźæşŢïijŇåźűåČŔäÿŃéÍćè£Źæăůæ ŇĞåőŽå■ŮèŁĆéąžåžŔïijŽ

```
>>> len(data)
16
>>> int.from_bytes(data, 'little')
69120565665751139577663547927094891008
>>> int.from_bytes(data, 'big')
94522842520747284487117727783387188
>>>
```

äÿžäžĘåřĘäÿĂäÿłåd’ğæŢt’æŢřè¡ñæ■ćäÿžäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšïijŇä¡£çŤÍ `int.to_bytes()` æŰźæşŢïijŇåźűåČŔäÿŃéÍćè£ŹæăůæŇĞåőŽå■ŮèŁĆæŢřåŠŇå■ŮèŁĆéąžåžŔïijŽ

```
>>> x = 94522842520747284487117727783387188
>>> x.to_bytes(16, 'big')
b'\x00\x124V\x00x\x90\xab\x00\xcd\xef\x01\x00#\x004'
>>> x.to_bytes(16, 'little')
b'4\x00#\x00\x01\xef\xcd\x00\xab\x90x\x00V4\x12\x00'
>>>
```

èőÍèőž

åd’ğæŢt’æŢřåŠŇå■ŮèŁĆå■Ůçňęäÿšäź ŃéŮt’çŽĎè¡ñæ■ćæŞ■ä¡IJåźűäÿ■åÿÿèğĄãĂĆ çĎűèĂŇïijŇåIJÍäÿĂäžŻåžŤçŤÍéćĘå§§æIJL’æŮűåĂŹäź§äijŽåĞžçŐřïijŇæŕŤåęĆåŕĘçăĄå■ęæĹŰèĂĚç¡ŚçżIJã ä¿ŃåęĆïijŇIPv6ç¡ŚçżIJåIJřåİĂä¡£çŤÍäÿĂäÿł128ä¡■çŽĎæŢt’æŢřèąÍçd’žãĂĆ åęĆæđIJä¡ăèęĄäžŐäÿĂäÿłæŢřæ■őèőřå¡Ţäÿ■æŔŘåŔŰè£ŹæăůçŽĎåĂijçŽĎæŮűåĂŹïijŇä¡ăåřśäijŽéÍćåŕźè£Ź

ä¡IJäÿžäÿĂçğ■æŻ£äžčæŰźæąĹïijŇä¡ăåŔŕèČ¡æČşä¡£çŤÍ6.11åřŔèŁĆäÿ■æL’ĂäżŃçż■çŽĎ `struct` æÍąåİŮæİěèğčåŐŃå■ŮèŁĆãĂĆ è£ŹæăůäźšèąŇå¿ŮéĂŽïijŇäÿ■è£ĞåĹl’çŤÍ `struct` æÍąåİŮæİěèğčåŐŃåŕźäžŐæŢt’æŢřçŽĎåd’ğåŕÆŸŕæIJL’éŹŘåĹűçŽĎãĂĆ åŽăæ■d’ïijŇä¡ăåŔŕèČ¡æČşèğčåŐŃåd’Žäÿłå■ŮèŁĆäÿšáźűåřĘçżŞædIJåŘĹáźűäÿžæIJĂçżĹçŽĎçżŞædIJïijŇårś

```
>>> data
b'\x00\x124V\x00x\x90\xab\x00\xcd\xef\x01\x00#\x004'
>>> import struct
>>> hi, lo = struct.unpack('>QQ', data)
>>> (hi << 64) + lo
```

```
94522842520747284487117727783387188
>>>
```

å■ÛèŁĆéąžåžŔèğĎåĹŹ(littleæĹŰbig)äżĚäżĚæŇĞåőŽäžĘæđĎåżžæŢťæŢřæŮűçŽĎå■ÛèŁĆçŽĎä¡Őä¡■æĹŚäžñäżŐäÿŃéİćçš¿å£ČæđĎéĂäçŽĎ16è£ŻåĹűæŢřçŽĎèąĺçďžäÿ■åŔřäžěå¿Ĺåőźæ ŸŞçŽĎçIJŃåĞžæİěïijŽ

```
>>> x = 0x01020304
>>> x.to_bytes(4, 'big')
b'\x01\x02\x03\x04'
>>> x.to_bytes(4, 'little')
b'\x04\x03\x02\x01'
>>>
```

åęĆæđIJä¡ăèŕŢçİĂåřĘäÿĂäÿłæŢťæŢřæĽŞåŇĚäÿžå■ÛèŁĆå■ÛçňęäÿšïijŇéĆčäźĹåőČåřśäÿ■åŘĹéĂĆäžĚ åęĆæđIJéIJĂèęAçŽĎèŕÏïijŇä¡ăåŔřäžěä¡£çŤÍ `int.bit_length()` æŰźæşŢæİěåĘşåőŽéIJĂèęAåď˙ŽåřŚå■ÛèŁĆä¡■æİěå■ŸåĆĺè£Źäÿłå Äijã ĂĆ

```
>>> x = 523 ** 23
>>> x
335381300113661875107536852714019056160355655333978849017944067
>>> x.to_bytes(16, 'little')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
OverflowError: int too big to convert
>>> x.bit_length()
208
>>> nbytes, rem = divmod(x.bit_length(), 8)
>>> if rem:
... nbytes += 1
...
>>>
>>> x.to_bytes(nbytes, 'little')
b'\x03X\xf1\x82iT\x96\xac\xc7c\x16\xf3\xb9\xcf...\xd0'
>>>
```

3.6 åď■æŢřçŽĎæŢřå■ęè£ŘçőÛ

éŮőécŸ

ä¡ăåĘŹçŽĎæIJĂæŰřçŽĎç¡ŚçżIJèőď˙èŕĄæŰźæąĹäžčçăĄéĄĞåĹřäžĘäÿĂäÿłéŽ¿éćŸïijŇåźűäÿŤä¡ăåŤřäÿ åĘ■æĹŰèĂĚæŸŕä¡ăäżĚäżĚéIJĂèęAä¡£çŤĺåď■æŢřæİěæĽğèąŇäÿĂäžŻèőąçőÛæŞ■ä¡IJãĂĆ

èğčåĘşæŰźæąĹ

åď■æŢřåŔřäžěçŤĺä¡£çŤĺåĞ¡æŢř `complex(real, imag)` æĹŰèĂĚæŸŕåÿęæIJĽåŘŐçijĂjçŽĎæţőçĆźæŢřæİěæŇĞåőŽãĂĆæŕŤåęĆïijŽ

```
>>> a = complex(2, 4)
>>> b = 3 - 5j
>>> a
(2+4j)
>>> b
(3-5j)
>>>
```

åŕźåžŤçŽĎåőđéČĺãĂĄèŹŽéČĺåŠŇåĚśè¡■åď■æŢřåŔŕäžěå¿Ĺåőźæ˙ŞçŽĎèŐůåŔŰãĂĆåřśåČŔäÿŃéÍćè£Ź

```
>>> a.real
2.0
>>> a.imag
4.0
>>> a.conjugate()
(2-4j)
>>>
```

åŔęåď˙ŰïijŇæĽ˙ĂæIJĽ˙åÿÿèğĄçŽĎæŢřå■ęè£ŘçőŮéČ¡åŔŕäžěåůěä¡IJïijŽ

```
>>> a + b
(5-1j)
>>> a * b
(26+2j)
>>> a / b
(-0.4117647058823529+0.6470588235294118j)
>>> abs(a)
4.47213595499958
>>>
```

åęĆæđIJèęĄæĽ˙ğèąŇåĚűäžŰçŽĎåď■æŢřåĞ¡æŢřæŕŤåęĆæ■čåijęãĂĄä¡ŹåijęǽĹŰåźşæŰźæăźïijŇä¡£çŤĺ `cmath` æĺąåİŮïijŽ

```
>>> import cmath
>>> cmath.sin(a)
(24.83130584894638-11.356612711218174j)
>>> cmath.cos(a)
(-11.36423470640106-24.8146514856341 87j)
>>> cmath.exp(a)
(-4.829809383269385-5.5920560936409816j)
>>>
```

èőÍèőž

Pythonäÿ■åď ğéČĺåĹĘäÿŐæŢřå■ęçŻÿåĚşçŽĎæĺąåİŮéČ¡èČ¡åď˙ĎçŘĘåď■æŢřãĂĆ æŕŤåęĆåęĆæđIJä¡ăä¡£çŤĺ `numpy` ïijŇåŔŕäžěå¿Ĺåőźæ˙ŞçŽĎæđĎéĂääÿĂäÿłåď■æŢřæŢřçżĎåźűåIJÍè£Źäÿłæ

```
>>> import numpy as np
>>> a = np.array([2+3j, 4+5j, 6-7j, 8+9j])
```

```
>>> a
array([ 2.+3.j,  4.+5.j,  6.-7.j,  8.+9.j])
>>> a + 2
array([ 4.+3.j,  6.+5.j,  8.-7.j, 10.+9.j])
>>> np.sin(a)
array([ 9.15449915 -4.16890696j, -56.16227422 -48.50245524j,
        -153.20827755-526.47684926j, 4008.42651446-589.49948373j])
>>>
```

PythonçŽĎæăĞåĞĘæŢřå■ęåĞ¡æŢřçąőåőđæČĚåĘţäÿŃåźűäÿ■èČ¡äžğçŤ§åd'■æŢřåĂijïijŇåŻäæ■d'ä¡ăçŽĺ

```
>>> import math
>>> math.sqrt(-1)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ValueError: math domain error
>>>
```

åęĆæđIJä¡ăæČşçŤ§æĹŘäÿĂäÿłåd'■æŢřè£ŤåŻđçżŞæđIJïijŇä¡ăå£Ěéążæ Ÿ¿çd'žçŽĎä¡£çŤÍ
`cmath` æÍąåİŮïijŇæĹŰèĂĚåIJÍæ§Řäÿłæ ŤŕæŇĄåd'■æŢřçŽĎåžŞäÿ■åčřæ ŸŐåd'■æŢřçśžåđŃçŽĎä¡£çŤÍãĂĆæ

```
>>> import cmath
>>> cmath.sqrt(-1)
1j
>>>
```

3.7 æŮăçľůåd'ğäÿŐNaN

éŮőécŸ

ä¡ăæČşåĹŻåžžæĹŰæţŃèŕŢæ■čæŮăçľůãĂĄèt'§æŮăçľůæĹŰNaN(éİđæŢřå■Ů)çŽĎæţőçĆźæŢřãĂĆ

èğčåĘşæŰźæąĹ

PythonåźűæšąæIJL'çL'źæőŁçŽĎèŕ■æşŢæİěèąÍçd'žè£ŹäžŻçL'źæőŁçŽĎæţőçĆźåĂijïijŇä¡ĘæŸŕåŔŕäžěä¡£
`float()` æİěåĹŻåžžåőČäžñãĂĆæŕŤåęĆïijŽ

```
>>> a = float('inf')
>>> b = float('-inf')
>>> c = float('nan')
>>> a
inf
>>> b
-inf
>>> c
nan
>>>
```

äÿžäžĘæţŃèŕŢè£ŹäžŻåĂijçŽĎå■ŸåIJĺïijŇä¡£çŤĺ `math.isinf()` åŠŇ `math.isnan()` åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> math.isinf(a)
True
>>> math.isnan(c)
True
>>>
```

èőĺèőž

æČşäžĘèğčæŻt'åd'Žè£ŹäžŻçL'źæőŁæţőçĆźåĂijçŽĎä£ąæĄŕïijŇåŔŕäžěåŔĆèĂČIEEE 754èğĎèŇČãĂĆ çĎűèĂŇïijŇäź§æIJL'äÿĂäžŻåIJřæŰźéIJĂèęĄä¡ăçL'źåĹńæşĺæĎŔïijŇçL'źåĹńæŸŕèů§æŕŤè¿

æŮăçl'ůåd'ğæŢřåIJĺæL'ğèąŇæŢřå■ęèőąçőŮçŽĎæŮűåĂŹäijŽäijăæŠ■ïijŇæŕŤåęĆïijŽ

```
>>> a = float('inf')
>>> a + 45
inf
>>> a * 10
inf
>>> 10 / a
0.0
>>>
```

ä¡ĘæŸŕæIJL'äžŻæŞ■ä¡IJæŮűæIJłåőŽäźL'çŽĎåźűäijŽè£ŤåŻđäÿĂäÿłNaNçżŞæđIJãĂĆæŕŤåęĆïijŽ

```
>>> a = float('inf')
>>> a/a
nan
>>> b = float('-inf')
>>> a + b
nan
>>>
```

NaNåĂijäijŽåIJĺæL'ĂæIJL'æŞ■ä¡IJäÿ■äijăæŠ■ïijŇèĂŇäÿ■äijŽäžğçŤ§åijĆåÿÿãĂĆæŕŤåęĆïijŽ

```
>>> c = float('nan')
>>> c + 23
nan
>>> c / 2
nan
>>> c * 2
nan
>>> math.sqrt(c)
nan
>>>
```

NaNåĂijçŽĎäÿĂäÿłçL'źåĹńçŽĎåIJřæŰźæŮűåőČäžňäźŃéŮt'çŽĎæŕŤè¿ČæŞ■ä¡IJæĂżæŸŕè£ŤåŻđFalseã

```
>>> c = float('nan')
>>> d = float('nan')
>>> c == d
False
>>> c is d
False
>>>
```

çŤśäžŐè£ŹäÿłåŐ§åŻäïijŇæţŃèŕŢäÿĂäÿłNaNåĂijå¿ŮåŤŕäÿĂåőĽåĚĺçŽĎæŰźæşŢåřśæŸŕä¡£çŤĺ `math.isnan()` ïijŇäź§åřśæŸŕäÿŁéİćæijŤçd'žçŽĎéĆčæăůãĂĆ

æIJL'æŮűåĂŹçÍŃåžŔåŚŸæČşæŤźåŔŸPythonézŸèőd'èąŇäÿžïijŇåIJÍè£ŤåŻđæŮăçl'ůåd'ğæĹŰNaNçżŞæ `fpectl` æÍąåİŮåŔŕäžěçŤĺæİěæŤźåŔŸè£Źçğ■èąŇäÿžïijŇä¡ĘæŸŕåőČåIJÍæăĞåĞĘçŽĎPythonædĎåžžäÿ■åźűåźűäÿŤéŠĹåŕźçŽĎæŸŕäÿŞåőűçžğçÍŃåžŔåŚŸãĂĆåŔŕäžěåŔĆèĂČåIJÍçž£çŽĎPythonæŰĞæąčèŐůåŔŰæŻt'åd

3.8 åĹĘæŢřè£ŘçőŮ

éŮőécŸ

ä¡ăè£ŻåĚěæŮűéŮt'æIJžåŹĺïijŇçłĄçĎűåŔŚçŐřä¡ăæ■čåIJÍåĄŽåřŔå■ęåőűåž■ä¡IJäÿŽïijŇåźűæűL'åŔŁåĹřåæĹŰèĂĚä¡ăåŔŕèČ¡éIJĂèęĄåĘŹäžčçăĄåŐżèőąçőŮåIJÍä¡ăçŽĎæIJÍåůěåůěåŐĆäÿ■çŽĎæţŃéĞŔåĂijãĂĆ

èğčåĘşæŰźæąĹ

`fractions` æÍąåİŮåŔŕäžěèćńçŤĺæİěæL'ğèąŇåŇĚåŘńåĹĘæŢřçŽĎæŢřå■ęè£ŘçőŮãĂĆæŕŤåęĆïijŽ

```
>>> from fractions import Fraction
>>> a = Fraction(5, 4)
>>> b = Fraction(7, 16)
>>> print(a + b)
27/16
>>> print(a * b)
35/64

>>> # Getting numerator/denominator
>>> c = a * b
>>> c.numerator
35
>>> c.denominator
64

>>> # Converting to a float
>>> float(c)
0.546875

>>> # Limiting the denominator of a value
>>> print(c.limit_denominator(8))
4/7
```

```
>>> # Converting a float to a fraction
>>> x = 3.75
>>> y = Fraction(*x.as_integer_ratio())
>>> y
Fraction(15, 4)
>>>
```

èőÍèőž

åIJÍåď ğåď ŽæŢřçÍŃåžŔäÿ■äÿĂèĹňäÿ■äijŽåĞžçŐřåĹĘæŢřçŽĎèőąçőŮéŮőéćŸïijŇä¡ĘæŸŕæIJL’æŮűåĂŽ
æŕŤåęĆïijŇåIJÍäÿĂäÿłåĚĄèőÿæŐěåŔŮåĹĘæŢřå¡ćåijŔçŽĎæţŃèŕŢå■Ţä¡■åźűäżěåĹĘæŢřå¡ćåijŔæL’ğèąŇè£Ř
çŻť’æŐěä¡£çŤĺåĹĘæŢřåŔŕäżěåĞŔåřŚæL’ŃåŁÍè¡ňæ■ćäÿžåřŔæŢřæĹŰæţőçĆźæŢřçŽĎåůěä¡IJãĂĆ

3.9 åď ğåđŃæŢřçżĎè£ŘçőŮ

éŮőéćŸ

ä¡ăéIJĂèęĄåIJÍåď ğæŢřæ■őéŻĘ(æŕŤåęĆæŢřçżĎæĹŰç¡Śæăij)äÿŁéİćæL’ğèąŇèőąçőŮãĂĆ

èğčåĘşæŰźæąĹ

æűL’åŔŁåĹřæŢřçżĎçŽĎéĞ■éĞŔçžğè£ŘçőŮæŞ■ä¡IJïijŇåŔŕäżěä¡£çŤÍ `NumPy` åžŞãĂĆ
`NumPy` çŽĎäÿĂäÿłäÿżèęĄçL’źå¿ĄæŸŕåőČäijŽçżŹPythonæŔŘä¿ŻäÿĂäÿłæŢřçżĎåŕźèśąïijŇçŻÿæŕŤæăĞåĞĘ
äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎåřŔä¿Ńå■ŘïijŇåŘŚä¡ăåśŢçď žæăĞåĞĘåĹŮèąĺåŕźèśąåŠŇ
`NumPy` æŢřçżĎåŕźèśąäźŃéŮť’çŽĎåůőåĹńïijŽ

```
>>> # Python lists
>>> x = [1, 2, 3, 4]
>>> y = [5, 6, 7, 8]
>>> x * 2
[1, 2, 3, 4, 1, 2, 3, 4]
>>> x + 10
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: can only concatenate list (not "int") to list
>>> x + y
[1, 2, 3, 4, 5, 6, 7, 8]

>>> # Numpy arrays
>>> import numpy as np
>>> ax = np.array([1, 2, 3, 4])
>>> ay = np.array([5, 6, 7, 8])
>>> ax * 2
array([2, 4, 6, 8])
>>> ax + 10
```

```
array([11, 12, 13, 14])
>>> ax + ay
array([ 6, 8, 10, 12])
>>> ax * ay
array([ 5, 12, 21, 32])
>>>
```

æ■čåęĆæĽ Ăè ğĄïijŇäÿd' çğ■æŰźæąĹäÿ■æŢřçżĎçŽĎå§žæIJňæŢřå■ęè£ŘçőŮçżŞædIJåźűäÿ■çŻÿåŘŇãĂ
çĽźåĹńçŽĎïijŇ NumPy äÿ■çŽĎæăĞéĞŔè£ŘçőŮ(æŕŤåęĆ ax
* 2 æĹŰ ax + 10)äijŽä¡IJçŤĺåIJĺæŕŔäÿĂäÿłåĚČçt' ääÿŁãĂĆ
åŔęåd' ŰïijŇå¡Şäÿd' äÿłæŞ■ä¡IJæŢřéČ¡æŸŕæŢřçżĎçŽĎæŮűåĂŹæĽğèąŇåĚČçt' ääŕźç■Ľä¡■ç¡őëőąçőŮïijŇåźű

åŕźæŢt' äÿłæŢřçżĎäÿ■æĽĂæIJĽåĚČçt' ääŘŇæŮűæĽğèąŇæŢřå■ęè£ŘçőŮåŔŕäżěä¡£å¿Ůä¡IJçŤĺåIJĺæŢt' ä
æŕŤåęĆïijŇåęĆæđIJä¡ăæČşèőąçőŮåd' ŽéąźåijŕçŽĎåĂijïijŇåŔŕäżěè£ŹæăůåĄŽïijŽ

```
>>> def f(x):
... return 3*x**2 - 2*x + 7
...
>>> f(ax)
array([ 8, 15, 28, 47])
>>>
```

NumPy è£ŸäÿžæŢřçżĎæŞ■ä¡IJæŔŘä¿ŻäžĘåd' ğéĞŕçŽĎéĂŽçŤĺåĞ¡æŢřïijŇè£ŹäžŻåĞ¡æŢřåŔŕäżěä¡IJäÿ
math æĺąåİŮäÿ■çśžäijijåĞ¡æŢřçŽĎæŻ£äżčãĂĆæŕŤåęĆïijŽ

```
>>> np.sqrt(ax)
array([ 1. , 1.41421356, 1.73205081, 2. ])
>>> np.cos(ax)
array([ 0.54030231, -0.41614684, -0.9899925 , -0.65364362])
>>>
```

ä¡£çŤĺè£ŹäžŻéĂŽçŤĺåĞ¡æŢřèęĄæŕŤå¿łçŐŕæŢřçżĎåźűä¡£çŤĺ
math æĺąåİŮäÿ■çŽĎåĞ¡æŢřæĽğèąŇèőąçőŮèęĄå£ńçŽĎåd' ŽãĂĆ
åŻăæ■d' ïijŇåŔłèęĄæIJĽåŔŕèČ¡çŽĎèŕĺår¡éĞŔéĂĽæŃl' NumPy çŽĎæŢřçżĎæŰźæąĹãĂĆ

åžŢåśĆåőđçŐŕäÿ■ïijŇ NumPy æŢřçżĎä¡£çŤĺäžĘCæĹŰèĂĚFortranèŕ■èĺĂçŽĎæIJžåĹűåĹĘéĚ■åĘĚå■Ÿã
äź§åřśæŸŕèŕt' ïijŇåőČäžňæŸŕäÿĂäÿłéİđåÿÿåd' ğçŽĎè£đçż■çŽĎåźűçŤśåŘŇçśžådŃæŢřæ■őçżĎæĹŘçŽĎåĘĚå
æĽĂäżěïijŇä¡ăåŔŕäżěædĎéĂääÿĂäÿłæŕŤæŹőéĂŽPythonåĹŮèąĺåd' ğçŽĎåd' ŽçŽĎæŢřçżĎãĂĆ
æŕŤåęĆïijŇåęĆæđIJä¡ăæČşædĎéĂääÿĂäÿł10,000*10,000çŽĎætőçĆźæŢřäžŇçżt' ç¡ŚæäijïijŇå¿Łè¡żæİ¿ïijŽ

```
>>> grid = np.zeros(shape=(10000,10000), dtype=float)
>>> grid
    array([[ 0., 0., 0., ..., 0., 0., 0.],
    [ 0., 0., 0., ..., 0., 0., 0.],
    [ 0., 0., 0., ..., 0., 0., 0.],
    ...,
    [ 0., 0., 0., ..., 0., 0., 0.],
    [ 0., 0., 0., ..., 0., 0., 0.],
    [ 0., 0., 0., ..., 0., 0., 0.]])
>>>
```

æL’ĂæIJL’çŽĎæŹőéĂŽæŞ■ä¡IJè£ŸæŸŕäijŽåŘŇæŮűä¡IJçŤĺåIJĺæL’ĂæIJL’åĚČçť’ääýŁïijŽ

```
>>> grid += 10
>>> grid
array([[ 10., 10., 10., ..., 10., 10., 10.],
    [ 10., 10., 10., ..., 10., 10., 10.],
    [ 10., 10., 10., ..., 10., 10., 10.],
    ...,
    [ 10., 10., 10., ..., 10., 10., 10.],
    [ 10., 10., 10., ..., 10., 10., 10.],
    [ 10., 10., 10., ..., 10., 10., 10.]])
>>> np.sin(grid)
array([[-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    ...,
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
    [-0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111]])
>>>
```

åĚşäžŐ NumPy æIJL’äýĂçĆźéIJĂèęĄçL’źåĹńçŽĎäýżæĎŔïijŇéĆčåřśæŸŕåőČæL’l’åśŢPythonåĹŮèąĺçŽĎ - çL’źåĹńæŸŕåŕźäžŐåď’Žçžť’æŢřçżĎãĂĆ äýžäžĘèŕť’æŸŐæýĚæěŽïijŇåĚĹæđĎéĂääýĂäýłçőĂå■ŢçŽĎäžŇçżt

```
>>> a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
>>> a
array([[ 1, 2, 3, 4],
[ 5, 6, 7, 8],
[ 9, 10, 11, 12]])

>>> # Select row 1
>>> a[1]
array([5, 6, 7, 8])

>>> # Select column 1
>>> a[:,1]
array([ 2, 6, 10])

>>> # Select a subregion and change it
>>> a[1:3, 1:3]
array([[ 6, 7],
        [10, 11]])
>>> a[1:3, 1:3] += 10
>>> a
array([[ 1, 2, 3, 4],
```

```
        [ 5, 16, 17, 8],
        [ 9, 20, 21, 12]])

>>> # Broadcast a row vector across an operation on all rows
>>> a + [100, 101, 102, 103]
array([[101, 103, 105, 107],
        [105, 117, 119, 111],
        [109, 121, 123, 115]])
>>> a
array([[ 1, 2, 3, 4],
        [ 5, 16, 17, 8],
        [ 9, 20, 21, 12]])

>>> # Conditional assignment on an array
>>> np.where(a < 10, a, 10)
array([[ 1, 2, 3, 4],
        [ 5, 10, 10, 8],
        [ 9, 10, 10, 10]])
>>>
```

èőÍèőž

NumPy æŸŕPythonéćĘå§§äÿ■å¿Ĺåď'ŽçğŚå■ęäÿŐåůęçÍŃåžŞçŽĎå§žçąĂïijŇåŘŇæŮűäź§æŸŕèćńåź£æşŻ
å■şä¿£åęĆæ■d'ïijŇåIJĺåĹŽåijÅåğŃçŽĎæŮűåĂŹéĂŽè£ĞäÿĂäžŽçőÅå■ŢçŽĎä¿Ńå■ŘåŠŃçŐl'åĚůçÍŃåžŔäź§

éĂŽåÿÿæĹŚäžňåŕijåĚě NumPy æĺąåİŮçŽĎæŮűåĂŹäijŽä¡£çŤĺéŕ■åŘě import numpy as np ãĂĆ è£ŹæăůçŽĎèŕİä¡ăåŕšäÿ■çŤĺåĘ■ä¡ăçŽĎçÍŃåžŔéĞŃéİćäÿĂéĄ■éĄ■çŽĎæŢšåĚě numpy ïijŇåŘłéIJĂèęĄè¿ŞåĚě np åŕšèąŇäžĘïijŇèŁĆçIJĄäžĘäÿ■åŕŚæŮűéŮť'ãĂĆ

åęĆæđIJæČşèŐůåŔŰæŻť'åď'ŽçŽĎä£ąæĄŕïijŇä¡ăå¡ŞçĎűå¿ŮåŐż NumPy åőŸç¡ŚéĂŽéĂŽäžĘïijŇç¡ŚåİĂæŸŕïijŽ http://www.numpy.org

3.10 ç§l'éŸţäÿŐçž£æĂğäżčæŢřè£ŘçőŮ

éŮőécŸ

ä¡ăéIJĂèęĄæL'ğèąŇç§l'éŸţåŠŇçž£æĂğäżčæŢřè£ŘçőŮïijŇæŕŤåęĆç§l'éŸţäźŸæşŢãĂĄåŕżæL'¿èąŇåĹŮå

èğčåĘşæŰźæąĹ

NumPy åžŞæIJL'äÿĂäÿłç§l'éŸţåŕźèśąåŘŕäżěçŤĺæİěèğčåĘşè£ŹäÿłéŮőécŸãĂĆ

ç§l'éŸţçśżäijijäžŐ3.9åŕŘèŁĆäÿ■æŢřçżĎåŕźèśąïijŇä¡ĘæŸŕéĄţå¿łçž£æĂğäżčæŢřçŽĎèőąçőŮèğĎåĹŹãĂ

```
>>> import numpy as np
>>> m = np.matrix([[1,-2,3],[0,4,5],[7,8,-9]])
>>> m
```

```
matrix([[ 1, -2, 3],
        [ 0, 4, 5],
        [ 7, 8, -9]])

>>> # Return transpose
>>> m.T
matrix([[ 1, 0, 7],
        [-2, 4, 8],
        [ 3, 5, -9]])

>>> # Return inverse
>>> m.I
matrix([[ 0.33043478, -0.02608696, 0.09565217],
        [-0.15217391, 0.13043478, 0.02173913],
        [ 0.12173913, 0.09565217, -0.0173913 ]])

>>> # Create a vector and multiply
>>> v = np.matrix([[2],[3],[4]])
>>> v
matrix([[2],
        [3],
        [4]])
>>> m * v
matrix([[ 8],
        [32],
        [ 2]])
>>>
```

åŘŕäžěåIJÍ `numpy.linalg` å■ŘåŇĚäÿ■æĽ¿åĹřæŻť'åď'ŽçŽĎæŞ■ä¡IJåĞ¡æŢřiijŇæŕŤåęĆïijŽ

```
>>> import numpy.linalg

>>> # Determinant
>>> numpy.linalg.det(m)
-229.99999999999983

>>> # Eigenvalues
>>> numpy.linalg.eigvals(m)
array([-13.11474312, 2.75956154, 6.35518158])

>>> # Solve for x in mx = v
>>> x = numpy.linalg.solve(m, v)
>>> x
matrix([[ 0.96521739],
        [ 0.17391304],
        [ 0.46086957]])
>>> m * x
matrix([[ 2.],
        [ 3.],
        [ 4.]])
```

```
>>> v
matrix([[2],
        [3],
        [4]])
>>>
```

èőÍèőž

å¿ÍæŸ¿çĎűçž£æĂğäžčæŢřæŸŕäÿłéİđåÿÿåď˘ğçŽĎäÿžécŸïijŇåůšçżŔèűĚåĞžäžĘæIJňäźęèČ¡èőÍèőžçŽĎè
ä¡ĘæŸŕïijŇåęĆæđIJä¡ăéIJĂèęĄæŞ■ä¡IJæŢřçżĎåŠŇåŘŚéĞŔçŽĎèŕÍïijŇ NumPy
æŸŕäÿĂäÿłäÿ■éŤŹçŽĎåĚěåŔčçĆźãĂĆ åŘŕäžěèő£éŮő NumPy åőŸç¡Ś http://www.numpy.org
èŐůåŘŰæŻť˘åď˘Žä£ąæĄŕãĂĆ

3.11 éŽŔæIJžéĂĽ˘æŃĺ˘

éŮőécŸ

ä¡ăæČşäżŐäÿĂäÿłåžŔåĹŮäÿ■éŽŔæIJžæŁ¡åŘŰèŃěåźšåĚČçť˘ăïijŇæĹŰèĂĚæČşçŢ§æĹŘåĞääÿłéŽŔæIJ

èğčåĘşæŰźæąĹ

random æĺąåİŮæIJL˘åď˘ğéĞŔçŽĎåĞ¡æŢřçŤĺæİěäžğçŤ§éŽŔæIJžæŢřåŠŇéŽŔæIJžéĂĽ˘æŃĺ˘åĚČçť˘ăăĂĆ
æŕŤåęĆïijŇèęĄæČşäżŐäÿĂäÿłåžŔåĹŮäÿ■éŽŔæIJžçŽĎæŁ¡åŘŰäÿĂäÿłåĚČçť˘ăïijŇåŔŕäžěä¡£çŤÍ
random.choice() ïijŽ

```
>>> import random
>>> values = [1, 2, 3, 4, 5, 6]
>>> random.choice(values)
2
>>> random.choice(values)
3
>>> random.choice(values)
1
>>> random.choice(values)
4
>>> random.choice(values)
6
>>>
```

äÿžäžĘæŔŘåŘŰåĞžNäÿłäÿ■åŘŇåĚČçť˘ăçŽĎæăůæIJňçŤĺæİěåĄŽè£ŻäÿĂæ■ěçŽĎæŞ■ä¡IJïijŇåŔŕäžěä¡£
random.sample() ïijŽ

```
>>> random.sample(values, 2)
[6, 2]
>>> random.sample(values, 2)
[4, 3]
```

```
>>> random.sample(values, 3)
[4, 3, 1]
>>> random.sample(values, 3)
[5, 4, 1]
>>>
```

åęĆædIJä¡ăäżĔäżĔåŔłæŸŕæČşæĽŞäźśåžŔåĹŮäÿ■åĔČçť ăçŽĎéąžåžŔïijŇåŔŕäżěä¡£çŤÍ `random.shuffle()` ïijŽ

```
>>> random.shuffle(values)
>>> values
[2, 4, 6, 5, 3, 1]
>>> random.shuffle(values)
>>> values
[3, 5, 2, 1, 6, 4]
>>>
```

çŤ§æĹŘéŽŔæIJžæŢť æŢřïijŇèŕůä¡£çŤÍ `random.randint()` ïijŽ

```
>>> random.randint(0,10)
2
>>> random.randint(0,10)
5
>>> random.randint(0,10)
0
>>> random.randint(0,10)
7
>>> random.randint(0,10)
10
>>> random.randint(0,10)
3
>>>
```

äÿžäžĘçŤ§æĹŘ0åĹř1èŇČåŻť åĘĔåİĞåŇĂåĹĘåÿČçŽĎæţőçĆźæŢřïijŇä¡£çŤÍ `random.random()` ïijŽ

```
>>> random.random()
0.9406677561675867
>>> random.random()
0.133129581343897
>>> random.random()
0.4144991136919316
>>>
```

åęĆædIJèęĄèŐůåŔŰNä¡■éŽŔæIJžä¡■(äžŇè£ŻåĹű)çŽĎæŢť æŢřïijŇä¡£çŤÍ `random.getrandbits()` ïijŽ

```
>>> random.getrandbits(200)
335837000776573622800628485064121869519521710558559406913275
>>>
```

èőÍèőž

random æÍąåİŮä¡£çŤÍ *Mersenne Twister* çőŮæşŢæİèèőąçőŮçŤ§æĹŘéŽŔæIJžæŢřãĂĆè£ŹæŸŕäÿĂäÿłçą
ä¡ĘæŸŕä¡ăåŔŕäżěéĂŽè£Ğ random.seed() åĞ¡æŢřä£őæŤźåĹİåğŃåŇŰçğ■å■ŘãĂĆæŕŤåęĆïijŽ

```
random.seed() # Seed based on system time or os.urandom()
random.seed(12345) # Seed based on integer given
random.seed(b'bytedata') # Seed based on byte data
```

éŹď'äžĘäÿŁè£řäżŃçż■çŽĎåŁ§èČ¡ïijŇrandomæÍąåİŮè£ŸåŇĚåŘńå§žäžŐåİĞåŇĂåĹĘåÿČãĂĄéńŸæŰŕå
æŕŤåęĆïijŇ random.uniform() èőąçőŮåİĞåŇĂåĹĘåÿČéŽŔæIJžæŢřïijŇ
random.gauss() èőąçőŮæ■čæĂĄåĹĘåÿČéŽŔæIJžæŢřãĂĆ
åŕźäžŐåĚűäżŰçŽĎåĹĘåÿČæČĚåĘţèŕůåŔĆèĂČãIJÍçž£æŰĞæąčãĂĆ

åIJÍ random æÍąåİŮäÿ■çŽĎåĞ¡æŢřäÿ■åžŤèŕěçŤÍåIJÍåŠŇårĘçăĄå■ęçŻÿåĚşçŽĎçÍŃåžŔäÿ■ãĂĆ
åęĆæđIJä¡ăçąőåődéIJĂèęĄçśżäijijçŽĎåŁ§èČ¡ïijŇåŔŕäżěä¡£çŤÍsslæÍąåİŮäÿ■çŻÿåžŤçŽĎåĞ¡æŢřãĂĆ
æŕŤåęĆïijŇ ssl.RAND_bytes() åŔŕäżěçŤÍæÌěçŤ§æĹŘäÿĂäÿłåőĽåĚÍçŽĎéŽŔæIJžå■ŮèŁĆåžŔåĹŮãĂĆ

3.12 å§žælJńçŽĎæŮěælJ§äÿŐæŮűéŮť è¡ňæ■ć

éŮőécŸ

ä¡ăéIJĂèęĄæL'ğèąŇçőĂå■ŢçŽĎæŮűéŮť è¡ňæ■ćïijŇærŤåęĆåd'l'åĽřçğŠïijŇářŔæŮűåĽřåĹĘéŠ§ç■L'çŽĎ

èğčåĘşæŰźæąĹ

äÿžäžĘæL'ğèąŇäÿ■åŘŇæŮűéŮť'å■Ţä¡■çŽĎè¡ňæ■ćåŠŇèőąçőŮïijŇèřůä¡£çŤÍ
datetime æÍąåİŮãĂĆ æŕŤåęĆïijŇäÿžäžĘèąĹçd'žäÿĂäÿłæŮűéŮť'æőţïijŇåŔŕäżěåĹŻåžžäÿĂäÿł
timedelta åőđä¿ŃïijŇářšåČŔäÿŃéIćè£ŹæăůïijŽ

```
>>> from datetime import timedelta
>>> a = timedelta(days=2, hours=6)
>>> b = timedelta(hours=4.5)
>>> c = a + b
>>> c.days
2
>>> c.seconds
37800
>>> c.seconds / 3600
10.5
>>> c.total_seconds() / 3600
58.5
>>>
```

åęĆæđIJä¡ăæČşèąĹçd'žæŇĞåőŽçŽĎæŮěæIJ§åŠŇæŮűéŮť'ïijŇåĚĹåĹŻåžžäÿĂäÿł
datetime åőđä¿ŃçĎűåŘŐä¡£çŤÍæăĞåĞĘçŽĎæŢřå■ęè£ŘçőŮæÌěæŞ■ä¡IJåőČäžñãĂĆæŕŤåęĆïijŽ

```
>>> from datetime import datetime
>>> a = datetime(2012, 9, 23)
>>> print(a + timedelta(days=10))
2012-10-03 00:00:00
>>>
>>> b = datetime(2012, 12, 21)
>>> d = b - a
>>> d.days
89
>>> now = datetime.today()
>>> print(now)
2012-12-21 14:54:43.094063
>>> print(now + timedelta(minutes=10))
2012-12-21 15:04:43.094063
>>>
```

åIJÍèőąçőŮçŽĎæŮűůĂŹïijŇéIJĂèęĄæşÍæĎŔçŽĎæŸŕ `datetime` äijŽèĞłåŁÍåď˙ĎçŘĘéŮřåźť˙ãĂĆærŤåęĆïijŽ

```
>>> a = datetime(2012, 3, 1)
>>> b = datetime(2012, 2, 28)
>>> a - b
datetime.timedelta(2)
>>> (a - b).days
2
>>> c = datetime(2013, 3, 1)
>>> d = datetime(2013, 2, 28)
>>> (c - d).days
1
>>>
```

èőÍèőž

åŕźåď˙ğåď˙ŽæŢřå§žæIJňçŽĎæŮěæIJ§åŠŇæŮűéŮt˙åď˙ĎçŘĘéŮőécŸïijŇ `datetime` æÍąåİŮäžěåŘŁèűşåď˙§äžĘãĂĆ åęĆæđIJä¡ăéIJĂèęĄæL˙ğèąŇæŻť˙åŁååď˙■æİĆçŽĎæŮěæIJ§æŞ■ä¡IJïijŇærŤåęĆ åŘŕäžěèĂČèŹŠä¡£çŤÍ dateutilæÍąåİŮ

èőÿåď˙Žçśžäijijç ŽĎæŮűéŮt˙èőąçőŮåŘŕäžěä¡£çŤÍ `dateutil.relativedelta()` åĞ¡æŢřäžčæŻ£ãĂĆ ä¡ĘæŸŕïijŇæIJL˙äÿĂçĆźéIJĂèęĄæşÍæĎŔçŽĎåřśæŸŕïijŇåőČäijŽåIJÍåď˙ĎçŘĘæIJÍäż¡(è

```
>>> a = datetime(2012, 9, 23)
>>> a + timedelta(months=1)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: 'months' is an invalid keyword argument for this function
>>>
>>> from dateutil.relativedelta import relativedelta
>>> a + relativedelta(months=+1)
datetime.datetime(2012, 10, 23, 0, 0)
```

```
>>> a + relativedelta(months=+4)
datetime.datetime(2013, 1, 23, 0, 0)
>>>
>>> # Time between two dates
>>> b = datetime(2012, 12, 21)
>>> d = b - a
>>> d
datetime.timedelta(89)
>>> d = relativedelta(b, a)
>>> d
relativedelta(months=+2, days=+28)
>>> d.months
2
>>> d.days
28
>>>
```

3.13 èőąçőŮæIJĂåŘŐäÿĂäÿłåŚĺäžŤçŽĎæŮěæIJ§

éŮőécŸ

ä¡ăéIJĂèęĄæ§ěæL'¿æŸ§æIJ§äÿ■æ§ŘäÿĂåď'l'æIJĂåŘŐåĞžçŐřçŽĎæŮěæIJ§ïijŇæŕŤåęĆæŸ§æIJ§äžŤãĂ

èğčåĘşæŰźæąĹ

PythonçŽĎ `datetime` æĺąåİŮäÿ■æIJL'åůěåĚůåĞ¡æŢřåŠŇçśżåŔŕäżěåÿőåŁl'ä¡ăæL'ğèąŇè£ŹæăůçŽĎèőäÿŃéIćæŸŕåŕźçśżäijijè£ŹæăůçŽĎéŮőécŸçŽĎäÿĂäÿłéĂŽçŤĺèğčåĘşæŰźæąĹïijŽ

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
"""
Topic: æIJĂåŘŐçŽĎåŚĺäžŤ
Desc :
"""
from datetime import datetime, timedelta

weekdays = ['Monday', 'Tuesday', 'Wednesday', 'Thursday',
            'Friday', 'Saturday', 'Sunday']


def get_previous_byday(dayname, start_date=None):
    if start_date is None:
        start_date = datetime.today()
    day_num = start_date.weekday()
    day_num_target = weekdays.index(dayname)
    days_ago = (7 + day_num - day_num_target) % 7
    if days_ago == 0:
```

```
        days_ago = 7
    target_date = start_date - timedelta(days=days_ago)
    return target_date
```

åIJÍäžď'äžŠåijŔèğčéĞŁåŹÍäÿ■ä¡£çŤÍåęĆäÿŃïijŽ

```
>>> datetime.today() # For reference
datetime.datetime(2012, 8, 28, 22, 4, 30, 263076)
>>> get_previous_byday('Monday')
datetime.datetime(2012, 8, 27, 22, 3, 57, 29045)
>>> get_previous_byday('Tuesday') # Previous week, not today
datetime.datetime(2012, 8, 21, 22, 4, 12, 629771)
>>> get_previous_byday('Friday')
datetime.datetime(2012, 8, 24, 22, 5, 9, 911393)
>>>
```

åŘŕéĂL'çŽĎ `start_date` åŔĆæŢřåŘŕäžěçŤśåŔęåď'ŰäÿĂäÿł `datetime` åőđä¿ŃæİěæŔŘä¿ŻãĂĆæŕŤåęĆïijŽ

```
>>> get_previous_byday('Sunday', datetime(2012, 12, 21))
datetime.datetime(2012, 12, 16, 0, 0)
>>>
```

èőÍèőž

äÿŁéİćçŽĎçőŮæşŢåŐ§çŘĘæŸŕè£Źæăůç ŽĎïijŽåĚĹářĘåijĂåğŃæŮěæIJ§åŠŇçŻőæăĞæŮěæIJ§æŸăářĎ çĎűåŘŐéĂŽè£ĞæÍąè£ŘçőŮèőąçőŮåĞžçŻőæăĞæŮěæIJ§èęAçżŔè£Ğåď'ŽářŚåď'l'æL'■èČ¡åĹřè¿¿åijĂåğŃæŮ

åęĆædIJä¡ăèęĄåČŔè£ŹæăůæL'ğèąŇåď'ğéĞŔçŽĎæŮěæIJ§èőąçőŮçŽĎèŕİïijŇä¡ăæIJĂåě¡åőL'èčĚçňňäÿL `python-dateutil` æİěäžčæŻ£ãĂĆ æŕŤåęĆïijŇäÿŃéİćæŸŕæŸŕä¡£çŤÍ `dateutil` æÍąåİŮäÿ■çŽĎ `relativedelta()` åĞ¡æŢřæL'ğèąŇåŘŇæăůçŽĎèőąçőŮïijŽ

```
>>> from datetime import datetime
>>> from dateutil.relativedelta import relativedelta
>>> from dateutil.rrule import *
>>> d = datetime.now()
>>> print(d)
2012-12-23 16:31:52.718111

>>> # Next Friday
>>> print(d + relativedelta(weekday=FR))
2012-12-28 16:31:52.718111
>>>

>>> # Last Friday
>>> print(d + relativedelta(weekday=FR(-1)))
2012-12-21 16:31:52.718111
>>>
```

3.14 èőąçőŮå¡ŞåĽ■æĲĹäż¡çŽĎæŮĕæĲ§èŇČåŻť

éŮőéćŸ

ä¡ăçŽĎäżčçăĄéĲĂèęĄåĲÍå¡ŞåĽ■æĲĹäż¡äÿ■å¿łçŐŕæŕŘäÿĂåď'ľïijŇæČşæĽ¿åĹřäÿĂäÿłèőąçőŮè£Źäÿła

èğčåĘşæŰźæąĹ

åĲÍè£Źæăůçŕ ŽĎæŮĕæĲ§äÿŁå¿łçŐŕåźűéĲĂèęĄäžŃåĚĹæđĎéĂăäÿĂäÿłåŇĚåŘńæĽĂæĲĽæŮĕæĲ§çŽĎ
ä¡ăäåŘŕäžĕåĚĹèőąçőŮåĞžåijĂåğŃæŮĕæĲ§åŠŇçżŞæİ§æŮĕæĲ§ïijŇ
çĎűåŘŐåĲÍä¡ăæ■ěè£ŻçŽĎæŮűåĂŹä¡£çŤÍ datetime.timedelta
åŕźèśąéĂŠåćđè£Źäÿłæ ŮĕæĲ§åŘŸéĞŔå■şåŔŕãĂĆ

äÿŃéİćæŸŕäÿĂäÿłæŐěåŔŮäżżæĎŔ datetime åŕźèśąåźűè£ŤåŻđäÿĂäÿłçŤśå¡ŞåĽ■æĲĹäż¡åijĂåğŃæŮ

```
from datetime import datetime, date, timedelta
import calendar

def get_month_range(start_date=None):
    if start_date is None:
        start_date = date.today().replace(day=1)
    _, days_in_month = calendar.monthrange(start_date.year, start_
↪date.month)
    end_date = start_date + timedelta(days=days_in_month)
    return (start_date, end_date)
```

æĲĽäžĘè£Źäÿłåřśåŕäžĕå¿ĹåőźæŸŞçŽĎåĲÍè£ŤåŻđçŽĎæŮĕæĲ§èŇČåŻť äÿŁéİćåĄŽå¿ŐçŐŕæŞ■ä¡ĲäžĘ

```
>>> a_day = timedelta(days=1)
>>> first_day, last_day = get_month_range()
>>> while first_day < last_day:
...     print(first_day)
...     first_day += a_day
...
2012-08-01
2012-08-02
2012-08-03
2012-08-04
2012-08-05
2012-08-06
2012-08-07
2012-08-08
2012-08-09
#... and so on...
```

èőÍèőž

äÿŁéÌćçŽĎäžčçăĄåĚĹèőąçőŮåĞžäÿĂäÿłåŕźåžŤæIJĹäż¡çňňäÿĂåď'ľçŽĎæŮěæIJ§ãĂĆ äÿĂäÿłå£ńéĂ§çŽĎæŰźæşŢåřśæŸŕä¡£çŤÍ `date` æĹŰ `datetime` åŕźèśąçŽĎ `replace()` æŰźæşŢçőĂå■ŢçŽĎåřĘ `days` åśđæĂğèő¿ç¡őæĹŘ1å■şåŔŕãĂĆ `replace()` æŰźæşŢäÿĂäÿłåě¡åď'ĎåřśæŸŕåőČäijŽåĹŻåžžåŠŇä¡ăåijĂåğŃäijăåĚěåŕźèśąçśžåđŃçŻÿåŘŇçŽĎåŕźèśąãĂĆ æĽ'Ăäžěïij Ňåę Ćæđ IJè¿ŞåĚěåŔĆæŢřæŸŕäÿĂäÿł `date` åőđä¿ŃïijŇéĆčäźĹçżŞæđIJäź§æŸŕäÿĂäÿł `date` åőđä¿ŃãĂĆ åŘŇæăůçŽĎïijŇåęĆæđIJè¿ŞåĚěæŸŕäÿĂäÿł `datetime` åőđä¿ŃïijŇéĆčäźĹä¡ăå¿ŮåĹřçŽĎåřśæŸŕäÿĂäÿł `datetime` åőđä¿ŃãĂĆ

çĎŭåŘŐïijŇä¡£çŤÍ `calendar.monthrange()` åĞ¡æŢřæİěæĽ¿åĞžèŕěæIJĹçŽĎæĂžåď'ľæŢřãĂĆ äžžä¡ŢæŮŭåĂŹåŔłèęĄä¡ăæČşèŐůå¿ŮæŮěåŐĘä£ąæĄŕïijŇéĆčäźĹ `calendar` æÍąåİŮåřśéİđåÿÿæIJĽçŤÍäžĘãĂĆ `monthrange()` åĞ¡æŢřäijŽè£ŤåŻđåŇĚåŘńæŸ§æIJ§åŠŇèŕěæIJĹåď'ľæŢřçŽĎåĚČçżĎãĂĆ

äÿĂæŮęèŕěæIJĹçŽĎåď'ľæŢřåůšç§ěäžĘïijŇéĆčäźĹçżŞæİ§æŮěæIJ§åřśåŔŕäžěéĂŽè£ĞåIJÍåijĂåğŃæŮěæ æIJĽäÿłéIJĂèęĄæşÍæĎŔçŽĎæŸŕçżŞæİ§æŮěæIJ§ázűäÿ■åŇĚåŘńåIJÍè£ŹäÿłæŮěæIJ§èŇČåŻť'åĘĚ(äžŃåőđäÿ è£ŹäÿłåŠŇPythonçŽĎ `slice` äÿŐ `range` æŞ■ä¡IJèąŇäÿžä£ÍæŇĄäÿĂèĞť'ïijŇåŘŇæăůäź§äÿ■åŇĚåŘńçżŞåı

äÿžäžĘåIJÍæŮěæIJ§èŇČåŻť'äÿŁå¿łçŐŕïijŇèęĄä¡£çŤÍåĹřæăĞåĞĘçŽĎæŢřå■ęåŠŇærŤè¿ČæŞ■ä¡IJãĂĆ ærŤåęĆïijŇåŔŕäžěåĹl'çŤÍ `timedelta` åőđä¿ŃæÍěéĂŠåćđæŮěæIJ§ïijŇårŔäžŐåŔů<çŤÍæÍěæčĂæ§ěäÿĂäÿła

çŘĘæČşæČĚåĘţäÿŃïijŇåęĆæđIJèČ¡äÿžæŮěæIJ§è£■äžčåĹŻåžžäÿĂäÿłåŘŇåĘĚç¡őçŽĎ `range()` åĞ¡æŢřäÿĂæăůçŽĎåĞ¡æŢřåŕśåě¡äžĘãĂĆ åźÿè£ŘçŽĎæŸŕïijŇåŔŕäžěä¡£çŤÍäÿĂäÿłçŤ§æĹŘåŹÍçŽĎä¿Ńå■ŘïijŽ

```
def date_range(start, stop, step):
    while start < stop:
        yield start
        start += step
```

äÿŃéÍćæŸŕä¡£çŤÍè£ŹäÿłçŤ§æĹŘåŹÍçŽĎä¿Ńå■ŘïijŽ

```
>>> for d in date_range(datetime(2012, 9, 1), datetime(2012,10,1),
                        timedelta(hours=6)):
...     print(d)
...
2012-09-01 00:00:00
2012-09-01 06:00:00
2012-09-01 12:00:00
2012-09-01 18:00:00
2012-09-02 00:00:00
2012-09-02 06:00:00
...
>>>
```

è£Źçğ■åőđçŐŕäźŃæL'Ăäžěè£ŹäźĹçőĂå■ŢïijŇè£Ÿå¿Ůå¡ŠåŁ§äžŐPythonäÿ■çŽĎæŮěæIJ§åŠŇæŮűéŮť'ă

3.15 å■Ůçňęäÿšè¡ňæ■ćäÿžæŮěæIJ§

éŮőécŸ

ä¡ăçŽĎåžŤçŤÍçÍŃåžŔæŐěåŔŮå■ŮçňęäÿšæäijåijŔçŽĎè¿ŞåĚěïijŇä¡ĘæŸŕä¡ăæČşářĘåőČäžňè¡ňæ■ćäÿž datetime åŕźèśąäžěä¿£åIJÍäÿŁéÍćæĽğèąŇéÍđå■Ůçňęäÿšæş■ä¡IJãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤÍPythonçŽĎæăĞåĞĘæÍąåİŮ datetime åŔŕäžěå¿ĹåőźæŸ§çŽĎèğčåĘşè£ŹäÿłéŮőécŸãĂĆærŤåęĆ

```
>>> from datetime import datetime
>>> text = '2012-09-20'
>>> y = datetime.strptime(text, '%Y-%m-%d')
>>> z = datetime.now()
>>> diff = z - y
>>> diff
datetime.timedelta(3, 77824, 177393)
>>>
```

èőÍèőž

datetime.strptime() æŰźæşŢæŤŕæŇĄå¿Ĺåd'ŽçŽĎæäijåijŔåŇŰäžčçăĄïijŇ ærŤåęĆ %Y äžčèąÍ4ä¡■æŢřåźt'äž¡ïijŇ %m äžčèąÍäÿd'ä¡■æŢřæIJĹäž¡ãĂĆ è£ŸæIJL'äÿĂçĆźåĂijå¿ŮæşÍæĎŔçŽĎæŸŕè£ŹäžŻæäijåijŔåŇŰå■ää¡■çňęäź§åŔŕäžěåŔ■è£ĞæÍěä¡£çŤÍïijŇårE

ærŤåęĆïijŇåĄĞèő¿ä¡ăçŽĎäžčçăĄäÿ■çŤ§æĹŘäžĘäÿĂäÿł datetime åŕźèśąïijŇ ä¡ăæČşårĘåőČæäijåijŔåŇŰäÿžæijĆäžőæŸŞèŕźå¡ćåijŔåŘŐæŤ¿åIJÍèĞłåŁÍçŤ§æĹŘçŽĎä£ąäžűæĹŰèĂĚæŁěåS

```
>>> z
datetime.datetime(2012, 9, 23, 21, 37, 4, 177393)
>>> nice_z = datetime.strftime(z, '%A %B %d, %Y')
>>> nice_z
'Sunday September 23, 2012'
>>>
```

è£ŸæIJL'äÿĂçĆźéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijŇ strptime() çŽĎæĂğèČ¡èęĄærŤä¡ăæČşèśąäÿ■çŽĎåůőå¿Ĺåd'ŽïijŇ åŻăäÿžåőČæŸŕä¡£çŤÍçžfPythonåőđçŐřïijŇåźűäÿŤå£Ì åęĆædIJä¡ăèęĄåIJÍäžčçăĄäÿ■éIJĂèęĄèğčædŘåd'ğéĞŔçŽĎæŮěæIJ§åźűäÿŤåůšçżŔç§ěéĄŞäžĘæŮěæIJ§å■Ůç ærŤåęĆïijŇåęĆædIJä¡ăåůšçżŔç§ěéĄŞæĽĂäžěæŮěæIJ§æăijåijŔæŸŕ YYYY-MM-DD ïijŇä¡ăåŔŕäžěåČŔäÿŇéÍćè£ŹæăůåőđçŐřäÿĂäÿłèğčædŘåĞ¡æŢřïijŽ

```
from datetime import datetime
def parse_ymd(s):
    year_s, mon_s, day_s = s.split('-')
    return datetime(int(year_s), int(mon_s), int(day_s))
```

åőđéŹĚætŃèŕŢäÿ■ïijŇè£ŹäÿłåĞ¡æŢřærŤ datetime.strptime() å£ń7åĂ■åd'ŽãĂĆ åęĆædIJä¡ăèęĄåd'ĎçŘĘåd'ğéĞŔçŽĎæűĽåŔŁåĹřæŮěæIJ§çŽĎæŢřæ■őçŽĎèŕÏïijŇéĆčäźĹæIJĂåě¡èĂČèŹŠäy

3.16 çżŞåŘĹæŮűåŇžçŽĎæŮěæIJ§æŞ■ä¡IJ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłåőL'æŐŠåIJĺ2012åźt' 12æIJĹ21æŮěæŮl'äÿŁ9:30çŽĎçŤţèŕİäijŽèőőïijŇåIJřçĆźåIJĺèŁİåŁă
èĂŇä¡ăçŽĎæIJŃåŘŃåIJĺå■řåžęçŽĎçŔ■åŁăç¡ŮåřŤïijŇéĆčäźĹäžŰåžŤèŕěåIJĺå¡ŞåIJřæŮűéŮt' åĞăçĆźåŘĆåŁă

èğčåĘşæŰźæąĹ

åŕźåĞääźŐæL'ĂæIJL'æűL'åŘŁåĹřæŮűåŇžçŽĎéŮőécŸïijŇä¡äéČ¡åžŤèŕěä¡£çŤĺ
pytz æĺąåİŮãĂĆè£ŹäÿłåŇĚæŘŘä¿ŻäžĘOlsonæŮűåŇžæŢřæ■őåžŞïijŇ
åőČæŸřæŮűåŇžä£qæĄŕçŽĎäžŃåőđäÿŁçŽĎæăĞåĞĘïijŇåIJĺå¿Ĺåd'Žèŕ■èĺĂåŠŇæŞ■ä¡IJçşżçż§éĞŇéİćéČ¡åŘ

pytz æĺąåİŮäÿĂäÿłäÿżèęĄçŤĺéĂŤæŸřåřĘ datetime
åžŞåĹŻåżžçŽĎçőĂå■ŢæŮěæIJ§åŕźèśąæIJňåIJřåŇŰãĂĆ ærŤåęĆïijŇäÿŃéİcåęĆä¡Ţèąĺçd'žäÿĂäÿłèŁİåŁääŞěa

```
>>> from datetime import datetime
>>> from pytz import timezone
>>> d = datetime(2012, 12, 21, 9, 30, 0)
>>> print(d)
2012-12-21 09:30:00
>>>

>>> # Localize the date for Chicago
>>> central = timezone('US/Central')
>>> loc_d = central.localize(d)
>>> print(loc_d)
2012-12-21 09:30:00-06:00
>>>
```

äÿĂæŮęæŮěæIJ§èćńæIJňåIJřåŇŰäžĘïijŇ åőČåřśåŘŕäżěè¡ňæ■ćäÿžåĚűäżŰæŮűåŇžçŽĎæŮűéŮt' äžĘãĂĆ
äÿžäžĘå¿ŮåĹřçŔ■åŁăç¡ŮåřŤåŕźåžŤçŽĎæŮűéŮt'ïijŇä¡ăåŘŕäżěè£Źæăůå ĄŽïijŽ

```
>>> # Convert to Bangalore time
>>> bang_d = loc_d.astimezone(timezone('Asia/Kolkata'))
>>> print(bang_d)
2012-12-21 21:00:00+05:30
>>>
```

åęĆædIJä¡ăæL'ŞçőŮåIJĺæIJňåIJřåŇŰæŮěæIJ§äÿŁæL'ğèąŇèőąçőŮïijŇä¡ăéIJĂèęĄçL'źåĹńæşĺæĎŘåd'Ŕä
ærŤåęĆïijŇåIJĺ2013åźt'ïijŇç¿ŐåŻ¡æăĞåĞĘåd'Ŕäżd'æŮűæŮűéŮt'åijĂågŇäžŐæIJňåIJřæŮűéŮt' 3æIJĹ13æŮè
åęĆædIJä¡ăæ■čåIJĺæL'ğèąŇæIJňåIJřèőąçőŮïijŇä¡ăäijŽå¿ŮåĹřäÿĂäÿłéŤŹèŕřãĂĆærŤåęĆïijŽ

```
>>> d = datetime(2013, 3, 10, 1, 45)
>>> loc_d = central.localize(d)
>>> print(loc_d)
2013-03-10 01:45:00-06:00
>>> later = loc_d + timedelta(minutes=30)
>>> print(later)
```

```
2013-03-10 02:15:00-06:00 # WRONG! WRONG!
>>>
```

çżŞæđIJéŤŹèŕŕæŸŕåŻääÿžåőČåźűæšąæIJL'èĂČèŹŚåIJÍæIJňåIJřæŮűéŮt'äÿ■æIJL'äÿĂåřŔæŮűçŽĎèůşèů
äÿžäžĘä£őæ■čè£ŹäÿłéŤŹèŕŕijŇåŔŕäžěä¡£çŤÍæŮűåŇžåŕźèšą `normalize()`
æŰźæşŢãĂĆæŕŤåęĆïijŽ

```
>>> from datetime import timedelta
>>> later = central.normalize(loc_d + timedelta(minutes=30))
>>> print(later)
2013-03-10 03:15:00-05:00
>>>
```

èőÍèőž

äÿžäžĘäÿ■èől'ä¡ăèćńè£ŹäžŻäÿIJäÿIJåijĎçŽĎæŹŢåd'ť'è¡ňåŘŚïijŇåd'ĎçŘĘæIJňåIJřåŇŰæŮěæIJ§çŽĎéĂ
åźűçŤÍåőČæÍěæL'ğèąŇæL'ĂæIJL'çŽĎäÿ■éŮt'å■ŸåĆÍåŠŇæŞ■ä¡IJãĂĆæŕŤåęĆïijŽ

```
>>> print(loc_d)
2013-03-10 01:45:00-06:00
>>> utc_d = loc_d.astimezone(pytz.utc)
>>> print(utc_d)
2013-03-10 07:45:00+00:00
>>>
```

äÿĂæŮęè¡ňæ■ćäÿžUTCïijŇä¡ăåřśäÿ■çŤÍåŐżæŃĚå£Čèů§åd'Ŕäžd'æŮűçŻÿåĚşçŽĎéŮőécŸäžĘãĂĆ
åŻăæ■d'ïijŇä¡ăåŔŕäžěèů§äźŃåL'■äÿĂæăůæŤ¿å£ČçŽĎæL'ğèąŇåÿÿèğĄçŽĎæŮěæIJ§èőąçőŮãĂĆ
å¡Şä¡ăæČşåřĘè¿ŞåĞžåŔŸäÿžæIJňåIJřæŮűéŮt'çŽĎæŮűåĂŹïijŇä¡£çŤÍåŘĹéĂĆçŽĎæŮűåŇžåŐżè¡ňæ■ćäÿŃå

```
>>> later_utc = utc_d + timedelta(minutes=30)
>>> print(later_utc.astimezone(central))
2013-03-10 03:15:00-05:00
>>>
```

å¡ŞæűL'åŔŁåĹřæŮűåŇžæŞ■ä¡IJçŽĎæŮűåĂŹïijŇæIJL'äÿłéŮőécŸåřśæŸŕæĹŚäžňåęĆä¡Ţå¿ŮåĹřæŮűåŇ
æŕŤåęĆïijŇåIJÍè£ŹäÿłäżÑå■Řäÿ■ïijŇæĹŚäžňåęĆä¡Ţç§ěéĄŞâĂIJAsia/KolkataâĂİåřśæŸŕå■řåžęåŕźåžŤçŽĎæ
äÿžäžĘæ§ěæL'¿ïijŇåŔŕäžěä¡£çŤÍISO 3166åŻ¡åőűäžčçăĄä¡IJäÿžåĚşéŤőå■ŮåŐżæ§ěéŸĚå■ŮåĚÿ
`pytz.country_timezones` ãĂĆæŕŤåęĆïijŽ

```
>>> pytz.country_timezones['IN']
['Asia/Kolkata']
>>>
```

æşÍïijŽå¡Şä¡ăéŸĚèŕźåĹřè£ŹéĞŇçŽĎæŮűåĂŹïijŇæIJL'åŔŕèČ¡ `pytz`
æÍąåİŮåůšçżŔäÿ■åĘ■åžžèőőä¡£çŤÍäžĘïijŇåŽääÿžPEP431æŔŘåĞžäžĘæŻť'åĚÍè£ŻçŽĎæŮűåŇžæŤŕæŇĄãĂ
ä¡ĘæŸŕè£ŹéĞŇèřĹåĹřçŽĎå¿Ĺåd'ŽéŮőécŸè£ŸæŸŕæIJL'åŔĆèĂČäžůåÄijçŽĎ(æŕŤåęĆä¡£çŤÍUTCæŮěæIJ§ç

çňňåŻŻçńăïijŽè£■äžčåŹĺäÿŐçŤ§æĹŘåŹĺ

è£■äžčæŸŕPythonæIJĂåijžåď˘ğçŽĎåŁ§èČ¡äźŃäÿĂãĂĆåĹİçIJŃèţůæİěïijŇä¡ăåŔŕèČ¡äijŽçőĂå■ŢçŽĎèőc
çĎűèĂŇïijŇçżIéİđäžĚäžĚåřśæŸŕåęĆæ■ď˘ïijŇè£ŸæIJL’å¿Ĺåď˘Žä¡ăåŔŕèČ¡äÿ■ç§ěéĄŞçŽĎïijŇ
æŕŤåęĆåĹŻåžžä¡ăèĞłåůśçŽĎè£■äžčåŹĺåŕźèśąïijŇåIJĺitertoolsæĺąåİŮäÿ■ä¡£çŤĺæIJL’çŤĺçŽĎè£■äžčæĺąåijŔïijĺ
è£ŹäÿĂçńăçŽőçŽĎåřśæŸŕåŘŠä¡ăåśŢçď˘žèů§è£■äžčæIJL’åĚşçŽĎåŘĎçğ■åÿÿèğĄéŮőécŸãĂĆ

Contents:

4.1 æĽŃåŁĺéĄ■åŐĘè£■äžčåŹĺ

éŮőécŸ

ä¡ăæČşéĄ■åŐĘäÿĂäÿłåŔŕè£■äžčåŕźèśąäÿ■çŽĎæL’ĂæIJL’åĚČçť˘ăïijŇä¡ĘæŸŕå■ť˘äÿ■æČşä¡£çŤĺforå¿łçČ

èğčåĘşæŰźæąĹ

äÿžäžĘæL’ŃåŁĺçŽĎéĄ■åŐĘåŔŕè£■äžčåŕźèśąäÿ■çŽĎæL’ĂæIJL’åĚČçť˘ăïijŇä¡£çŤĺ `next()`
åĞ¡æŢřåźűåIJĺäžčçăĄäÿ■æ■ŢèŐů `StopIteration` åijĆåÿÿãĂĆ
æŕŤåęĆïijŇäÿŃéİćçŽĎä¿Ńå■ŘæL’ŃåŁĺèŕżåŘŰäÿĂäÿłæŰĞäžűäÿ■çŽĎæL’ĂæIJL’èąŇïijŽ

```
def manual_iter():
    with open('/etc/passwd') as f:
        try:
            while True:
                line = next(f)
                print(line, end='')
        except StopIteration:
            pass
```

éĂŽåÿÿæİěèőšïijŇ `StopIteration` çŤĺæİěæŇĞçď˘žè£■äžčçŽĎçżŞår¿ãĂĆ
çĎűèĂŇïijŇåęĆædIJä¡ăæL’ŃåŁĺä¡£çŤĺäÿŁéİćæijŢçď˘žçŽĎ `next()`
åĞ¡æŢřçŽĎèŕİïijŇä¡ăè£ŸåŔŕäžčéĂŽè£Ğè£ŤåŽđäÿĂäÿłæŇĞåőŽåĂijæİěæăĞèőřçżŞår¿ïijŇæŕŤåęĆ
`None` ãĂĆ äÿŃéİćæŸŕçď˘žä¿ŃïijŽ

```
with open('/etc/passwd') as f:
    while True:
        line = next(f, None)
        if line is None:
            break
        print(line, end='')
```

èőĺèőž

åď˘ğåď˘ŽæŢřæČĚåĘţäÿŃïijŇæĹŚäžňäijŽä¡£çŤĺ `for` å¿łçŐŕèŕ■åŔěçŤĺæİěéĄ■åŐĘäÿĂäÿłåŔŕè£■äžčåŕźè
ä¡ĘæŸŕïijŇåĄűåřŤäź§éIJĂèęĄåŕźè£■äžčåĄŽæŻť˘åŁăçš¿çąőçŽĎæŐğåĹűïijŇè£ŹæŮűåĂŹäžĘèğčåžŢåśĆè£■ä

äÿŃéİćçŽĎäžď'äžŠçď'žä¿ŃåŘŚæĹŚäžňæijŤçď'žäžĘè£■äžčæIJ§éŮť'æL'ĂåŔŚçŤ§çŽĎå§žæIJňçżĘèŁĆïijŽ

```
>>> items = [1, 2, 3]
>>> # Get the iterator
>>> it = iter(items) # Invokes items.__iter__()
>>> # Run the iterator
>>> next(it) # Invokes it.__next__()
1
>>> next(it)
2
>>> next(it)
3
>>> next(it)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
StopIteration
>>>
```

æIJňçńäæŐěäÿŃæİěåĞäåřŔèŁĆäijŽæŻť'æůśåĚěçŽĎèőšèğčè£■äžčçŻÿåĘşæŁĂæIJŕïijŇåL'■æŔŘæŸŕä¡ăă
æL'Ăäžěçąőä£Ïä¡ăåůšçżŔæŁŁè£Źçńäç ŽĎåĘĚåőźçL'ćçL'ćèőřåIJĺå£Čäÿ■ãĂĆ

4.2 äžčçŘĘè£■äžč

éŮőécŸ

ä¡ăæđĎåżžäžĘäÿĂäÿłèĞłåőŽäźL'åőźåŹĺåŕźèśąïijŇéĞŇéİćåŇĚåŘńæIJL'åĹŮèąĺãĂĄåĚČçżĎæĹŰåĚűäżŮ
ä¡ăæČşçŻť'æŐěåIJĺä¡äçŽĎè£ŹäÿłæŰřåőźåŹĺåŕźèśąäÿŁæL'ğèąŇè£■äžčæŞ■ä¡IJãĂĆ

èğčåĘşæŰźæąĹ

åőđéŹĚäÿŁä¡ăåŔłéIJĂèęĄåőŽäźL'äÿĂäÿł __iter__()
æŰźæşŢïijŇářĘè£■äžčæŞ■ä¡IJäžčçŘĘåĹřåőźåŹĺåĘĚéČĺçŽĎåŕźèśąäÿŁåŐžãĂĆærŤåęĆïijŽ

```
class Node:
    def __init__(self, value):
        self._value = value
        self._children = []

    def __repr__(self):
        return 'Node({!r})'.format(self._value)

    def add_child(self, node):
        self._children.append(node)

    def __iter__(self):
        return iter(self._children)

# Example
```

```
if __name__ == '__main__':
    root = Node(0)
    child1 = Node(1)
    child2 = Node(2)
    root.add_child(child1)
    root.add_child(child2)
    # Outputs Node(1), Node(2)
    for ch in root:
        print(ch)
```

åIJÍäÿŁéÍćäžčçăĄäÿ■ïijŇ `__iter__()` æŰźæşŢåŔłæŸŕçőĂå■ŢçŽĎåřĘè£■äžčèŕůæśĆäijăéĂŠçżŹåĘĚé `_children` åśđæĂğãĂĆ

èőÍèőž

PythonçŽĎè£■äžčåŹÍå■ŔèőőéIJĂèęĄ `__iter__()` æŰźæşŢè£ŤåŻđäÿĂäÿłåőđçŐřäžĘ `__next__()` æŰźæşŢçŽĎè£■äžčåŹÍåŕźèśąãĂĆ åęĆæđIJä¡ăåŔłæŸŕè£■äžčéĄ■åŐĘåĚűäžŰåőźåŹÍçŽĎåĘĚå

è£ŹéĞŇçŽĎ `iter()` åĞ¡æŢřçŽĎä¡£çŤÍçőĂåŇŰäžĘäžčçăĄïijŇ `iter(s)` åŔłæŸŕçőĂå■ŢçŽĎéĂŽè£ĞèřČçŤÍ `s.__iter__()` æŰźæşŢæÍèè£ŤåŻđåŕźåžŤçŽĎè£■äžčåŹÍåŕźèśąïijŇ åŕśèů§ `len(s)` äijŽèřČçŤÍ `s.__len__()` åŐ§çŘĘæŸŕäÿĂæăůçŽĎãĂĆ

4.3 ä¡£çŤÍçŢ§æĹŘåŹÍåĹŻåžžæŰřçŽĎè£■äžčæÍąåijŔ

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłèĞłåőŽäźL'è£■äžčæÍąåijŔïijŇèů§æŹőéĂŽçŽĎåĘĚç¡őåĞ¡æŢřæŕŤåęĆ `range()`，`reversed()` äÿ■äÿĂæăůãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşåőđçŐřäÿĂçğ■æŰřçŽĎè£■äžčæÍąåijŔïijŇä¡£çŤÍäÿĂäÿłçŢ§æĹŘåŹÍåĞ¡æŢřæÍèåőŽäźL'å äÿŃéÍćæŸŕäÿĂäÿłçŢ§äžğæ§ŘäÿłèŇČåŻť åĘĚæţőçĆźæŢřçŽĎçŢ§æĹŘåŹÍïijŽ

```
def frange(start, stop, increment):
    x = start
    while x < stop:
        yield x
        x += increment
```

äÿžäžĘä¡£çŤÍè£ŹäÿłåĞ¡æŢřïijŇ ä¡ăåŔŕäžěçŤÍforå¿łçŐrè£■äžčåőČæĹŰèĂĚä¡£çŤÍåĚűäžŰæŐěåŔŮäÿĂä `sum()`，`list()` ç■Ľ')ãĂĆçd'žä¿ŃåęĆäÿŇïijŽ

```
>>> for n in frange(0, 4, 0.5):
...     print(n)
```

```
...
0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
>>> list(frange(0, 1, 0.125))
[0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875]
>>>
```

èőÍèőž

äÿĂäÿłåĞ¡æŢřäÿ■éIJĂèęĄæIJL’äÿĂäÿł `yield` èí■åŔěå■şåŔŕåřĘåĚűè¡ňæ■ćäÿžäÿĂäÿłçŤ§æĹŘåŹĺãĂĆ èů§æŹőéĂŽåĞ¡æŢřäÿ■åŘŇçŽĎæŸŕïijŇçŤ§æĹŘåŹĺåŔłèČ¡çŤĺäžŐè£■äżčæŞ■ä¡IJãĂĆ äÿŃéÍćæŸŕäÿĂäÿłåőđéłŇïijŇåŘŚä¡ăåśŢçď žè£ŹæăůçŽĎåĞ¡æŢřåžŢåśĆåůëä¡IJæIJžåĹűïijŽ

```
>>> def countdown(n):
...     print('Starting to count from', n)
...     while n > 0:
...         yield n
...         n -= 1
...     print('Done!')
...

>>> # Create the generator, notice no output appears
>>> c = countdown(3)
>>> c
<generator object countdown at 0x1006a0af0>

>>> # Run to first yield and emit a value
>>> next(c)
Starting to count from 3
3

>>> # Run to the next yield
>>> next(c)
2

>>> # Run to next yield
>>> next(c)
1

>>> # Run to next yield (iteration stops)
>>> next(c)
Done!
Traceback (most recent call last):
```

```
    File "<stdin>", line 1, in <module>
StopIteration
>>>
```

äÿĂäÿłçŤ§æĹŘåŹĺåĞ¡æŢřäÿżèęĄçĽźå¿ĄæŸŕåőČåŔłäijŽåŻđåžŤåIJĺè£■äżčäÿ■ä¡£çŤĺåĹřçŽĎ
next æŞ■ä¡IJãĂĆ äÿĂæŮęçŤ§æĹŘåŹĺåĞ¡æŢřè£ŤåŻđéĂĂåĞžïijŇè£■äżčçżĹæ■ćãĂĆæĹŚäžňåIJĺè£■äżčäÿ■é

4.4 åőđçŐřè£■äżčåŹĺå■Ŕèőő

éŮőécŸ

ä¡ăæČşæđĎåžžäÿĂäÿłèČ¡æŤŕæŇĄè£■äżčæŞ■ä¡IJçŽĎèĞłåőŽäźĽ'åŕźèśąïijŇåźűåÿŇæIJŻæĽ'¿åĹřäÿĂäÿł

èğčåĘşæŰźæąĹ

çŽőåĽ'■äÿžæ■ćïijŇåIJĺäÿĂäÿłåŕźèśąäÿŁåőđçŐřè£■äżčæIJĂçőĂå■ŢçŽĎæŰźåijŔæŸŕä¡£çŤĺäÿĂäÿłçŤ§æ
åIJĺ4.2åřŔèŁĆäÿ■ïijŇä¡£çŤĺNodeçśżæİèèąĺçđ'žæăŚå¡ćæŢřæ■őçżŞæđĎãĂĆä¡ăåŔŕèČ¡æČşåőđçŐřäÿĂäÿłäžěa
äÿŃéİćæŸŕäżčçăĄçđ'žä¿ŃïijŽ

```
class Node:
    def __init__(self, value):
        self._value = value
        self._children = []

    def __repr__(self):
        return 'Node({!r})'.format(self._value)

    def add_child(self, node):
        self._children.append(node)

    def __iter__(self):
        return iter(self._children)

    def depth_first(self):
        yield self
        for c in self:
            yield from c.depth_first()

# Example
if __name__ == '__main__':
    root = Node(0)
    child1 = Node(1)
    child2 = Node(2)
    root.add_child(child1)
    root.add_child(child2)
    child1.add_child(Node(3))
    child1.add_child(Node(4))
```

```
    child2.add_child(Node(5))

    for ch in root.depth_first():
        print(ch)
    # Outputs Node(0), Node(1), Node(3), Node(4), Node(2), Node(5)
```

åIJÍè£Źæőţäžčçă Ąäÿ■ïijŇdepth_first() æŰźæşŢçőĂå■ŢçŻť'èğĆãĂĆ åőČéęŰåĚĹè£ŤåŻđèĞłåůśæIJňèžńåźűè£■äžčæŕŔäÿĂäÿłå■ŘèŁĆçĆźåźű éĂŽè£ĞèřČçŤĺå■ŘèŁĆçĆźçŽĎ depth_first() æŰźæşŢ(ä¡£çŤĺ yield from èŕ■åŘě)è£ŤåŻđåŕźåžŤåĚČçť'ããĂĆ

èőĺèőž

PythonçŽĎè£■äžčå■Ŕèőőèę Ąæ śĆäÿĂäÿł __iter__() æŰźæşŢè£ŤåŻđäÿĂäÿłçL'źæőŁçŽĎè£■äžčåŹĺåŕźèśąïijŇ è£Źäÿłè£■äžčåŹĺåŕźèśąåőđçŐřäžĘ __next__() æŰźæşŢåźűéĂŽè£Ğ StopIteration åijĆåÿÿæăĞèŕĘè£■äžčçŽĎåőŇæĹŘãĂĆ ä¡ĘæŸŕïijŇåőđçŐřè£ŹäžŽéĂŽåÿÿäijŽæŕŤè¿ČçźĄçŘŘãĂĆ äÿŃéİćæĹŚäžňæijŤçď'žäÿŃè£Źçğ■æŰźåijŔïijŇå depth_first() æŰźæşŢïijŽ

```
class Node2:
    def __init__(self, value):
        self._value = value
        self._children = []

    def __repr__(self):
        return 'Node({!r})'.format(self._value)

    def add_child(self, node):
        self._children.append(node)

    def __iter__(self):
        return iter(self._children)

    def depth_first(self):
        return DepthFirstIterator(self)


class DepthFirstIterator(object):
    '''
    Depth-first traversal
    '''

    def __init__(self, start_node):
        self._node = start_node
        self._children_iter = None
        self._child_iter = None

    def __iter__(self):
        return self
```

```
    def __next__(self):
        # Return myself if just started; create an iterator for␣
↪children
        if self._children_iter is None:
            self._children_iter = iter(self._node)
            return self._node
        # If processing a child, return its next item
        elif self._child_iter:
            try:
                nextchild = next(self._child_iter)
                return nextchild
            except StopIteration:
                self._child_iter = None
                return next(self)
        # Advance to the next child and start its iteration
        else:
            self._child_iter = next(self._children_iter).depth_
↪first()
            return next(self)
```

DepthFirstIterator çśżåŠŇäÿŁéİćä¡£çŤÍçŤ§æĹŘåŹÍçŽĎçL'ĹæIJňåůěä¡IJåŐ§çŘĘçśżäijijïijŇ
ä¡ĘæŸŕåőČåĘŹèţůæİěå¿ĹçźAçŘŘïijŇåŻääÿžè£■äžčåŹÍå£ĚéążåIJÍè£■äžčåď'ĎçŘĘè£ĞçÍŃäÿ■çżt'æŁď'åď'ğéČ
åİęçŹ¡æİěèőšïijŇæšąäžžæĎ£æĎŔåĘŹè£ŹäźĹæŹęæűl'çŽĎäžčçăĄãĂĆåřĘä¡ăçŽĎè£■äžčåŹÍåőŽäźL'äÿžäÿĂäÿ

4.5 åŔ■åŘŚè£■äžč

éŮőécŸ

ä¡ăæČşåŔ■æŰźåŘŚè£■äžčäÿĂäÿłåžŔåĹŮ

èğčåĘşæŰźæąĹ

ä¡£çŤÍåĘĚç¡őçŽĎ `reversed()` åĞ¡æŢřïijŇæŕŤåęĆïijŽ

```
>>> a = [1, 2, 3, 4]
>>> for x in reversed(a):
...     print(x)
...
4
3
2
1
```

åŔ■åŘŚè£■äžčäžĚäžĚå¡ŞåŕźèśąçŽĎåď'ğåřŔåŔŕécĎåĚĹçąőåőŽæĹŰèĂĚåŕźèśąåőđçŐřäžĘ
__reversed__() çŽĎçL'źæőŁæŰźæşŢæŮűæL'■èČ¡çŤ§æŢĹãĂĆ
åęĆæđIJäÿď'èĂĚéČ¡äÿ■çňęåŘĹïijŇéĆčä¡åå£ĚéążåĚĹåřĘåŕźèśąè¡ňæ■ćäÿžäÿĂäÿłåĹŮèąĺæL'■èąŇïijŇæŕŤåęČ

```
# Print a file backwards
f = open('somefile')
for line in reversed(list(f)):
    print(line, end='')
```

èęĄæşÍæĎŘçŽĎæŸŕåęĆæđIJåŘŕè£■äžčåŕźèśąåĚČçť ăå¿Ĺåď ŽçŽĎèŕÍïijŇářĘåĚűéćĎåĚĹè¡ňæ■ćäÿžäÿĂ

èőÍèőž

å¿Ĺåď ŽçÍŃåžŘåŚŸåźűäÿ■ç§ěéĄŞåŘŕäżěéĂŽè£ĞåIJÍèĞłáőŽäźL’çśżäÿŁáőđçŐř
`__reversed__()` æŰźæşŢæÌěáőđçŐřåŘ■åŘŚè£■äžčãĂĆæŕŤåęĆïijŽ

```
class Countdown:
    def __init__(self, start):
        self.start = start

    # Forward iterator
    def __iter__(self):
        n = self.start
        while n > 0:
            yield n
            n -= 1

    # Reverse iterator
    def __reversed__(self):
        n = 1
        while n <= self.start:
            yield n
            n += 1

for rr in reversed(Countdown(30)):
    print(rr)
for rr in Countdown(30):
    print(rr)
```

áőŽäźL’äÿĂäÿłåŘ■åŘŚè£■äžčåŹĺåŘŕäżěä¡£å¿ŮäžčçăĄéÌđåÿÿçŽĎéńŸæŢĹïijŇ
åŻăäÿžáőČäÿ■åĘ■éIJĂèęĄářĘæŢřæ■őåąńåĚĚåĹřäÿĂäÿłåĹŮèąĺäÿ■çĎűåŘŐåĘ■åŐżåŘ■åŘŚè£■äžčè£ŹäÿłåĹ

4.6 åÿęælJL’åď ŰéČÍçŁűæĂĄçŽĎçŤ§æĹŘåŹĺåĞ¡æŢř

éŮőécŸ

ä¡ăæČşáőŽäźL’äÿĂäÿłçŤ§æĹŘåŹĺåĞ¡æŢřijÑä¡ĘæŸŕåőČäijŽèřČçŤÍæ§Řäÿłä¡ăæČşæŽť éIJšçżŹçŤÍæĹůă

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăæČşèőľ'ä¡ăçŽĎçŤ§æĹŘåŹĺæŽť'éIJšåď'ŰéČĺçŁűæĂĄçżŹçŤĺæĹůïijŇ
åĹńå£ŸäžĘä¡ăåŔŕäžěçőĂå■ŢçŽĎåřĘåőČåőđçŐřäÿžäÿĂäÿłçśžïijŇçĎűåŘŐæŁŁçŤ§æĹŘåŹĺåĞ¡æŢřæŤ¿åĹř
`__iter__()` æŰźæşŢäÿ■è£ĞåŐżãĂĆæŕŤåęĆïijŽ

```
from collections import deque

class linehistory:
    def __init__(self, lines, histlen=3):
        self.lines = lines
        self.history = deque(maxlen=histlen)

    def __iter__(self):
        for lineno, line in enumerate(self.lines, 1):
            self.history.append((lineno, line))
            yield line

    def clear(self):
        self.history.clear()
```

äÿžäžĘä¡£çŤĺè£ŹäÿłçśžïijŇä¡ăåŔŕäžěåřĘåőČå¡ŞåĄŽæŸŕäÿĂäÿłæŹőéĂŽçŽĎçŤ§æĹŘåŹĺåĞ¡æŢřãĂĆ
çĎűěĂŇïijŇçŤśäžŐåŔŕäžěåĹŻåžžäÿĂäÿłåőđä¿ŃåŕźèśąïijŇäžŐæŸŕä¡ăåŔŕäžěèő£éŮőåĘĚéČĺåšđæĂğåĂijïijŇ
æŕŤåęĆ `history` åšđæĂğæĹŰèĂĚæŸŕ `clear()` æŰźæşŢãĂĆäžčçăĄçď'žä¿ŃåęĆäÿŇïijŽ

```
with open('somefile.txt') as f:
    lines = linehistory(f)
    for line in lines:
        if 'python' in line:
            for lineno, hline in lines.history:
                print('{}:{}'.format(lineno, hline), end='')
```

èőĺèőž

åĚşäžŐçŤ§æĹŘåŹĺïijŇå¿ĹåőźæŸ§æŐL'è£ŻåĞ¡æŢřæŮăæL'Ăäÿ■èČ¡çŽĎéŹůéŸśãĂĆ
åęĆædIJçŤ§æĹŘåŹĺåĞ¡æŢřéIJĂèęĄèů§ä¡ăçŽĎçĺŃåžŘåĚűäžŰéČĺåĹĘæL'Şäžď'éĄŞçŽĎèŕĺ(æŕŤåęĆæŽť'éIJšă
åŘŕèČ¡äijŽåŕijèĞť'ä¡ăçŽĎäžčçăĄåijĆåÿÿçŽĎåď'■æĺĆãĂĆ åęĆædIJæŸŕè£ŹçĞ■æČĚåĘţçŽĎèŕĺïijŇåŘŕäžěèĂĆ
åIJĺ `__iter__()` æŰźæşŢäÿ■åőŽäźL'ä¡ăçŽĎçŤ§æĹŘåŹĺäÿ■äijŽæŤźåŔŸä¡ääżä¡ŢçŽĎçőŮæşŢéĂżè¿ŚãĂĆ
çŤśäžŐåőČæŸŕçśžçŽĎäÿĂéČĺåĹĘïijŇæL'ĂäžěåĚĄèőÿä¡ăåőŽäźL'åŘĎçĞ■åšđæĂğåŠŇæŰźæşŢæÌěä¿ŻçŤĺæĹ

äÿĂäÿłéIJĂèęĄæşĺæĎŔçŽĎåřŔåIJřæŰźæŸŕïijŇåęĆædIJä¡ăåIJĺè£■äžčæŞ■ä¡IJæŮűäÿ■ä¡£çŤĺforå¿łçŐŕè
`iter()` åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> f = open('somefile.txt')
>>> lines = linehistory(f)
>>> next(lines)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: 'linehistory' object is not an iterator
```

```
>>> # Call iter() first, then start iterating
>>> it = iter(lines)
>>> next(it)
'hello world\n'
>>> next(it)
'this is a test\n'
>>>
```

4.7 è£■äżčåŹĺåĹĞçL'Ğ

éŮőécŸ

ä¡ăæČşå¿ŮåĹřäÿĂäÿłçŤśè£■äżčåŹĺçŤ§æĹŘçŽĎåĹĞçL'ĞåŕźèśąïijŇä¡ĘæŸŕæăĞåĞĘåĹĞçL'ĞæŞ■ä¡IJåź

èğčåĘşæŰźæąĹ

åĞ¡æŢř `itertools.islice()` æ■čåě¡éĂĆçŤĺäžŐåIJĺè£■äżčåŹĺåŠŇçŤ§æĹŘåŹĺäÿŁåĄŽåĹĞçL'ĞæŞ

```
>>> def count(n):
...     while True:
...         yield n
...         n += 1
...
>>> c = count(0)
>>> c[10:20]
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: 'generator' object is not subscriptable

>>> # Now using islice()
>>> import itertools
>>> for x in itertools.islice(c, 10, 20):
...     print(x)
...
10
11
12
13
14
15
16
17
18
19
>>>
```

èőÍèőž

è£■äžčåŹÍåŠŇçŤ§æĹŘåŹÍäÿ■èČ¡ä¡£çŤÍæăĞåĞĘçŽĎåĹĞçL'ĞæŞ■ä¡IJïijŇåŻăäÿžåőČäžńçŽĎéŢ£åžęäžŇ
åĞ¡æŢř `islice()` è£ŤåŻđäÿĂäÿłåŘŕäżěçŤ§æĹŘæŇĞåőŽåĚČçť'ăçŽĎè£■äžčåŹÍïijŇåőČéĂŽè£ĞéĄ■åŐĘå
çĎűåŘŐæL'■åijĂåğŃäÿĂäÿłäÿłçŽĎè£ŤåŻđåĚČçť'ăïijŇåźűçŻť'åĹŕåĹĞçL'ĞçżŞæİ§çť'ćåijŢä¡■ç¡őãĂĆ

è£ŹéĞŇèęĄçİĂéĞ■åijžèřČçŽĎäÿĂçĆźæŸŕ `islice()`
äijŽæűĹèĂŮæŐL'äijăåĚěçŽĎè£■äżčåŹÍäÿ■çŽĎæŢřæ■őãĂĆ å£Ěéążè ĂČèŹŚåĹřè£■äżčåŹÍæŸŕäÿ■åŘréĂĘçŽ
æL'Ăäžěåę ĆæđIJä¡äéIJĂèęĄäźŃåŘŐåĘ■æňąèő£éŮőè£Źäÿłè£■äżčåŹÍçŽĎèŕİïijŇéĆčä¡ääřšå¿ŮåĚĹářĘåőČéĞ

4.8 èůşè£ĞåŘŕè£■äžčåŕźèšąçŽĎåijĂåğŃéČÍåĹĘ

éŮőécŸ

ä¡ăæČşéĄ■åŐĘäÿĂäÿłåŘŕè£■äžčåŕźèšąïijŇä¡ĘæŸŕåőČåijĂåğŃçŽĎæ§ŘäžŻåĚČçť'ăä¡ääźűäÿ■æĎ§åĚt'è

èğčåĘşæŰźæąĹ

`itertools` æĺąåİŮäÿ■æIJL'äÿĂäžŻåĞ¡æŢřåŘŕäżěåőŇæĹŘè£ŹäÿłäżżåŁąãĂĆ
éęŰåĚĹäżŃçż■çŽĎæŸŕ `itertools.dropwhile()` åĞ¡æŢřãĂĆä¡£çŤÍæŮűïijŇä¡ăçżŹåőČäijăéĂŠäÿĂäÿ
åőČäijŽè£ŤåŻđäÿĂäÿłè£■äžčåŹÍåŕźèšąïijŇäÿćåijČåŐ§æIJL'åžŘåĹŮäÿ■çŻť'åĹřåĞ¡æŢřè£ŤåŻđFlaseäźŃåL'■ç

äÿžäžĘæijŤçđ'žïijŇåĄĞåőŽä¡ăåIJÍèŕżåŘŰäÿĂäÿłåijĂåğŃéČÍåĹĘæŸŕåĞăèąŇæşÍéĞŁçŽĎæžŘæŰĞäžűã

```
>>> with open('/etc/passwd') as f:
...     for line in f:
...         print(line, end='')
...
##
# User Database
#
# Note that this file is consulted directly only when the system is␣
↪running
# in single-user mode. At other times, this information is provided␣
↪by
# Open Directory.
...
##
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
...
>>>
```

åęĆæđIJä¡ăæČşèůşè£ĞåijĂåğŃéČÍåĹĘçŽĎæşÍéĞŁèąŇçŽĎèŕİïijŇåŘŕäżěè£ŹæăůåĄŽïijŽ

```
>>> from itertools import dropwhile
>>> with open('/etc/passwd') as f:
...     for line in dropwhile(lambda line: line.startswith('#'), f):
```

```
...         print(line, end='')
...
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
...
>>>
```

è£Źäÿłä¿Ńå■ŘæŸŕå§žäžŐæäźæ■őæ§ŘäÿłæţŃèŕŢåĞ¡æŢřèůşè£ĞåijĂåğŃçŽĎåĚČçť˘ăãĂĆ åęĆæđIJä¡ăåůšçżŔæŸŐçąőç§ěéĄŞäžĘèęĄèůşè£ĞçŽĎåĚČçť˘ăçŽĎäÿłæŢřçŽĎèŕÍïijŇéĆčäźĹåŔŕäžěä¡£çŤÍ `itertools.islice()` æİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
>>> from itertools import islice
>>> items = ['a', 'b', 'c', 1, 4, 10, 15]
>>> for x in islice(items, 3, None):
...     print(x)
...
1
4
10
15
>>>
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇ `islice()` åĞ¡æŢřæIJĂåŘŐéĆčäÿł `None` åŔĆæŢřæŇĞåőŽäžĘä¡ăèęĄèŐůåŔŰäžŐçňň3äÿłåĹřæIJĂåŘŐçŽĎæL’ĂæIJL’åĚČçť˘äïijŇ åęĆæđIJ `None` åŠŇ3çŽĎä¡■ç¡őåŕźèřČïijŇæĎŔæĂİåřśæŸŕäžĚäžĚèŐůåŔŰåL’■äÿL’äÿłåĚČçť˘ăæĄřæĄřçŻÿåŔ (è£Źäÿłèů§åĹĞçL’ĞçŽĎçŻÿåŔ■æŞ■ä¡IJ `[3:]` åŠŇ `[:3]` åŐ§çŘĘæŸŕäÿĂæăůçŽĎ)ãĂĆ

èőÍèőž

åĞ¡æŢř `dropwhile()` åŠŇ `islice()` åĚűåőđåřśæŸŕäÿd’äÿłåÿőåŁl’åĞ¡æŢřïijŇäÿžçŽĎåřśæŸŕéĄ£ål

```
with open('/etc/passwd') as f:
    # Skip over initial comments
    while True:
        line = next(f, '')
        if not line.startswith('#'):
            break

    # Process remaining lines
    while line:
        # Replace with useful processing
        print(line, end='')
        line = next(f, None)
```

èůşè£ĞäÿĂäÿłåŔŕè£■äžčåŕźèśąçŽĎåijĂåğŃéČĺåĹĘèů§éĂŽåÿÿçŽĎè£ĞæżďæŸŕäÿ■åŘŇçŽĎãĂĆ æŕŤåęĆïijŇäÿŁè£řäžčçăĄçŽĎçňňäÿĂäÿłéČĺåĹĘåŔŕèČ¡äijŽè£ŹæăúéĞ■åĘŹïijŽ

```
with open('/etc/passwd') as f:
    lines = (line for line in f if not line.startswith('#'))
```

```
    for line in lines:
        print(line, end='')
```

è£Źæăůå ĘŹçąőåőđåŘŕäžěèůşè£ĞåijĂåğŃéČĺåĹĘçŽĎæşĺéĞŁèąŇïijŇä¡ĘæŸŕåŘŇæăůäź§äijŽèůşè£ĞæŰ
æ■ćåŘěèŕİèőšïijŇæĹŚäžňçŽĎèğčåĘşæŰźæąĹæŸŕäžĚäžĚèůşè£ĞåijĂåğŃéČĺåĹĘæżąèűşæţŃèŕŢæİąäžűçŽĎè

æIJĂåŘŐéIJĂèęĄçİĂéĞ■åijžèřČçŽĎäÿĂçĆźæŸŕïijŇæIJňèŁĆçŽĎæŰźæąĹéĂĆçŤĺäžŐæĽ’ĂæIJL’åŘŕè£
æŕŤåęĆçŤ§æĹŘåŹĺïijŇæŰĞäžűåŘŁåĚűçśžäijijçŽĎåŕźèśąãĂĆ

4.9 æŐŠåĹŮçżĎåŘĹçŽĎè£■äžč

éŮőécŸ

ä¡ăæČşè£■äžčéĄ■åŐĘäÿĂäÿłéŽĘåŘĹäÿ■åĚČçť ăçŽĎæĽ’ĂæIJL’åŘŕèČ¡çŽĎæŐŠåĹŮæĹŰçżĎåŘĹ

èğčåĘşæŰźæąĹ

itertoolsæĺąåİŮæŔŘä¿ŻäžĘäÿL’äÿłåĞ¡æŢřæİěèğčåĘşè£ŹçśźéŮőécŸãĂĆ
åĚűäÿ■äÿĂäÿłæŸŕ `itertools.permutations()` ïijŇ
åőČæŐěåŘŮäÿĂäÿłéŽĘåŘĹåźűäžğçŢ§äÿĂäÿłåĚČçżĎåžŘåĹŮïijŇæŕŔäÿłåĚČçżĎçŢśéŽĘåŘĹäÿ■æĽ’ĂæIJL’
äź§åřśæŸŕèŕť éĂŽè£ĞæĽ’Şäźśé ŽĘåŘĹäÿ■åĚČçť ăæŐŠåĹŮéąžåžŘçŢ§æĹŘäÿĂäÿłåĚČçżĎïijŇæŕŤåęĆïijŽ

```
>>> items = ['a', 'b', 'c']
>>> from itertools import permutations
>>> for p in permutations(items):
...     print(p)
...
('a', 'b', 'c')
('a', 'c', 'b')
('b', 'a', 'c')
('b', 'c', 'a')
('c', 'a', 'b')
('c', 'b', 'a')
>>>
```

åęĆæđIJä¡ăæČşå¿ŮåĹřæŇĞåőŽéŢ£åžęçŽĎæĽ’ĂæIJL’æŐŠåĹŮïijŇä¡ääŔŕäžěäijăéĂŠäÿĂäÿłåŘŕéĂL’çŽ

```
>>> for p in permutations(items, 2):
...     print(p)
...
('a', 'b')
('a', 'c')
('b', 'a')
('b', 'c')
('c', 'a')
('c', 'b')
>>>
```

ä¡£çŤĺ `itertools.combinations()` åŔŕå¿ŮåĹřè¿ŞåĚěéŻĘåŘĹäÿ■åĚČçť ăçŽĎæĽĂæIJĽçŽĎçżĽ

```
>>> from itertools import combinations
>>> for c in combinations(items, 3):
...     print(c)
...
('a', 'b', 'c')

>>> for c in combinations(items, 2):
...     print(c)
...
('a', 'b')
('a', 'c')
('b', 'c')

>>> for c in combinations(items, 1):
...     print(c)
...
('a',)
('b',)
('c',)
>>>
```

åŕźäžŐ `combinations()` æÍěèőšïijŇåĚČçť ăçŽĎéąžåžŔåůšçżŔäÿ■éĞ■èęĄäžĘãĂĆ äź§åřśæŸŕèŕťˇïijŇçżĎåŘĹ `('a', 'b')` èů§ `('b', 'a')` åĚűåőđæŸŕäÿĂæăůçŽĎ(æIJĂçżĹåŔłäijŽè¿ŞåĞžåĚűäÿ■äÿĂäÿł)ãĂĆ

åIJÍèőąçőŮçżĎåŘĹçŽĎæŮűåĂŹïijŇäÿĂæŮęåĚČçť ăèćńéĂĽˇåŔŰåřśäijŽäżŐåĂŹéĂĽˇäÿ■åĽˇŤéŹďˇæŐĽˇ èĂŇåĞ¡æŢř `itertools.combinations_with_replacement()` åĚĄèőÿåŘŇäÿĂäÿłåĚČçť ăèćńéĂĽˇæŃľåďˇŽæňąïijŇæŕŤåęĆïijŽ

```
>>> for c in combinations_with_replacement(items, 3):
...     print(c)
...
('a', 'a', 'a')
('a', 'a', 'b')
('a', 'a', 'c')
('a', 'b', 'b')
('a', 'b', 'c')
('a', 'c', 'c')
('b', 'b', 'b')
('b', 'b', 'c')
('b', 'c', 'c')
('c', 'c', 'c')
>>>
```

èőÍèőž

è£ŹäÿĂåřŔèŁĆæĹŚäżňåŘŚä¡ăåśŢçďˇžçŽĎäżĚäżĚæŸŕ `itertools` æÍąåİŮçŽĎäÿĂéČÍåĹĘåŁ§èČ¡ãĂĆ åř¡çőąä¡ăäź§åŔŕäžèèĞłåůśæĽˇŃåŁÍåőđçŐřæŐŠåĹŮçżĎåŘĹçőŮæşŢïijŇä

å¡ŞæĹŚäžňçćřåĹřçIJŃäÿŁåŐżæIJL'äžŻåď■æİĆçŽĎè£■äžčéŮőécŸæŮűïijŇæIJĂåě¡åŔŕäžěåĚĹåŐżçIJŃçIJŃi
åęĆæđIJè£ŹäÿłéŮőécŸå¿ĹæŹőéĄ■ïijŇéĆčäźĹå¿ĹæIJL'åŔŕèČ¡äijŽåIJĺéĞŇéİćæL'¿åĹřèğčåĘşæŰźæąĹïijĄ

4.10 åžŔåĹŮäÿŁçť'ćåijŢåĂijè£■äžč

éŮőécŸ

ä¡ăæČşåIJĺè£■äžčäÿĂäÿłåžŔåĹŮçŽĎåŘŇæŮűèů§èÿłæ■čåIJĺèćńåď'ĎçŘĘçŽĎåĚČçť'ăçť'ćåijŢãĂĆ

èğčåĘşæŰźæąĹ

åĘĚç¡őçŽĎ `enumerate()` åĞ¡æŢřåŔŕäžěå¿Ĺåě¡çŽĎèğčåĘşè£ŹäÿłéŮőécŸïijŽ

```
>>> my_list = ['a', 'b', 'c']
>>> for idx, val in enumerate(my_list):
...     print(idx, val)
...
0 a
1 b
2 c
```

äÿžäžĘæŇL'äijăçż§èąŇåŔůè¿ŞåĞž(èąŇåŔůäžŐ1åijĂåğŃ)ïijŇä¡ăåŔŕäžěäijăéĂŠäÿĂäÿłåijĂåğŃåŔĆæŢř

```
>>> my_list = ['a', 'b', 'c']
>>> for idx, val in enumerate(my_list, 1):
...     print(idx, val)
...
1 a
2 b
3 c
```

è£Źçğ■æČĚåĘţåIJĺä¡ăéĄ■åŐĘæŰĞäžűæŮűæČşåIJĺéŤŹèŕŕæűĹæĄŕäÿ■ä¡£çŤĺèąŇåŔůåőŽä¡■æŮűåĂŹéĺ

```
def parse_data(filename):
    with open(filename, 'rt') as f:
        for lineno, line in enumerate(f, 1):
            fields = line.split()
            try:
                count = int(fields[1])
                ...
            except ValueError as e:
                print('Line {}: Parse error: {}'.format(lineno, e))
```

`enumerate()` åŕźäžŐèů§èÿłæ§ŘäžŻåĂijåIJĺåĹŮèąĺäÿ■åĞžçŐřçŽĎä¡■ç¡őæŸŕå¿ĹæIJL'çŤĺçŽĎãĂĆ
æL'ĂäžëïijŇåęĆæđIJä¡ăæČşåřĘäÿĂäÿłæŰĞäžűäÿ■åĞžçŐřçŽĎå■Ţèŕ■æŸåřĎåĹřåőČåĞžçŐřçŽĎèąŇåŔůäÿ
`enumerate()` æİěåőŇæĹŘïijŽ

```
word_summary = defaultdict(list)

with open('myfile.txt', 'r') as f:
    lines = f.readlines()

for idx, line in enumerate(lines):
    # Create a list of words in current line
    words = [w.strip().lower() for w in line.split()]
    for word in words:
        word_summary[word].append(idx)
```

åęĆædIJä¡ăåd'ĎçŘĘåőŇæŰĞäżűåŘŐæL'Şå■ř word_summary ïijŇäijŽåŔŚçŐřåőČæŸŕäÿĂäÿłå■ŮåĚÿ(åĞEçąőæİèèőšæŸŕäÿĂäÿł defaultdict)ïijŇ åŕźäžŐæŕŔäÿłå■Ţèŕ■æIJL'äÿĂäÿł key ïijŇæŕŔäÿł key åŕźåžŢçŽĎåĂijæŸŕäÿĂäÿłçŢśè£Źäÿłå■Ţèŕ■åĞžçŐřè£Ğäÿd'æňąïijŇéĆčäźĹè£ŹäÿłèąŇåŔůżĎæĹŘçŽĎåĹŮèąĺãĂĆ åęĆædIJæ§Ŕäÿłå■Ţèŕ■åIJĺäÿĂèąŇäÿ■åĞžçŐřè£Ğäÿd'æňąïijŇéĆčäźĹè£ŹäÿłèąŇåŔůäź§äijŽåĞžçŐřäÿd'æňąi åŘŇæŮűäź§åŔŕäżěä¡IJäÿžæŰĞæIJńçŽĎäÿĂäÿłçőĂå■Ţçż§èőąãĂĆ

èőĺèőž

å¡Şä¡ăæČşéćİåd'ŮåőŽäźĽäÿĂäÿłèőąæŢřåŔŸéĞŔçŽĎæŮűåĂŹïijŇä¡£çŤĺ enumerate() åĞ¡æŢřäijŽæŻt'åŁăçőĂå■ŢãĂĆä¡ăåŔŕèČ¡äijŽåČŔäÿŃéİćè£ŹæăůåĘŹäżčçăĄïijŽ

```
lineno = 1
for line in f:
    # Process line
    ...
    lineno += 1
```

ä¡ĘæŸŕåęĆædIJä¡£çŤĺ enumerate() åĞ¡æŢřæİěäžčæŻ£åřśæŸ¿å¿ŮæŻt'åŁääijŸéŻĚäžĘïijŽ

```
for lineno, line in enumerate(f):
    # Process line
    ...
```

enumerate() åĞ¡æŢřè£ŤåŻđçŽĎæŸŕäÿĂäÿł enumerate åŕźèśąåőđä¿ŃïijŇ åőČæŸŕäÿĂäÿłè£■äžčåŹĺïijŇè£ŤåŻđè£đçż■çŽĎåŇĚåŘńäÿĂäÿłèőąæŢřåŠŇäÿĂäÿłåĂijçŽĎåĚČçżĎïijŇ åĚČçżĎäÿ■çŽĎåĂijéĂŽè£ĞåIJĺäijăåĚěåžŔåĹŮäÿŁèřČçŤĺ next() è£ŤåŻđãĂĆ

è£ŸæIJL'äÿĂçĆźåŔŕèČ¡åźűäÿ■å¿ĹéĞ■èęĄïijŇä¡ĘæŸŕäź§åĂijå¿ŮæşĺæĎŔïijŇ æIJL'æŮűåĂŹå¡Şä¡ăåIJĺäÿĂäÿłåůšçżŔèğčåŐŇåŘŐçŽĎåĚČçżĎåžŔåĹŮäÿŁä¡£çŤĺ enumerate() åĞ¡æŢřæŮűå¿Ĺåőźæ ŸŞèřČåĚěéŹůéŸśãĂĆ ä¡ăå¿ŮåČŔäÿŃéİćæ■čçąőçŽĎæŰźåijŔè£ŹæăůåĘŹïijŽ

```
data = [ (1, 2), (3, 4), (5, 6), (7, 8) ]

# Correct!
for n, (x, y) in enumerate(data):
    ...
```

```
# Error!
for n, x, y in enumerate(data):
    ...
```

4.11 åŘŇæŮűè£■äžčåď'Žäÿłåž'ŘåĹŮ

éŮőécŸ

ä¡ăæČşåŘŇæŮűè£■äžčåď'Žäÿłåž'ŘåĹŮïijŇærŔæňąåĹĘåĹńäżŐäÿĂäÿłåž'ŘåĹŮäÿ■åŔŰäÿĂäÿłåĚČçť'ăãĂ

èğčåĘşæŰźæąĹ

äÿžäžĘåŘŇæŮűè£■äžčåď'Žäÿłåž'ŘåĹŮïijŇä¡£çŤÍ `zip()` åĞ¡æŢřãĂĆærŤåęĆïijŽ

```
>>> xpts = [1, 5, 4, 2, 10, 7]
>>> ypts = [101, 78, 37, 15, 62, 99]
>>> for x, y in zip(xpts, ypts):
...     print(x,y)
...
1 101
5 78
4 37
2 15
10 62
7 99
>>>
```

`zip(a, b)` äijŽçŤ§æĹŘäÿĂäÿłåŔŕè£ŤåŻđåĚČçżĎ `(x, y)` çŽĎè£■äžčåŹĺïijŇåĚűäÿ■xæİěèĞłaïijŇyæİěèĞłbãĂĆ äÿĂæŮęåĚűäÿ■æ§Řäÿłåž'ŘåĹŮåĹřåžŢçżŞář¿ïijŇè£■äž
åŻăæ■ď'è£■äžčéŢ£åžęèů§åŘĆæŢřäÿ■æIJĂç§■åžŘåĹŮéŢ£åžęäÿĂèĞť'ãĂĆ

```
>>> a = [1, 2, 3]
>>> b = ['w', 'x', 'y', 'z']
>>> for i in zip(a,b):
...     print(i)
...
(1, 'w')
(2, 'x')
(3, 'y')
>>>
```

åęĆæđIJè£Źäÿłäÿ■æŸŕä¡ăæČşèęĄçŽĎæŢĹæđIJïijŇéĆčäźĹè£ŸåŔŕäžěä¡£çŤÍ `itertools.zip_longest()` åĞ¡æŢřæİěäžčæŻ£ãĂĆærŤåęĆïijŽ

```
>>> from itertools import zip_longest
>>> for i in zip_longest(a,b):
...     print(i)
```

```
...
(1, 'w')
(2, 'x')
(3, 'y')
(None, 'z')

>>> for i in zip_longest(a, b, fillvalue=0):
...     print(i)
...
(1, 'w')
(2, 'x')
(3, 'y')
(0, 'z')
>>>
```

èőÍèőž

å¡Şä¡äæČşæĹŘåŕźåď'ĎçŘĘæŢřæ■őçŽĎæŮűåĂŹ `zip()` åĞ¡æŢřæŸŕå¿ĹæIJL'çŤÍçŽĎãĂĆ æŕŤåęĆïijŇåĄĞèő¿ä¡ääď'ť'åĹŮèąĺåŠŇäÿĂäÿłåĂijåĹŮèąĺïijŇåřśåČŔäÿŇéIć

```
headers = ['name', 'shares', 'price']
values = ['ACME', 100, 490.1]
```

ä¡£çŤÍzip()åŔŕäžèèőľ'ä¡ääřĘåőČäžňæL'ŞåŇĚåźűçŤ§æĹŘäÿĂäÿłå■ŮåĚÿïijŽ

```
s = dict(zip(headers,values))
```

æĹŰèĂĚä¡äää§åŔŕäžěåČŔäÿŇéIćè£Źæăůäžğç Ť§è¿ŞåĞžïijŽ

```
for name, val in zip(headers, values):
    print(name, '=', val)
```

èŹ¡çĎűäÿ■åÿÿèğĄïijŇä¡ĘæŸŕ `zip()` åŔŕäžěæŐěåŔŮåď'ŽäžŐäÿď'äÿłçŽĎåžŔåĹŮçŽĎåŔĆæŢřãĂĆ è£ŹæŮűåĂŹæL'ĂçŤ§æĹŘçŽĎçżŞæđIJåĚČçżĎäÿ■åĚČçţ'ääÿłæŢřèů§è¿ŞåĚěåžŔåĹŮäÿłæŢřäÿĂæăůãĂĆæŕŤ

```
>>> a = [1, 2, 3]
>>> b = [10, 11, 12]
>>> c = ['x','y','z']
>>> for i in zip(a, b, c):
...     print(i)
...
(1, 10, 'x')
(2, 11, 'y')
(3, 12, 'z')
>>>
```

æIJĂåŘŐåijžèřČäÿĂçĆźåřśæŸŕïijŇ `zip()` äijŽåĹŻåżžäÿĂäÿłè£■äžčåŹĺæİěä¡IJäÿžçżŞæđIJè£ŤåŻđãĂĆ åęĆæđIJä¡ăéIJĂèęĄåřĘçżŞåŕźçŽĎåĂijå■ŸåĆĺåIJĺåĹŮèąĺäÿ■ïijŇèęĄä¡£çŤÍ `list()` åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> zip(a, b)
<zip object at 0x1007001b8>
>>> list(zip(a, b))
[(1, 10), (2, 11), (3, 12)]
>>>
```

4.12 äÿ■åŘŇéŻĘåŘĹäÿŁåĚČçťäçŽĎè£■äžč

éŮőécŸ

ä¡ăæČşåIJĺåď'ŽäÿłåŕźèśąæL'ğèąŇçŻÿåŘŇçŽĎæŞ■ä¡IJïijŇä¡ĘæŸŕè£ŹäžŻåŕźèśąåIJĺäÿ■åŘŇçŽĎåőźåŹĺä

èğčåĘşæŰźæąĹ

itertools.chain() æŰźæşŢåŔŕäžěçŤĺæİěçőĂåŇŰè£ŹäÿłäżżåŁąãĂĆ åőČæŐěåŘŮäÿĂäÿłåŔŕè£■äžčåŕźèśąåĹŮèąĺä¡IJäÿžè¿ŞåĚčïijŇåźűè£ŤåŻđäÿĂäÿłè£■äžčåŹĺïijŇæIJL'æŢĹçŽĎ äÿžäžĘæijŤçď'žæÿĚæěŽïijŇèĂČèŹŚäÿŃéİćè£ŹäÿłäżŃå■ŘïijŽ

```
>>> from itertools import chain
>>> a = [1, 2, 3, 4]
>>> b = ['x', 'y', 'z']
>>> for x in chain(a, b):
... print(x)
...
1
2
3
4
x
y
z
>>>
```

ä¡£çŤÍ chain() çŽĎäÿĂäÿłåÿÿèğĄåIJžæŹŕæŸŕå¡Şä¡ăæČşåŕźäÿ■åŘŇçŽĎéŻĘåŘĹäÿ■æL'ĂæIJL'åĚČç

```
# Various working sets of items
active_items = set()
inactive_items = set()

# Iterate over all items
for item in chain(active_items, inactive_items):
    # Process item
```

è£Źçğ■èğčåĘşæŰźæąĹèęĄæŕŤåČŔäÿŃéİćè£Źæăůä¡£çŤĺäÿď'äÿłå■ŢçŃňçŽĎå¿łçŐŕæŻť'åŁääijŸéŻĚïijŇ

```
for item in active_items:
    # Process item
```

```
    ...

for item in inactive_items:
    # Process item
    ...
```

èőÍèőž

itertools.chain() æŐěåŔŮäÿĂäÿłæĹŰåď'ŽäÿłåŔŕè£■äžčåŕźèśąæIJĂäÿžè¿ŞåĚěåŔĆæŢřãĂĆ çĎűåŘŐåĹŻåžžäÿĂäÿłè£■äžčåŹĺïijÑä¿İæňąè£đçż■çŽĎè£ŤåŻđæŕŔäÿłåŔŕè£■äžčåŕźèśąäÿ■çŽĎåĚČçt'ããĂĆ è£Źçğ■æŰźåijŔèęĄæŕŤåĚĹářĘåžŔåĹŮåŘĹåźűåĘ■è£■äžčèęĄéńŸæŢĹçŽĎåď'ŽãĂĆæŕŤåęĆïijŽ

```
# Inefficent
for x in a + b:
    ...

# Better
for x in chain(a, b):
    ...
```

çňňäÿĂçğ■æŰźæąĹäÿ■ïijÑ a + b æŞ■ä¡IJäijŽåĹŻåžžäÿĂäÿłåĚĺæŰřçŽĎåžŔåĹŮåźűèęĄæśĆaåŠŇbçŽ chian() äÿ■äijŽæIJL'è£ŹäÿĂæ■čïijÑæL'Ăäžěåę ĆæđIJè¿ŞåĚěåžŔåĹŮéİđåÿÿåď'ğçŽĎæŮűåĂŹäijŽå¿ĹçIJA åźűäÿŤå¡ŞåŔŕè£■äžčåŕźèśąçśžåđŃäÿ■äÿĂæăůçŽĎæŮűåĂŹ chain() åŘŇæăůåŔŕäžěå¿Ĺåě¡çŽĎåůěä¡IJãĂĆ

4.13 åĹŻåžžæŢřæ■őåď'ĎçŘĘçőąéĄŞ

éŮőécŸ

ä¡ăæČşäžěæŢřæ■őçőąéĄŞ(çśžäijijUnixçőąéĄŞ)çŽĎæŰźåijŔè£■äžčåď'ĎçŘĘæŢřæ■őãĂĆ ærŤåęĆïijÑä¡ăæIJL'äÿłåď'ğéĞŔçŽĎæŢřæ■őéIJĂèęĄåď'ĎçŘĘïijÑä¡ĘæŸŕäÿ■èČ¡årĘåőČäžñäÿĂæňąæĂğæŤ¿

èğčåĘşæŰźæąĹ

çŤ§æĹŘåŹĺåĞ¡æŢřæŸŕäÿĂäÿłåőđçŐřçőąéĄŞæIJžåĹűçŽĎåě¡åŁđæşŢãĂĆ äÿžäžĘæijŤçd'žïijÑåĄĞåőŽä¡ăèęĄåď'ĎçŘĘäÿĂäÿłéİđåÿÿåď'ğçŽĎæŮěå£ŮæŰĞäžűçŻőå¡ŢïijŽ

```
foo/
    access-log-012007.gz
    access-log-022007.gz
    access-log-032007.gz
    ...
    access-log-012008
bar/
    access-log-092007.bz2
```

```
    ...
    access-log-022008
```

åĄĞèő¿æŕŔäÿłæŮěå£ŮæŰĞäżűåŇĚåŘńè£ŹæăůçŽĎæŢřæ■őïijŽ

```
124.115.6.12 - - [10/Jul/2012:00:18:50 -0500] "GET /robots.txt ..."␣
↪200 71
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /ply/ ..." 200␣
↪11875
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /favicon.ico ..
↪." 404 369
61.135.216.105 - - [10/Jul/2012:00:20:04 -0500] "GET /blog/atom.xml␣
↪..." 304 -
...
```

äÿžäžĘåď ĎçŘĘè£ŹäžŻæŰĞäżűïijŇä¡åŔŕäżěåőŽäźĽäÿĂäÿłçŤšåď ŽäÿłæĽğèąŇçĽźåőŽäżżåŁąçŃňçńŇ

```
import os
import fnmatch
import gzip
import bz2
import re

def gen_find(filepat, top):
    '''
    Find all filenames in a directory tree that match a shell␣
↪wildcard pattern
    '''
    for path, dirlist, filelist in os.walk(top):
        for name in fnmatch.filter(filelist, filepat):
            yield os.path.join(path,name)

def gen_opener(filenames):
    '''
    Open a sequence of filenames one at a time producing a file␣
↪object.
    The file is closed immediately when proceeding to the next␣
↪iteration.
    '''
    for filename in filenames:
        if filename.endswith('.gz'):
            f = gzip.open(filename, 'rt')
        elif filename.endswith('.bz2'):
            f = bz2.open(filename, 'rt')
        else:
            f = open(filename, 'rt')
        yield f
        f.close()

def gen_concatenate(iterators):
    '''
```

```
    Chain a sequence of iterators together into a single sequence.
    '''
    for it in iterators:
        yield from it

def gen_grep(pattern, lines):
    '''
    Look for a regex pattern in a sequence of lines
    '''
    pat = re.compile(pattern)
    for line in lines:
        if pat.search(line):
            yield line
```

çŐřåIJÍä¡ăåŔŕäžěå¿Ĺåőźæ˙ŸŞçŽĎåřĘè£ŹäžŻåĞ¡æŢřè£đèţůæİěåĹŻåžžäÿĂäÿłåď˙ĎçŘĘçőąéĄŞãĂĆ
æŕŤåęĆïijÑäÿžäžĘæ§ěæL’¿åŇĚåŘńå■Ţèŕ■pythonçŽĎæL’ĂæIJL’æŮěå£ÛèąÑïijÑä¡ăåŔŕäžěè£ŹæăůåĄŽïijŽ

```
lognames = gen_find('access-log*', 'www')
files = gen_opener(lognames)
lines = gen_concatenate(files)
pylines = gen_grep('(?i)python', lines)
for line in pylines:
    print(line)
```

åęĆæđIJåřĘæİěçŽĎæŮűåĂŹä¡ăæČşæL’l’åśŢçőąéĄŞïijÑä¡ăçŤŽèĞşåŔŕäžěåIJÍçŤ§æĹŘåŹÍèąÍè¿¿åijŔäÿ■
æŕŤåęĆïijÑäÿÑéÍćè£ŹäÿłçL’ĹæIJňèőąçőŮåĞžäijăè¿ŞçŽĎå■ÛèŁĆæŢřåźűèőąçőŮåĚűæĂżåŠÑãĂĆ

```
lognames = gen_find('access-log*', 'www')
files = gen_opener(lognames)
lines = gen_concatenate(files)
pylines = gen_grep('(?i)python', lines)
bytecolumn = (line.rsplit(None,1)[1] for line in pylines)
bytes = (int(x) for x in bytecolumn if x != '-')
print('Total', sum(bytes))
```

èőÍèőž

äžěçőąéĄŞæŰźåijŔåď˙ĎçŘĘæŢřæ■őåŔŕäžěçŤÍæİěèğčåĘşåŘĎçśżåĚűäžŰéŮőéćŸïijÑåŇĚæŃňèğčæđŘïi

äÿžäžĘçŘĘèğčäÿŁè£řäžčçăĄïijÑéĞ■çĆźæŸŕèęĄæŸŐçŹ¡ `yield`
èŕ■åŘëä¡IJäÿžæŢřæ■őçŽĎçŤ§äžğèĂĚèĂŇ `for` å¿łçŐŕèŕ■åŘëä¡IJäÿžæŢřæ■őçŽĎæűĹèť´źèĂĚãĂĆ
å¡Şè£ŹäžŻçŤ§æĹŘåŹÍèćńè£đåIJÍäÿĂèţůåŘŐïijÑærŔäÿł `yield`
äijŽåřĘäÿĂäÿłå■ŢçŃňçŽĎæŢřæ■őåĚČçť´ăäijăéĂŠçżŹè£■äžčåď˙ĎçŘĘçőąéĄŞçŽĎäÿÑäÿĂéŸűæőţãĂĆ
åIJÍä¿Ńå■ŘæIJĂåŘŐéČÍåĹĘïijÑ `sum()` åĞ¡æŢřæŸŕæIJĂçżĹçŽĎçÍÑåžŔél’šåŁÍèĂĚïijÑærŔæąäžŐçŤ§æĹŘ

è£Źçğ■æŰźåijŔäÿĂäÿłéIđåÿÿåě¡çŽĎçL’źçĆźæŸŕærŔäÿłçŤ§æĹŘåŹÍåĞ¡æŢřå¿ĹåřŔåźűäÿŤéČ¡æŸŕçŃňç
å¿Ĺåď˙ŽæŮűåĂŹïijÑè£ŹäžŻåĞ¡æŢřåęĆæđIJærŕŤè¿ČéĂŽçŤÍçŽĎèŕÍåŔŕäžěåIJÍåĚűäžŰåIJžæŹŕéĞ■åď˙■ä¡£çŤÍ
åźűäÿŤæIJĂçżĹåřĘè£ŹäžŻçżĎäžűçżĎåŘÍèţůæİěçŽĎäžčçăĄçIJŃäÿŁåŐżéIđåÿÿçőĂå■ŢïijÑäź§å¿Ĺåőźæ˙ŸŞç

ä¡£çŤÍè£ŹçğæŰźåijŔçŽĎåĘĚå■ŸæŢĹçŐĞäź§äÿ■å¿Ůäÿ■æŔŘãĂĆäÿŁè£řäžčçăĄå■şä¿£æŸŕåIJÍäÿĂäÿ
äžŇåőđäÿŁïijŇçŤšäžŐä¡£çŤÍäžĘè£■äžčæŰźåijŔåď ĎçŘĘïijŇäžčçăĄè£ŘèąŇè£ĞçÍŇäÿ■åŔłéIJĂèęĄå¿ĹářŔå

åIJÍèřČçŤÍ `gen_concatenate()` åĞ¡æŢřçŽĎæŮűåĂŹä¡ääŔŕèČ¡äijŽæIJL'äžŽäÿ■åď łæŸŐçŹ¡ãĂĆ
è£ŹäÿłåĞ¡æŢřçŽĎçŽőçŽĎæŸŕářĘè¿ŞåĚěåžŔåĹŮæŃijæŐěæĹŘäÿĂäÿłå¿ĹéŢ£çŽĎèąŇåžŔåĹŮãĂĆ
`itertools.chain()` åĞ¡æŢřåŘŇæăůæIJL'çśżäijijçŽĎåŁ§èČ¡ïijŇä¡ĘæŸŕåőČéIJĂèęĄářĘæL'ĂæIJL'åŔŕè
åIJÍäÿŁéİćè£ŹäÿłäżŃå■Řäÿ■ïijŇä¡ääŔŕèČ¡äijŽåĘŹçśżäijijè£ŹæăůçŽĎèŕ■åŔě
`lines = itertools.chain(*files)` ïijŇ ä¡£å¿Ů `gen_opener()`
çŤ§æĹŘåŹÍèČ¡èćńåĚÍéČÍæűĹèť´źæŐL'ãĂĆ ä¡ĘçŤšäžŐ `gen_opener()`
çŤ§æĹŘåŹÍæŕŘæňąçŤ§æĹŘäÿĂäÿłæL'ŞåijĂè£ĞçŽĎæŰĞäżűïijŇ
ç■L'åĹřäÿŃäÿĂäÿłè£■äžčæ■ěéłď´æŮűæŰĞäżűåřšåĚşéŮ■äžĘïijŇåŽăæ■ď `chain()`
åIJÍè£ŹéĞŇäÿ■èČ¡è£Źæăůä¡£çŤÍãĂĆ äÿŁéİćçŽĎæŰźæąĹåŔŕäžěéĄ£åĚ■è£Źçğ■æČĚåĘţãĂĆ

`gen_concatenate()` åĞ¡æŢřäÿ■åĞžçŐřè£Ğ `yield from` èŕ■åŔěïijŇåőČářĘ
`yield` æŞ■ä¡IJäžčçŘĘåĹřçĹűçŤ§æĹŘåŹÍäÿŁåŐžãĂĆ èŕ■åŔě `yield from`
`it` çőĂå■ŢçŽĎè£ŤåŽđçŤ§æĹŘåŹÍ `it` æL'Ăäžğç Ť§çŽĎæL'ĂæIJL'åĂijãĂĆ
åĚşäžŐè£ŹäÿłæĹŚäžňåIJÍ4.14ářŔèŁĆäijŽæIJL'æŻť´è£ŽäÿĂæ■ěçŽĎæŔŘè£řãĂĆ

æIJĂåŘŐè£ŸæIJL'äÿĂçĆźéIJĂèęĄæşÍæĎŘçŽĎæŸŕïijŇçőąéĄŞæŰźåijŔåźűäÿ■æŸŕäÿĞèČ¡çŽĎãĂĆ
æIJL'æŮűåĂŹä¡ăæČşçńÑå■şåď ĎçŘĘæL'ĂæIJL'æŢřæ■őãĂĆ çĎűèĂŇïijŇå■şä¿£æŸŕè£Źçğ■æČĚåĘţïijŇä¡£ç

David Beazley åIJÍäžŰçŽĎ Generator Tricks for Systems Programmers
æŢŹçÍŃäÿ■åŕźäžŐè£Źçğ■æŁĂæIJŕæIJL'éİđåÿÿæůśåĚěçŽĎèőšèğčãĂĆåŔŕäžěåŘĆèĂČè£ŹäÿłæŢŹçÍŃèŐůåŔ

4.14 åśŢåijĂåţŇåěŮçŽĎåžŔåĹŮ

éŮőécŸ

ä¡ăæČşåřĘäÿĂäÿłåď ŽåśĆåţŇåěŮçŽĎåžŔåĹŮåśŢåijĂæĹŘäÿĂäÿłå■ŢåśĆåĹŮèąÍ

èğčåĘşæŰźæąĹ

åŔŕäžěåĘŹäÿĂäÿłåŇĚåŘń `yield from` èŕ■åŔěçŽĎéĂŠå¡ŠçŤ§æĹŘåŹÍæİěè¡żæİ¿èğčåĘşè£ŹäÿłéŮőéć

```
from collections import Iterable

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_
↪types):
            yield from flatten(x)
        else:
            yield x

items = [1, 2, [3, 4, [5, 6], 7], 8]
# Produces 1 2 3 4 5 6 7 8
for x in flatten(items):
    print(x)
```

åIJÍäÿŁéİćäžčçăĄäÿ■ïijŇ `isinstance(x, Iterable)` æčĂæ§ěæ§ŘäÿłåĚČçť ăæŸŕåŘęæŸŕåŘŕè£■äžčçŽĎãĂĆ åęĆæđIJæŸŕçŽĎèŕİïijŇ `yield from` åřśäijŽè£ŤåŽđæĽĂæIJL'å■Řä¿ŇçÍŇçŽĎåĂijãĂĆæIJĂçżĹè£ŤåŽđçżŞæđIJåřśæŸŕäÿĂäÿłæšąæIJL'åţŇå

éćİåď'ŰçŽĎåŔĆæŢř `ignore_types` åŠŇæčĂæţŃèŕ■åŘě `isinstance(x, ignore_types)` çŤÍæİěåřĘå■Ůçňęäÿšå ŠŇå■ŮèŁĆæŐŠéŹď'åIJÍåŘŕè£■äžčåřźèšąåď'ŰïijŇéŸšæ■ćåřĘåőČäz è£ŹæăůçŽĎèŕİå■ŮçňęäÿšæŢřçżĎåřśèČ¡æIJĂçźĹè£ŤåŽđæĹŚäžňæĽĂæIJ§æIJŻçŽĎçżŞæđIJäžĘãĂĆæŕŤåęĆ

```
>>> items = ['Dave', 'Paula', ['Thomas', 'Lewis']]
>>> for x in flatten(items):
...     print(x)
...
Dave
Paula
Thomas
Lewis
>>>
```

èőÍèőž

èŕ■åŘě `yield from` åIJÍä¡ăæČşåIJÍçŤ§æĹŘåŹÍäÿ■èřČçŤÍåĚűäžŰçŤ§æĹŘåŹÍä¡IJäÿžå■Řä¿ŇçÍŇçŽĎa åęĆæđIJä¡ääÿ■ä¡£çŤÍåőČçŽĎèŕİïijŇéĆčäźĹåřśå£Ěéąžå ĘŹéćİåď'ŰçŽĎ `for` å¿łçŐŕäžĘãĂĆæŕŤåęĆïijŽ

```
def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_
↪types):
            for i in flatten(x):
                yield i
        else:
            yield x
```

åř¡çőąåŘłæŤźäžĘäÿĂçĆźçĆźïijŇä¡ĘæŸŕ `yield from` èŕ■åŘěçIJŇäÿŁåŐżæĎ§èğL'æŻť'åě¡ïijŇåźűäÿŤäź§ä¡£å¿ŮäžčçăĄæŻť'çőĂæť'Ąæÿ ĚçĹ¡ãĂĆ

äźŇåL'■æŘŘåĹřçŽĎårźäžŐå■Ůçňęäÿšå ŠŇå■ŮèŁĆçŽĎéćİåď'ŰæčĂæ§ěæŸŕäÿžäžĘéŸšæ■ćåřĘåőČäžňå åęĆæđIJè£ŸæIJL'åĚűäžŰä¡ääÿ■æČşåśŢåijĂçŽĎçśžåđŇïijŇä£őæŤźåŔĆæŢř `ignore_types` å■şåŔŕãĂĆ

æIJĂåŘŐèęĄæşĺæĎŘçŽĎäÿĂçĆźæŸŕïijŇ `yield from` åIJÍæűL'åŘŁåĹřå§žäžŐå■ŘçÍŇåŠŇçŤ§æĹŘåŹÍçŽĎåźűåŘŠçijŰçÍŇäÿ■æL'őæijŤçİĂæŻť'åŁáéĞ■èęĄçŽĎèğŠè åŔŕäžěåŔĆèĂČ12.12åřŘèŁĆæ§ěçIJŇåŘęåď'ŰäÿĂäÿłä¿Ňå■ŘãĂĆ

4.15 éąžäžŔè£■äžčåŘĹåźűåŘŐçŽĎæŐŠäžŔè£■äžčåŕźèšą

éŮőécŸ

ä¡ăæIJL'äÿĂçşžåĹŮæŐŠäžŔäžŔåĹŮïijŇæČşåřĘåőČäžňåŘĹåźűåŘŐå¿ŮåĹřäÿĂäÿłæŐŠäžŔäžŔåĹŮåźűå

èğčåĘşæŰźæąĹ

heapq.merge() åĞ¡æŢřåŔŕäžěåÿőä¡ăèğčåĘşè£ŹäÿłéŮőécŸãĂĆæŕŤåęĆïijŽ

```
>>> import heapq
>>> a = [1, 4, 7, 10]
>>> b = [2, 5, 6, 11]
>>> for c in heapq.merge(a, b):
...     print(c)
...
1
2
4
5
6
7
10
11
```

èőÍèőž

heapq.merge åŔŕè£■äžčçL'źæĂğæĎŔåŠşçİĂåőČäÿ■äijŽçńÑél'ňèřžåŔŰæL'ĂæIJL'åžŔåĹŮãĂĆè£ŹåřśæĎŔåŠşçİĂä¡ăåŔŕäžěåIJÍéİđåÿÿéŢ£çŽĎåžŔåĹŮäÿ■ä¡£çŤĺåőČïijŇèĂŇäÿ■äijŽæIJL'åď'łåď'ğçŽĎåijÄéæŕŤåęĆïijŇäÿÑéİćæŸŕäÿĂäÿłä¿Ńå■ŘæİěæijŤçď'žåęĆä¡ŢåŘĹåźűäÿď'äÿłæŐŠåžŔæŰĞäžűïijŽ

```
with open('sorted_file_1', 'rt') as file1, \
    open('sorted_file_2', 'rt') as file2, \
    open('merged_file', 'wt') as outf:

    for line in heapq.merge(file1, file2):
        outf.write(line)
```

æIJL'äÿĂçĆźèęĄåijžěřČçŽĎæŸŕ heapq.merge() éIJĂèęĄæL'ĂæIJL'è¿ŞåĚěåžŔåĹŮå£Ěéążæ ŸŕæŐŠçL'źåĹńçŽĎïijŇåőČåźűäÿ■äijŽécĎåĚĹèŕžåŘŰæL'ĂæIJL'æŢřæ■őåĹřåäĘæăĹäÿ■æĹŰèĂĚécĎåĚĹæŐŠåžŔïijåőČäžĚäżĚæŸŕæčĂæ§ěæL'ĂæIJL'åžŔåĹŮçŽĎåijĂäğŃéČĺåĹĘåźűè£ŤåŻđæIJĂårŔçŽĎéĆčäÿłïijŃè£ŹäÿłèŞČ

4.16 è£■äžčåŹĺäžčæŻ£whileæŮäéŹŘå¿łçŐŕ

éŮőécŸ

ä¡ăåIJĺäžčçăĄäÿ■ä¡£çŤĺ while å¿łçŐŕæİěè£■äžčåď'ĎçŘĘæŢřæ■őïijŇåŻääÿžåőČéIJĂèęĄèřĆçŤĺæ§ŘäÿèČ¡äÿ■èČ¡çŤĺè£■äžčåŹĺæİěéĞ■åĘŹè£ŹäÿłåżłçŐŕåŚćïij§

èğčåĘşæŰźæąĹ

äÿĂäÿłåÿÿèğĄçŽĎIOæŞ■ä¡IJçÍŃåžŔåŔŕèČ¡äijŽæČşäÿŃéİćè£ŹæăůïijŽ

```
CHUNKSIZE = 8192

def reader(s):
    while True:
        data = s.recv(CHUNKSIZE)
        if data == b'':
            break
        process_data(data)
```

è£Źçğ■äžčçăĄéĂŽåÿÿåŔŕäžěä¡£çŤÍ `iter()` æİěäžčæŻ£ïijŇåęĆäÿŇæĽĂçď žïijŽ

```
def reader2(s):
    for chunk in iter(lambda: s.recv(CHUNKSIZE), b''):
        pass
        # process_data(data)
```

åęĆæđIJä¡ăæĂĂçŰŠåőČåĹŕåžŢèČ¡äÿ■èČ¡æ■čåÿÿåůěä¡IJïijŇåŔŕäžěèŕŢéłŇäÿŇäÿĂäÿłçőÅå■ŢçŽĎä¿Ňå

```
>>> import sys
>>> f = open('/etc/passwd')
>>> for chunk in iter(lambda: f.read(10), ''):
...     n = sys.stdout.write(chunk)
...
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
daemon:*:1:1:System Services:/var/root:/usr/bin/false
_uucp:*:4:4:Unix to Unix Copy Protocol:/var/spool/uucp:/usr/sbin/
 ↪uucico
...
>>>
```

èőĺèőž

`iter` åĞ¡æŢřäÿĂäÿłéšIJäÿžäžžçş§ěçŽĎçĽźæĂğæŸŕåőČæŐěåŔŮäÿĂäÿłåŔŕéĂĽ'çŽĎ `callable` åŕźèśąåŠŇäÿĂäÿłæăĞèőř(çżŞář¿)åĂijä¡IJäÿžè¿ŞåĚěåŔĆæŢřãĂĆ å¡Şäžěè£Źçğ■æŰźåijŔä¡£çŤĺçŽĎæŮűåĂŹïijŇåőČäijŽåĹŻåžžäÿĂäÿłè£■äžčåŹĺïijŇ è£Źäÿłè£■äžčåŹĺäijŽäÿ■æŰ■èřČçŤÍ `callable` åŕźèśąçŻt'åĹřè£ŤåŻđåĂijåŠŇæăĞèőřåĂijçŻÿç■L'äÿžæ■ćãĂ

è£Źçğ■çL'źæőŁçŽĎæŰźæşŢåŕźäžŐäÿĂäžŻçL'źåőŽçŽĎäijŽèćńéĞ■åď'■èřČçŤĺçŽĎåĞ¡æŢřå¿ĹæIJL'æŢĺ äÿ¿ä¿ŃæİèèőšïijŇåęĆæđIJä¡ăæČşäžŐåěŮæŐěå■ŮæĹŰæŰĞäžűäÿ■äžěæŢřæ■őåİŮçŽĎæŰźåijŔèŕżåŔŰæŢřæ `read()` æĹŰ `recv()` ïijŇ åźűåIJĺåŘŐéİćçt'ğèů§äÿĂäÿłæŰĞäžűçżŞář¿æţŃèŕŢæİěåĘşåőŽæŸŕåŘęçżĹæ■ćãĂ `iter()` èřČçŤĺåršåŔŕäžěårĘäÿd'èĂĚçżŞåŘĹètůæİěäžĘãĂĆ åĚűäÿ■ `lambda` åĞ¡æŢřåŔĆæŢřæŸŕäÿžäžžĘåĹŻåžžäÿĂäÿłæŮăåŔĆçŽĎ `callable` åŕźèśąïijŇåźűäÿž `recv` æĹŰ `read()` æŰźæşŢæŘŘä¿ŻäžĘ `size` åŔĆæŢřãĂĆ

çňňäžŤçńäïijŽæŰĞäżűäÿŐIO

æĽĂæIJL’çÍŃåžŔéČ¡èęĄåď’ĎçŘĘè¿ŞåĚěåŠŇè¿ŞåĞžãĂĆ
è£ŹäÿĂçńäářĘæűţçŻŰåď’ĎçŘĘäÿ■åŘŇçśžåđŃçŽĎæŰĞäżűïijŇåŇĚæŃňæŰĞæIJňåŠŇäžŇè£ŻåĹűæŰĞäżűï
åŕźæŰĞäżűåŘ■åŠŇçŻőå¡ŢçŽĎæŞ■ä¡IJäź§äijŽæűĽ’åŔŁåĹřãĂĆ

Contents:

5.1 èŕżåĘŹæŰĞælJňæŢřæ■ő

éŮőécŸ

ä¡ăéIJĂèęĄèŕżåĘŹåŘĎçğ■äÿ■åŘŇçijŰçăĄçŽĎæŰĞæIJňæŢřæ■őïijŇærŤåęĆASCIIïijŇUTF-8æĹŰUTF-16çijŰçăĄç■L’ãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤĺåÿęæIJL’ `rt` æĺąåijŔçŽĎ `open()` åĞ¡æŢřèŕżåŔŰæŰĞæIJňæŰĞäżűãĂĆåęĆäÿŇæL’Ăçď’žïijŽ

```
# Read the entire file as a single string
with open('somefile.txt', 'rt') as f:
    data = f.read()

# Iterate over the lines of the file
with open('somefile.txt', 'rt') as f:
    for line in f:
        # process line
        ...
```

çśżäijijçŽĎïijŇäÿžäžĘåĘŹåĚěäÿĂäÿłæŰĞæIJňæŰĞäżűïijŇä¡£çŤĺåÿęæIJL’ `wt` æĺąåijŔçŽĎ `open()` åĞ¡æŢřïijŇ åęĆædIJäźŇåL’■æŰĞäżűåĘĚåőźå■ŸåIJĺåĹŹæÿĚéŹď’åźűèęĘçŻŰæŐL’ãĂĆ

```
# Write chunks of text data
with open('somefile.txt', 'wt') as f:
    f.write(text1)
    f.write(text2)
    ...

# Redirected print statement
with open('somefile.txt', 'wt') as f:
    print(line1, file=f)
    print(line2, file=f)
    ...
```

åęĆædIJæŸŕåIJĺåůšå■ŸåIJĺæŰĞäżűäÿ■æůżåŁääĘĚåőźïijŇä¡£çŤĺæĺąåijŔäÿž `at` çŽĎ `open()` åĞ¡æŢřãĂĆ

æŰĞäžűçŽĎèřžåĘŹæŞ■ä¡IJéżŸèőďä¡£çŤĺçşżçż§çijŰçăĄïijŇåŔřäžěéĂŽè£ĞèřČçŤĺ sys.getdefaultencoding() æİěå¿ŮåĹřãĂĆ åIJĺåď ğåď ŽæŢřæIJžåŹĺäÿŁéİćéČ¡æŸŕutf-8çijŰçăĄãĂĆåęĆæđIJä¡ăåůšçżŔç§ěéĄŞä¡ăèęĄèřžåĘŹçŽĎæŰĞæIJňæŸŕåĚűäžŰçijŰçăĄæŰźåijŔïijŇéĆčäźĹåŔřäžěéĂŽè£ĞäijáéĂŠäÿĂäÿłåŔřéĂL'çŽĎ encoding åŔĆæŢřçżŹopen()åĞ¡æŢřãĂĆåęĆäÿŇæL'Ăçď žïijŽ

```
with open('somefile.txt', 'rt', encoding='latin-1') as f:
    ...
```

PythonæŤŕæŇĄéİđåÿÿåď ŽçŽĎæŰĞæIJňçijŰçăĄãĂĆåĞääÿłåÿÿèğĄçŽĎçijŰçăĄæŸŕascii, latin-1, utf-8åŠŇutf-16ãĂĆ åIJĺwebåžŤçŤĺçÍŇåžŔäÿ■éĂŽåÿÿéČ¡ä¡£çŤĺçŽĎæŸŕUTF-8ãĂĆ asciiåŕźåžŤäžŐU+0000åĹřU+007FèŇČåŻť åĘĚçŽĎ7ä¡■å■Ůçňęã ĂĆ latin-1æŸŕå■ŮèŁĆ0-255åĹřU+0000èĞşU+00FFèŇČåŻť åĘĚUnicodeå■Ůçňęç ŽĎçŻť æŐěæŸäåřĎãĂĆ å¡Şèŕźå ŔŰäÿĂäÿłæIJłç§ěçijŰçăĄçŽĎæŰĞæIJňæŮűä¡£çŤĺlatin-1çijŰçăĄæřÿè£IJäÿ■äijŽäžğçŤ§èğčçăĄéŤŹèŕŕãĂĆ ä¡£çŤĺlatin-1çijŰçăĄèŕźåŔŰäÿĂäÿłæŰĞäžűçŽĎæŮůåŹäź§èőÿäÿ■èČ¡äžğçŤ§åőŇåĚĺæ■čçąőçŽĎæŰĞæIJňèğčçăĄæŢřä¡ĘæŸŕåőČäź§èČ¡äžŐäÿ■æŔŘåŔŰåĞžèűşåď §åď ŽçŽĎæIJL'çŤĺæŢřæ■őãĂĆåŘŇæŮűïijŇåęĆæđIJä¡ääźŇåŘ

èőĺèőž

èŕźåĘŹæŰĞæIJňæŰĞäžűäÿĂèĹňæİèèőšæŸŕæŕŤè¿ČçőĂå■ŢçŽĎãĂĆä¡ĘæŸŕäź§åĞäçĆźæŸŕéIJĂèęĄæşéęŰåĚĹïijŇåIJĺä¿Ńå■ŘçĺŇåžŔäÿ■çŽĎwithèŕ■åŔężŹèćńä¡£çŤĺåĹřçŽĎæŰĞäžűåĹŻåžžäžĘäÿĂäÿłäÿŁäÿŇæä¡Ę with æŐğåĹűåİÙçżŞæİ§æŮűïijŇæŰĞäžűäijŽèĞłåŁĺåĚşéŮ■ãĂĆä¡ääź§åŔŕäžěäÿ■ä¡£çŤĺ with èŕ■åŔěïijŇä¡ĘæŸŕè£ŹæŮűåĂŹä¡ääřšå£Ěéąžèőřå¿ŮæĽŃåŁĺåĚşéŮ■æŰĞäžűïijŽ

```
f = open('somefile.txt', 'rt')
data = f.read()
f.close()
```

åŔęåď ŰäÿĂäÿłéŮőécŸæŸŕåĚşäžŐæ■ćèąŇçňęçŽĎèŕĘåĹńéŮőécŸïijŇåIJĺUnixåŠŇWindowsäÿ■æŸŕäÿ■éżŸèőď æČĚåĘţäÿŇïijŇPythonäijŽäžěçż§äÿĂæĺąåijŔåď ĎçŘĘæ■ćèąŇçňęãĂĆ è£ŹçğæĺąåijŔäÿŇïijŇåIJĺèŕźåŔŰæŰĞæIJňçŽĎæŮůåĂŹïijŇPythonåŔŕäžěèŕĘåĹńæĽĂæIJL'çŽĎæŹőéĂŽæ■ \n å■Ůçňęã ĂĆ çśžäijijçŽĎïijŇåIJĺè¿ŞåĞžæŮűäijŽářĘæ■ćèąŇçňę \n è¡ňæ■ćäÿžçşżçż§éżŸèőď çŽĎæ■ćèąŇçňęã ĂĆ åęĆæđIJä¡ääÿ■åÿŇæIJŽè£ŹçğæéżŸèőď çŽĎåď ĎçŘĘæŰźåijŔïijŇéĆčäźĹåŔřäžěéĂŽè£ĞäijáéĂŠäÿĂäÿłåŔřéĂL'çŽĎ open() åĞ¡æŢřäijååĚěåŔĆæŢř newline='' ïijŇářšåČŔäÿŇéİćè£ŹæăůïijŽ

```
# Read with disabled newline translation
with open('somefile.txt', 'rt', newline='') as f:
    ...
```

äÿžäžĘèŕť æŸŐäÿď èĂĚäźŇéŮť çŽĎåůőåijĆïijŇäÿŇéİćæĹŠåIJĺUnixæIJžåŹĺäÿŁéİćèŕźåŔŰäÿĂäÿłWindows hello world!\r\n ïijŽ

```
>>> # Newline translation enabled (the default)
>>> f = open('hello.txt', 'rt')
>>> f.read()
'hello world!\n'

>>> # Newline translation disabled
```

```
>>> g = open('hello.txt', 'rt', newline='')
>>> g.read()
'hello world!\r\n'
>>>
```

æIJĂåŘŐäÿĂäÿłéŮőéćŸåřśæŸŕæŰĞæIJňæŰĞäżűäÿ■åŔŕèČ¡åĞžçŐřçŽĎçijŰçăĄéŤŹèŕŕãĂĆ
ä¡Ęä¡ăèŕżåŔŰæĹŰèĂĚåĘŹåĚěäÿĂäÿłæŰĞæIJňæŰĞäżűæŮűïijŇä¡ăåŔŕèČ¡äijŽéĄĞåĹřäÿĂäÿłçijŰçăĄæĹŰè

```
>>> f = open('sample.txt', 'rt', encoding='ascii')
>>> f.read()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "/usr/local/lib/python3.3/encodings/ascii.py", line 26, in␣
 ↪decode
        return codecs.ascii_decode(input, self.errors)[0]
UnicodeDecodeError: 'ascii' codec can't decode byte 0xc3 in position
12: ordinal not in range(128)
>>>
```

åęĆæđIJåĞžçŐřè£ŹäÿłéŤŹèŕŕïijŇéĂŽåÿÿèąĺçd'žä¡ăèŕżåŔŰæŰĞæIJňæŮűæŇĞåőŽçŽĎçijŰçăĄäÿ■æ■čç
ä¡ăæIJĂåě¡äžŤçżĘéŸĚèŕżèŕť'æŸŐåźűçąőèőd'ä¡ăçŽĎæŰĞäżűçijŰçăĄæŸŕæ■čçąőçŽĎ(æŕŤåęĆä¡£çŤĺUTF-
8èĂŇäÿ■æŸŕLatin-1çijŰçăĄæĹŰåĚűäżŰ)ãĂĆ åęĆæđIJçijŰçăĄéŤŹèŕŕè£ŸæŸŕå■ŸåIJĺçŽĎèŕİïijŇä¡ăåŔŕäžě
`open()` åĞ¡æŢřäijăéĂŠäÿĂäÿłåŔŕéĂL'çŽĎ `errors` åŔĆæŢřæİěåď'ĎçŘĘè£ŹäžŻéŤŹèŕŕãĂĆ
äÿŃéIćæŸŕäÿĂäžŽåď'ĎçŘĘåÿÿèğĄéŤŹèŕŕçŽĎæŰźæşŢïijŽ

```
>>> # Replace bad chars with Unicode U+fffd replacement char
>>> f = open('sample.txt', 'rt', encoding='ascii', errors='replace')
>>> f.read()
'Spicy Jalape?o!'
>>> # Ignore bad chars entirely
>>> g = open('sample.txt', 'rt', encoding='ascii', errors='ignore')
>>> g.read()
'Spicy Jalapeo!'
>>>
```

åęĆæđIJä¡ăçżŔåÿÿä¡£çŤĺ `errors` åŔĆæŢřæİěåď'ĎçŘĘçijŰçăĄéŤŹèŕŕïijŇåŔŕèČ¡äijŽèőľä¡ăçŽĎçŢ§æt
åŕźäžŐæŰĞæIJňåď'ĎçŘĘçŽĎéęŰèęĄåŐ§åĹŹæŸŕçąőä£Ïä¡ăæĂżæŸŕä¡£çŤĺçŽĎæŸŕæ■čçąőçijŰçăĄãĂĆå¡Şa
8)ãĂĆ

5.2 æĽŞå■řè¿ŞåĞžèĞşæŰĞäżűäÿ■

éŮőécŸ

ä¡ăæČşåřĘ `print()` åĞ¡æŢřçŽĎè¿ŞåĞžéĞ■åőŽåŘŚåĹřäÿĂäÿłæŰĞäżűäÿ■åŐżãĂĆ

èğčåĘşæŰźæąĹ

åIJÍ `print()` åĞ¡æŢřäÿ■æŇĞåőŽ `file` åĚşéŤőå■ŮåŔĆæŢřïijŇåČŔäÿŃéİćè£ŹæăůïijŽ

```
with open('d:/work/test.txt', 'wt') as f:
    print('Hello World!', file=f)
```

èőĺèőž

åĚşäžŐè¿ŞåĞžéĞ■åőŽåŘŠåĹřæŰĞäżűäÿ■åřśè£ŹäžŻäžĘãĂĆä¡ĘæŸŕæIJL'äÿĂçĆźèęĄæşĺæĎŔçŽĎåřśæåęĆæđIJæŰĞäżűæŸŕäžŇè£ŻåĹűæĺąåijŔçŽĎèŕİïijŇæL'Şå■řåřśäijŽåĞžéŤŹãĂĆ

5.3 ä¡£çŤĺåĚűäżŰåĹĘéŽŤçňęæĹŰèąŇçżĹæ■ćçňęæL'Şå■ř

éŮőécŸ

ä¡ăæČşä¡£çŤÍ `print()` åĞ¡æŢřè¿ŞåĞžæŢřæ■őïijŇä¡ĘæŸŕæČşæŤźåŘŸéżŸèőđ'çŽĎåĹĘéŽŤçňęæĹŰè

èğčåĘşæŰźæąĹ

åŘŕäżěä¡£çŤĺåIJÍ `print()` åĞ¡æŢřäÿ■ä¡£çŤÍ `sep` åŠŇ `end` åĚşéŤőå■ŮåŔĆæŢřïijŇäżěä¡ăæČşèęĄçŽĎæŰźåijŔè¿ŞåĞžãĂĆærŤåęĆïijŽ

```
>>> print('ACME', 50, 91.5)
ACME 50 91.5
>>> print('ACME', 50, 91.5, sep=',')
ACME,50,91.5
>>> print('ACME', 50, 91.5, sep=',', end='!!\n')
ACME,50,91.5!!
>>>
```

ä¡£çŤÍ `end` åŘĆæŢřäź§åŘŕäżěåIJÍè¿ŞåĞžäÿ■çęĄæ■ćæ■ćèąŇãĂĆærŤåęĆïijŽ

```
>>> for i in range(5):
...     print(i)
...
0
1
2
3
4
>>> for i in range(5):
...     print(i, end=' ')
...
0 1 2 3 4 >>>
```

èőÍèőž

å¡Şä¡ăæČşä¡£çŤÍéİđçľžæăijåĹĘéŽŤçňęæİěè¿ŞåĞžæŢřæ■őçŽĎæŮűåĂŹïijÑçżŹ `print()` åĞ¡æŢřäijăéĂŠäÿĂäÿł `sep` åŔĆæŢřæŸŕæIJĂçőÅå■ŢçŽĎæŰźæąĹãĂĆæIJľæŮűåĂŹä¡ăäijŽçIJÑåĹřäÿĂäžŽçÍÑåžŔåŚŸäijŽä¡£çŤÍ `str.join()` æİěåőÑæĹŘåŘÑæăůçŽĎäžÑæČĚãĂĆærŤåęĆïijŽ

```
>>> print(','.join(('ACME','50','91.5')))
ACME,50,91.5
>>>
```

`str.join()` çŽĎéŮőécŸåIJÍäžŐåőČäżĚäżĚéĂĆçŤÍäžŐå■Ůçňęäÿšã ĂĆè£ŹæĎŔåŚşçİĂä¡ăéĂŽåÿÿéIJ

```
>>> row = ('ACME', 50, 91.5)
>>> print(','.join(row))
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: sequence item 1: expected str instance, int found
>>> print(','.join(str(x) for x in row))
ACME,50,91.5
>>>
```

ä¡ăå¡ŞçĎűåŔŕäżěäÿ■çŤÍéĆčäźĹéžżçČęïijÑäżĚäżĚåŔłéIJĂèęĄåČŔäÿÑéİćè£ŹæăůåĘŹïijŽ

```
>>> print(*row, sep=',')
ACME,50,91.5
>>>
```

5.4 èŕżåĘŹå■ŮèŁĆæŢřæ■ő

éŮőécŸ

ä¡ăæČşèŕżåĘŹäžÑè£ŽåĹűæŰĞäżűïijÑæŕŤåęĆåŻ¿çL’ĞïijÑåčŕé§şæŰĞäżűç■L’ç■L’ãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤÍæÍąåijŔäÿž `rb` æĹŰ `wb` çŽĎ `open()` åĞ¡æŢřæİěèŕżåŔŰæĹŰåĘŹåĚěäžÑè£ŽåĹűæŢřæ■őãĂĆærŤ

```
# Read the entire file as a single byte string
with open('somefile.bin', 'rb') as f:
    data = f.read()

# Write binary data to a file
with open('somefile.bin', 'wb') as f:
    f.write(b'Hello World')
```

åIJÍèŕżåŔŰäžÑè£ŽåĹűæŢřæ■őæŮűïijÑéIJĂèęĄæÑĞæŸŐçŽĎæŸŕæL’ĂæIJL’è£ŤåŻđçŽĎæŢřæ■őéČ¡æ
çśżäijijçŽĎïijÑåIJÍåĘŹåĚěçŽĎæŮűåĂŹïijÑå£Ěéążä£İèŕĄåŔĆæŢřæŸŕäžěå■ŮèŁĆå¡ćåijŔåŕźåď’ŰæŽť’éIJšæŢ

èőÍèőž

åIJÍèŕžåŘŰäžÑè£ŻåĹűæŢřæ■őçŽĎæŮűåĂŹïijÑå■ŮèŁĆå■ŮçňęäÿšåŠÑæŰĞæIJňå■ŮçňęäÿšçŽĎèŕ■äźĽ
çĽźåĹńéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijÑçť´ćåijŢåŠÑè£■äžčåŁÍä¡IJè£ŤåŽđçŽĎæŸŕå■ŮèŁĆçŽĎåĂijèĂÑäÿ■æŸ

```
>>> # Text string
>>> t = 'Hello World'
>>> t[0]
'H'
>>> for c in t:
...     print(c)
...
H
e
l
l
o
...
>>> # Byte string
>>> b = b'Hello World'
>>> b[0]
72
>>> for c in b:
...     print(c)
...
72
101
108
108
111
...
>>>
```

åęĆæđIJä¡ăæČşäżŐäžÑè£ŻåĹűæÍąåijŔçŽĎæŰĞäžűäÿ■èŕžåŘŰæĹŰåĘŹåĚěæŰĞæIJňæŢřæ■őïijÑå£Ěéą

```
with open('somefile.bin', 'rb') as f:
    data = f.read(16)
    text = data.decode('utf-8')

with open('somefile.bin', 'wb') as f:
    text = 'Hello World'
    f.write(text.encode('utf-8'))
```

äžÑè£ŻåĹűI/Oè£ŸæIJL'äÿĂäÿłéšIJäÿžäžžç§ěçŽĎçL'źæĂğåřśæŸŕæŢřçżĎåŠŇCçżŞæđĎä¡ŞçśžåđŃèČ¡çŽ

```
import array
nums = array.array('i', [1, 2, 3, 4])
with open('data.bin','wb') as f:
    f.write(nums)
```

è£ŹäÿłéĂĆçŤÍäžŐäżżä¡ŢåőđçŐřäžĘèćńçğřäźŃäÿž”çijŞåĘšæŐěåŔč”çŽĎåŕźèśąïijÑè£Źçğ■åŕźèśąäijŽçŻt

äžÑè£ŻåĹűæŢřæ■őçŽĎåĘźåĚěářśæŸŕè£Źçśżæş■ä¡IJäźÑäÿĂãĂĆ

å¿Ĺåď Žářźèśąè£ŸåĚĄèőÿéĂŽè£Ğä¡£çŤĺæŰĞäžűářźèśąçŽĎ `readinto()` æŰźæşŢçŻť æŐěèŕżåŔŰäžÑè£ŻåĹűæŢřæ■őåĹřåĚűåžŢåśĆçŽĎåĘĚå■Ÿäÿ■åŐżãĂĆæŕŤåęĆïijŽ

```
>>> import array
>>> a = array.array('i', [0, 0, 0, 0, 0, 0, 0, 0])
>>> with open('data.bin', 'rb') as f:
...     f.readinto(a)
...
16
>>> a
array('i', [1, 2, 3, 4, 0, 0, 0, 0])
>>>
```

ä¡ĘæŸŕä¡£çŤĺè£Źçğ■æŁĂæIJŕçŽĎæŮűåĂŹéIJĂèęĄæäijåď ŰåřŔå£ČïijÑåŻääÿžåőČéĂŽåÿÿåĚůæIJL'åź åŔŕäžěæ§ěçIJŃ5.9åřŔèŁĆäÿ■åŔęåď ŰäÿĂäÿłèŕżåŔŰäžÑè£ŻåĹűæŢřæ■őåĹřåŔŕä£őæŤźçijŞåĘšåŇžçŽĎä¿Ńå

5.5 æŰĞäžűäÿ■å■ŸåIJĺæĽ■èČ¡åĘźåĚě

éŮőécŸ

ä¡ăæČşåČŔäÿĂäÿłæŰĞäžűäÿ■åĘźåĚěæŢřæ■őïijÑä¡ĘæŸŕåĽ■æŔŘå£ĚéążæŸŕè£ŹäÿłæŰĞäžűåIJĺæŰĞ äźş§åřśæŸŕäÿ■åĚĄèöÿèęEçŻŰåůšå■ŸåIJĺçŽĎæŰĞäžűåĘĚåőźãĂĆ

èğčåĘşæŰźæąĹ

åŔŕäžěåIJĺ `open()` åĞ¡æŢřäÿ■ä¡£çŤĺ `x` æĺąåijŔæİěäžčæŻ£ `w` æĺąåijŔçŽĎæŰźæşŢæİěèğčåĘşè£ŹäÿłéŮőécŸãĂĆæŕŤåęĆïijŽ

```
>>> with open('somefile', 'wt') as f:
...     f.write('Hello\n')
...
>>> with open('somefile', 'xt') as f:
...     f.write('Hello\n')
...
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
FileExistsError: [Errno 17] File exists: 'somefile'
>>>
```

åęĆæđIJæŰĞäžűæŸŕäžÑè£ŻåĹűçŽĎïijÑä¡£çŤĺ `xb` æİěäžčæŻ£ `xt`

èőÍèőž

è£ŹäÿĂåřŔèŁĆæijŤçď žäžĘåIJĺåĘźæŰĞäžűæŮűéĂŽåÿÿäijŽéĄĞåĹřçŽĎäÿĂäÿłéŮőécŸçŽĎåőÑç¿Őèğ äÿĂäÿłæŻ£äžčæŰźæąĹæŸŕåĚĹæţÑèŕŢè£ŹäÿłæŰĞäžűæŸŕåŘęå■ŸåIJĺïijÑåČŔäÿÑéÍćè£Źæăůïijž

```
>>> import os
>>> if not os.path.exists('somefile'):
...     with open('somefile', 'wt') as f:
...         f.write('Hello\n')
... else:
...     print('File already exists!')
...
File already exists!
>>>
```

æŸ¿èĂŇæŸŞèğĄïijŇä¡£çŤĺxæŰĞäžűæĺąåijŕæŻťåŁăçőÅå■ŢãĂĆèęĄæşĺæĎŕçŽĎæŸŕxæĺąåijŕæŸŕäÿ
`open()` åĞ¡æŢřçL'źæIJL'çŽĎæL'ľåśŢãĂĆ åIJĺPythonçŽĎæŮğçL'ĹæIJňæĹŰèĂĚæŸŕPythonåőđçŐřçŽĎåžŢ

5.6 å■Ůçňęäÿšç ŽĎl/OæŞ■ä¡lJ

éŮőécŸ

ä¡ăæČşä¡£çŤĺæŞ■ä¡IJcśżæŰĞäžűåŕźèśąçŽĎçÍŃåžŕæİěæŞ■ä¡IJæŰĞæIJňæĹŰäžŇè£ŻåĹűå■Ůçňęäÿšã

èğčåĘşæŰźæąĹ

ä¡£çŤĺ `io.StringIO()` åŠŇ `io.BytesIO()` çśżæİěåĹŻåžžçśżæŰĞäžűåŕźèśąæŞ■ä¡IJå■Ůçňęäÿšæ

```
>>> s = io.StringIO()
>>> s.write('Hello World\n')
12
>>> print('This is a test', file=s)
15
>>> # Get all of the data written so far
>>> s.getvalue()
'Hello World\nThis is a test\n'
>>>

>>> # Wrap a file interface around an existing string
>>> s = io.StringIO('Hello\nWorld\n')
>>> s.read(4)
'Hell'
>>> s.read()
'o\nWorld\n'
>>>
```

`io.StringIO` åŔłèČ¡çŤĺäžŐæŰĞæIJñãĂĆåęĆæđIJä¡ăèęĄæŞ■ä¡IJäžŇè£ŻåĹűæŢřæ■őïijŇèęĄä¡£çŤĺ
`io.BytesIO` çśżæİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
>>> s = io.BytesIO()
>>> s.write(b'binary data')
>>> s.getvalue()
```

```
b'binary data'
>>>
```

èőÍèőž

å¡Şä¡äæČşæÍąæŃ§äÿĂäÿłæŹőéĂŽçŽĎæŰĞäžűçŽĎæŮűåĂŹ `StringIO` åŠŇ `BytesIO` çśżæŸŕå¿ĹæIJL'çŤÍçŽĎãĂĆ æŕŤåęĆïijŇåIJÍå■ŢåĚČæţŃèŕŢäÿ■ïijŇä¡ääŔŕäžěä¡£çŤÍ `StringIO` æİěåĹŽåžžäÿĂäÿłåŇĚåŘńæţŃèŕŢæŢřæ■őçŽĎçśżæŰĞäžűåŕźèśąïijŇè£ŹäÿłåŕźèśąåŔŕäžěèćńäijäçżŹæ§ŘäÿłåŔĆæŢřäÿžæŹőéĂŽæŰĞäžűåŕźèśąçŽĎåĞ¡æŢřãĂĆ

éIJĂèęĄæşĺæĎŘçŽĎæŸŕïijŇ `StringIO` åŠŇ `BytesIO` åőđä¿ŃåźűæšąæIJL'æ■čçąőçŽĎæŢt'æŢřçśżåđŃçŽĎæŰĞäžűæŔŔè£řçňęãĂĆ åŽäæ■d'ïijŇåőČäžñäÿ■èČ¡åIJĺéĆčäžŽéIJĂèęĄä¡£çŤÍçIJ§åőđçŽĎçşżçż§çžğæŰĞäžűåęĆæŰĞäžűïijŇçőąéĄŞ

5.7 èŕżåĘŹåŐŃçijľæŰĞäžű

éŮőécŸ

ä¡äæČşèŕżåĘŹäÿĂäÿłgzipæĹŰbz2æäijåijŔçŽĎåŐŃçijľæŰĞäžűãĂĆ

èğčåĘşæŰźæąĹ

`gzip` åŠŇ `bz2` æĺąåİŮåŔŕäžěå¿Ĺåőźæ ŸŞçŽĎåd'ĎçŘĘè£ŹäžŽæŰĞäžűãĂĆ äÿd'äÿłæĺąåİŮéČ¡äÿž `open()` åĞ¡æŢřæŔŘä¿ŻäžĘåŔęåd'ŰçŽĎåőđçŐřæİèèğčåĘşè£ŹäÿłéŮőécŸãĂĆ æŕŤåęĆïijŇäÿžäžĘäžěæŰĞæIJňå¡ćåijŔèŕżåŔŰåŐŃçijľæŰĞäžűïijŇåŔŕäžěè£ŹæäůåĄŽïijŽ

```
# gzip compression
import gzip
with gzip.open('somefile.gz', 'rt') as f:
    text = f.read()

# bz2 compression
import bz2
with bz2.open('somefile.bz2', 'rt') as f:
    text = f.read()
```

çśżäijijçŽĎïijŇäÿžäžĘåĘŹåĚěåŐŃçijľæŢřæ■őïijŇåŔŕäžěè£ŹæäůåĄŽïijŽ

```
# gzip compression
import gzip
with gzip.open('somefile.gz', 'wt') as f:
    f.write(text)

# bz2 compression
import bz2
```

```
with bz2.open('somefile.bz2', 'wt') as f:
    f.write(text)
```

åęĆäÿŁïijŇæL’ĂæIJL’çŽDI/OæŞ■ä¡IJéČ¡ä¡£çŤĺæŰĞæIJňæĺąåijŔåźűæL’ğèąŇUnicodeçŽDçijŰçăĄ/èğčç çśżäijijçŽDïijŇåęĆæđIJä¡ăæČşæŞ■ä¡IJäžŇè£ŻåĹűæŢřæ■őïijŇä¡£çŤĺ `rb` æĹŰèĂĚ `wb` æŰĞäżűæĺąåijŔå■şåŔŕãĂĆ

èőĺèőž

åd’ğéČĺåĹĘæČĚåĘţäÿŃèŕżåĘŹåŐŃçijl’æŢřæ■őéČ¡æŸŕå¿ĹçőĂå■ŢçŽDãĂĆä¡ĘæŸŕèęĄæşĺæĎŔçŽDæŸ åęĆæđIJä¡ääÿ■æŇĞåőŽæĺąåijŔïijŇéĆčäźĹéżŸèőd’çŽDåřśæŸŕäžŇè£ŻåĹűæĺąåijŔïijŇåęĆæđIJè£ŹæŮűåĂŹç `gzip.open()` åŠŇ `bz2.open()` æŐěåŔŮèů§åĘĚç¡őçŽD `open()` åĞ¡æŢřäÿĂæăůçŽDåŔĆæŢřïijŇ åŇĚæŃň `encoding`ïijŇ`errors`ïijŇ`newline` ç■L’ç■L’ãĂĆ

å¡ŞåĘŹåĚěåŐŃçijl’æŢřæ■őæŮűïijŇåŔŕäżěä¡£çŤĺ `compresslevel` è£ŹäÿłåŔŕéĂL’çŽDåĚşéŤőå■ŮåŔĆæŢřæİěæŇĞåőŽäÿĂäÿłåŐŃçijl’çžğåĹńãĂĆærŤåęĆïijŽ

```
with gzip.open('somefile.gz', 'wt', compresslevel=5) as f:
    f.write(text)
```

éżŸèőd’çŽDç■L’çžğæŸŕ9ïijŇäź§æŸŕæIJĂénŸçŽDåŐŃçijl’ç■L’çžğãĂĆç■L’çžğèűŁä¡ŐæĂğèČ¡èűŁåě¡ïij

æIJĂåŘŐäÿĂçĆźïijŇ `gzip.open()` åŠŇ `bz2.open()` è£ŸæIJL’äÿĂäÿłå¿ĹářŠèćńç§ěéĄŞçŽDçL’źæĂğïijŇ åőČäžňåŔŕäżěä¡IJçŤĺåIJĺäÿĂäÿłåůšå■ŸåIJĺåźűäżěäžŇè£

```
import gzip
f = open('somefile.gz', 'rb')
with gzip.open(f, 'rt') as g:
    text = g.read()
```

è£ŹæăůářšåĚĄèőÿ `gzip` åŠŇ `bz2` æĺąåİŮåŔŕäżěåůěä¡IJåIJĺèőÿåd’ŽçźżæŰĞäżűářźèśąäÿŁïijŇærŤåęĆåě

5.8 åŻžåőŽåd’ğářŔèőřå¡ŢçŽDæŰĞäżűè£■äžč

éŮőécŸ

ä¡ăæČşåIJĺäÿĂäÿłåŻžåőŽéŢ£åžęèőřå¡ŢæĹŰèĂĚæŢřæ■őåİŮçŽDéŻĘåŘĹäÿŁè£■äžčïijŇèĂŇäÿ■æŸŕåIJ

èğčåĘşæŰźæąĹ

éĂŽè£ĞäÿŃéİćè£ŹäÿłářŔæŁĂåůğä¡£çŤĺ `iter` åŠŇ `functools.partial()` åĞ¡æŢřijŽ

```
from functools import partial

RECORD_SIZE = 32
```

```
with open('somefile.data', 'rb') as f:
    records = iter(partial(f.read, RECORD_SIZE), b'')
    for r in records:
        ...
```

è£Źäÿłä¿Ńå■Řäÿ■çŽĎ `records` åŕźèśąæŸŕäÿĂäÿłåŔŕè£■äžčåŕźèśąïijŇåőČäijŽäÿ■æŰ■çŽĎäžğçŢ§åŻžèęĄæşĺæĎŔçŽĎæŸŕåęĆæđIJæĂżèőŕå¡Ţåď ğåřŔäÿ■æŸŕåİŮåď ğåřŔçŽĎæŢť æŢřåĂ■çŽĎèŕÏïijŇæIJĂåŘŐäÿĂ

èőÍèőž

`iter()` åĞ¡æŢřæIJL'äÿĂäÿłéšIJäÿžäžžç§ěçŽĎçL'źæĂğåřśæŸŕïijŇåęĆæđIJä¡äçżŹåőČäijåéĂŠäÿĂäÿłåI
è£Źäÿłè£■äžčåŹĺäijŽäÿĂçŻť'èřČçŢĺäijååĚěçŽĎåŔŕèřČçŢĺåŕźèśąçŻť'åĹřåőČè£ŢåŻđæăĞèőŕåÄijäÿžæ■ćïijŇè£

åIJĺä¿Ńå■Řäÿ■ïijŇ `functools.partial` çŢĺæİěåĹŻåžžäÿĂäÿłæŕŔæňąèćńèřČçŢĺæŮűäžŐæŰĞäžűä
æăĞèőŕåĂij `b''` åřśæŸŕå¡ŞåĹřè¿¿æŰĞäžűçżŞår¿æŮűçŽĎè£ŢåŻđåÄijãĂĆ

æIJĂåŘŐåĘ■æŔŘäÿĂçĆźïijŇäÿŁéİćçŽĎä¿Ńå■Řäÿ■çŽĎæŰĞäžűæŮűäžěäžŇè£ŻåĹűæĺąåijŔæĽŞåijĂç
åęĆæđIJæŸŕèŕźåŘŰåŻžåőŽåď ğåřŔçŽĎèőŕå¡ŢïijŇè£ŹéĂŽåÿÿæŸŕæIJĂæŹőéĄ■çŽĎæČĚåĘţãĂĆ
èĂŇåŕźäžŐæŰĞæIJñæŰĞäžűïijŇäÿĂèąŇäÿĂèąŇçŽĎèŕźåŘŰ(éżŸèőď'çŽĎè£■äžčèąŇäÿž)æŻť'æŹőéĄ■çĆźã

5.9 èŕźåŘŰäžŇè£ŻåĹűæŢřæ■őåĹřåŔŕåŘŸçijŞåĘšåŇžäÿ■

éŮőécŸ

ä¡ăæČşçŻť'æŐěèŕźåŘŰäžŇè£ŻåĹűæŢřæ■őåĹřäÿĂäÿłåŔŕåŘŸçijŞåĘšåŇžäÿ■ïijŇèĂŇäÿ■éIJĂèęĄåĄŽäź
æĹŰèĂĚä¡ăæČşåŐ§åIJřä£őæŤźæŢřæ■őåźűårĘåőČåĘŹåŻđåĹřäÿĂäÿłæŰĞäžűäÿ■åŐżãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘèŕźåŘŰæŢřæ■őåĹřäÿĂäÿłåŔŕåŘŸæŢřçżĎäÿ■ïijŇä¡£çŢĺæŰĞäžűåŕźèśąçŽĎ `readinto()` æŰźæşŢãĂĆæŕŤåęĆïijŽ

```
import os.path

def read_into_buffer(filename):
    buf = bytearray(os.path.getsize(filename))
    with open(filename, 'rb') as f:
        f.readinto(buf)
    return buf
```

äÿŃéİćæŸŕäÿĂäÿłæijŤçd'žè£Źäÿłåğ¡æŢřä¡£çŢĺæŰźæşŢçŽĎä¿Ńå■ŘïijŽ

```
>>> # Write a sample file
>>> with open('sample.bin', 'wb') as f:
...     f.write(b'Hello World')
...
```

```
>>> buf = read_into_buffer('sample.bin')
>>> buf
bytearray(b'Hello World')
>>> buf[0:5] = b'Hallo'
>>> buf
bytearray(b'Hallo World')
>>> with open('newsample.bin', 'wb') as f:
...     f.write(buf)
...
11
>>>
```

èőÍèőž

æŰĞäżűåŕźèśąçŽĎ `readinto()` æŰźæşŢèČ¡èćńçŤÍæİěäÿžécĎåĚĹåĹĘéĚ■åĘĚå■ŸçŽĎæŢřçżĎåąńåĚ `array` æÍąåİŮæĹŰ `numpy` åžŞåĹŻåżžçŽĎæŢřçżĎãĂĆ åŠŇæŹőéĂŽ `read()` æŰźæşŢäÿ■åŘŇçŽĎæŸŕïijŇ `readinto()` åąńåĚĚåůšå■ŸåIJÍçŽĎçijŞåĘšåŇžèĂŇäÿ■æŸŕäÿžæŰřåŕźèśąéĞ åŽăæ■ď˙ïijŇä¡ăåŔŕäžěä¡£çŤÍåőČæİěéĄ£åĚ■åď˙ğéĞŔçŽĎåĘĚå■ŸåĹĘéĚ■æŞ■ä¡IJãĂĆ æŕŤåęĆïijŇåęĆæđIJä¡ăèŕžåŔŰäÿĂäÿłçŤśçŻÿåŘŇåď˙ğåřŔçŽĎèőřå¡ŢçżĎæĹŘçŽĎäžŇè£ŻåĹűæŰĞäżűæŮűïij

```
record_size = 32 # Size of each record (adjust value)

buf = bytearray(record_size)
with open('somefile', 'rb') as f:
    while True:
        n = f.readinto(buf)
        if n < record_size:
            break
        # Use the contents of buf
        ...
```

åŔęåď˙ŰæIJL˙äÿĂäÿłæIJL˙èűčçL˙źæĂğåřśæŸŕ `memoryview` ïijŇ åőČåŔŕäžěéĂŽè£ĞéŽűåď˙■åĹűçŽĎæŰźåijŔåŕźåůšå■ŸåIJÍçŽĎçijŞåĘšåŇžæL˙ğèąŇåĹĞçL˙ĞæŞ■ä¡IJïijŇçŤŽè

```
>>> buf
bytearray(b'Hello World')
>>> m1 = memoryview(buf)
>>> m2 = m1[-5:]
>>> m2
<memory at 0x100681390>
>>> m2[:] = b'WORLD'
>>> buf
bytearray(b'Hello WORLD')
>>>
```

ä¡£çŤÍ `f.readinto()` æŮűéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijŇä¡ăå£ĚéążæčĂæ§ěåőČçŽĎè£ŤåŽđåĂijïijŇä

åęĆæđIJå■ŮèŁĆæŢřåŕŔäžŐçijŞåĘšåŇžåď˙ğåřŔïijŇèąÍæŸŐæŢřæ■őèćńæĹłæŰ■æĹŰèĂĚèćńçăť˙åİŔäžĘ

æIJĂåŘŐïijŇçŢŹå£ČèğĆåŕ§åĚűäžŰåĞ¡æŢřåžŞåŠŇæÍąåİŮäÿ■åŠŇ `into`

çŻÿåĚşçŽĎåĞ¡æŢř(æŕŤåęĆ `recv_into()` ïijŇ `pack_into()` ç■Ľ)ãĂĆ PythonçŽĎå¿Ĺåď˙ŽåĚűäžŰéČĺåĹĘåůšçżŔèČ¡æŤŕæŇĄçŻť˙æŐěçŽĎI/OæĹŰæŢřæ■ő€ő£éŮőæŞ■ä¡IJïijŇè£Źä

åĚşäžŐè§čæđŘäžŇè£ŽåĹűçżŞæđĎåŠŇ `memoryviews` ä¡£çŤĺæŰźæşŢçŽĎæŻť˙énŸçžğä¿Ńå■ŘïijŇèŕůåŔĆèĂČ6.12åřŔèŁĆãĂĆ

5.10 åĘĚå■ŸæŸăåřĎçŽĎäžŇè£ŽåĹűæŰĞäżű

éŮőécŸ

ä¡ăæČşåĘĚå■ŸæŸăåřĎäÿĂäÿłäžŇè£ŽåĹűæŰĞäżűåĹřäÿĂäÿłåŔŕåŔŸå■ŮèŁĆæŢřçżĎäÿ■ïijŇçŽőçŽĎåĿ

èğčåĘşæŰźæąĹ

ä¡£çŤĺ `mmap` æĺąåİŮæİěåĘĚå■ŸæŸăåřĎæŰĞäżűãĂĆ äÿŃéİćæŸŕäÿĂäÿłåůěåĚůåĞ¡æŢřïijŇåŘŠä¡ăæijŤçď˙žäžĘåęĆä¡ŢæĽ˙Şåij Ääÿ Ääÿ łæŰĞäżűåźűäżěäÿĂçğ■ä¿£æ

```
import os
import mmap

def memory_map(filename, access=mmap.ACCESS_WRITE):
    size = os.path.getsize(filename)
    fd = os.open(filename, os.O_RDWR)
    return mmap.mmap(fd, size, access=access)
```

äÿžäžĘä¡£çŤĺè£ŹäÿłåĞ¡æŢřïijŇä¡ăéIJĂèęĄæIJL˙äÿĂäÿłåůšåĹŻåžžåźűäÿŤåĘĚåőźäÿ■äÿžçľ˙žçŽĎæŰĞäż äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijŇæŢŹä¡ăæĂŐæăůåĹİåğŃåĹŻåžžäÿĂäÿłæŰĞäżűåźűåřĘåĚűåĘĚåőźæĽľ˙åĚĚåĹř

```
>>> size = 1000000
>>> with open('data', 'wb') as f:
...     f.seek(size-1)
...     f.write(b'\x00')
...
>>>
```

äÿŃéİćæŸŕäÿĂäÿłåĹľçŤĺ `memory_map()` åĞ¡æŢřçśżåĘĚå■ŸæŸăåřĎæŰĞäżűåĘĚåőźçŽĎä¿Ńå■ŘïijŽ

```
>>> m = memory_map('data')
>>> len(m)
1000000
>>> m[0:10]
b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
>>> m[0]
0
>>> # Reassign a slice
>>> m[0:11] = b'Hello World'
>>> m.close()
```

```
>>> # Verify that changes were made
>>> with open('data', 'rb') as f:
... print(f.read(11))
...
b'Hello World'
>>>
```

mmap() è£ŤåŻđçŽĎ mmap åŕźèśąåŘŇæăůäź§åŔŕäžěä¡IJäÿžäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹÍæÌěä¡£çŤĺïi
è£ŹæŮűůåĂŹåžŢåśÇçŽĎæŰĞäžűäijŽèćńèĞłåŁĺåĚşéŮ■ãĂĆærŤåęĆïijŽ

```
>>> with memory_map('data') as m:
...     print(len(m))
...     print(m[0:10])
...
1000000
b'Hello World'
>>> m.closed
True
>>>
```

éżŸèőd'æČĚåĘţäÿŃïijŇ memeory_map() åĞ¡æŢřæL'Şåij ĂçŽĎæŰĞäžűåŘŇæŮűæŤŕæŇĄèŕžåŠŇåĘŹ
äżżä¡ŢçŽĎä£őæŤźåĘĚåőźéČ¡äijŽåd'■åĹűåŻđåŐ§æÌěçŽĎæŰĞäžűäÿ■ãĂĆ
åęĆædIJéIJĂèęĄåŔłèŕżçŽĎè£éŮőæĺąåijŔïijŇåŔŕäžěçżŹåŔĆæŢř access èţŃåĂijäÿž
mmap.ACCESS_READ ãĂĆærŤåęĆïijŽ

```
m = memory_map(filename, mmap.ACCESS_READ)
```

åęĆædIJä¡ăæČşåIJĺæIJñåIJřä£őæŤźæŢřæ■őïijŇä¡ĘæŸŕåŔĹäÿ■æČşåřĘä£őæŤźåĘŹåŻđåĹřåŐ§ågŃæŰĞ
mmap.ACCESS_COPY ïijŽ

```
m = memory_map(filename, mmap.ACCESS_COPY)
```

èőÍèőž

äÿžäžĘéŽŔæIJžèő£éŮőæŰĞäžűçŽĎåĘĚåőźïijŇä¡£çŤÍ mmap
åřĘæŰĞäžűæŸăåřĎåĹřåĘĚå■ŸäÿæŸŕäÿĂäÿłéńŸæŢĹåŠŇäijŸéŻĘçŽĎæŰźæşŢãĂĆ
ä¿ŃåęĆïijŇä¡ăæŮăéIJĂæL'ŞåijĂäÿĂäÿłæŰĞäžűåźűæL'ğèąŇåď'ğéĞŔçŽĎ seek() ïijŇ
read() ïijŇ write() èřČçŤÍïijŇ åŔłéIJĂèęĄçőĂå■ŢçŽĎæŸăåřĎæŰĞäžűåźűä¡£çŤÍåĹĞçL'ĞæŞ■ä¡IJèő£é

äÿĂèĹñæÌěèőšïijŇ mmap() æL'ĂæŽť'éIJšçŽĎåĘĚå■ŸçIJŃäÿŁåŐżåřśæŸŕäÿĂäÿłäžŇè£ŻåĹűæŢřçżĎårź
ä¡ĘæŸŕïijŇä¡ăåŔŕäžěä¡£çŤÍäÿĂäÿłåĘĚå■ŸèğĘåŻ¿æÌèèğčædŘåĚűäÿ■çŽĎæŢřæ■őãĂĆærŤåęĆïijŽ

```
>>> m = memory_map('data')
>>> # Memoryview of unsigned integers
>>> v = memoryview(m).cast('I')
>>> v[0] = 7
>>> m[0:4]
b'\x07\x00\x00\x00'
>>> m[0:4] = b'\x07\x01\x00\x00'
```

```
>>> v[0]
263
>>>
```

éIJÄèęĄåijžèřČçŽĎäÿĂçĆźæŸŕïijŇåĘĚå■ŸæŸăářĎäÿĂäÿłæŰĞäżűåźűäÿ■äijŽářijèĞťæŢťäÿłæŰĞäżűë
äź§åřśæŸŕèŕťïijŇæŰĞäżűåźűæšąæIJL’èćńåď■åĹűåĹřåĘĚå■ŸçijŞå■ŸæĹŰæŢřçżĎäÿ■ãĂĆçŻÿåŔ■ïijŇæŞ■ä¡
å¡Şä¡ãèő£éŮőæŰĞäżűçŽĎäÿ■åŘŇåŇžå§§æŮűïijŇè£ŹäžŻåŇžå§§çŽĎåĘĚåőźæL’■æăźæ■őéIJÄèęĄèćńèŕžåŔ
èĂŇéĆčäžŻäżŐæšąèćńèő£éŮőåĹřçŽĎéČĺåĹĘè£ŸæŸŕçŢŹåIJÍçčĄçŻŸäÿŁãĂĆæL’ĂæIJL’è£ŹäžŻè£ĞçÍŃæŸ

åęĆæđIJåď’Žäÿ†Pythonèğčé GŁåŹĺåĘĚå■ŸæŸăářĎåŘŇäÿĂäÿłæŰĞäżűïijŇå¿ŮåĹřçŽĎ
`mmap` åŕźèśąèČ¡åď§èćńçŤĺæİěåIJÍèğčéĞŁåŹĺçŻť’æŐëäžď’æ■ćæŢřæ■őãĂĆ
äź§åřśæŸŕèŕťïijŇæL’ĂæIJL’èğčéĞŁåŹĺéČ¡èČ¡åŘŇæŮűèŕżåĘŹæŢřæ■őïijŇåźűäÿŤåĔűäÿ■äÿĂäÿłèğčéĞŁåŹĺa
å¿ĹæŸŐæŸ¿ïijŇè£ŹéĞŇéIJÄèęĄèĂČèŹŠåŘŇæ■ěçŽĎéŮőécŸãĂĆä¡ĘæŸŕè£Źçğ■æŰźæşŢæIJL’æŮűåĂŹåŔ

è£ŹäÿĂåřŔèŁĆäÿ■åĞ¡æŢřår¡éĞŔåĘŹå¿Ůå¿ĹéĂŽçŤĺïijŇåŘŇæŮűéĂĆçŤĺäžŐUnixåŠŇWindowsåźşåŔì
èęĄæşĺæĎŔçŽĎæŸŕä¡£çŤĺ `mmap()` åĞ¡æŢřæŮűäijŽåIJÍåžŢåśĆæIJL’äÿĂäžŻåźşåŘŕçŽĎåůőåijĆæĂğãĂĆ
åŔęåď’ŰïijŇè£ŸæIJL’äÿĂäžŻéĂL’éąźåŔŕäžěçŤĺæİěåĹŻåżžåŇ£åŘ■çŽĎåĘĚå■ŸæŸăářĎåŇžå§§ãĂĆ
åęĆæđIJä¡ăářźè£ŹäÿłæĎ§åĚt’èűčïijŇçąőä£Ïä¡ăäžŤçżĘçăŤèŕżäžĘPythonæŰĞæąčäÿ■
è£ŹæŰźéĺćçŽĎåĘĚåőź ãĂĆ

5.11 æŰĞäżűèůŕå¿ĎåŘ■çŽĎæŞ■ä¡IJ

éŮőécŸ

ä¡ăéIJÄèęĄä¡£çŤĺèůŕå¿ĎåŘ■æİěèŐůåŔŰæŰĞäżűåŘ■ïijŇçŻőå¡ŢåŘ■ïijŇçżİåŕźèůŕå¿Ďç■L’ç■L’ãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤĺ `os.path` æĺąåİŮäÿ■çŽĎåĞ¡æŢřæİěæŞ■ä¡IJèůŕå¿ĎåŘ■ãĂĆ
äÿŇéĺćæŸŕäÿĂäÿłäžď’äžŠåijŔä¿Ńå■ŘæİěæijŤçď’žäÿĂäžŻåĘşéŤőçŽĎçL’źæĂğïijŽ

```
>>> import os
>>> path = '/Users/beazley/Data/data.csv'

>>> # Get the last component of the path
>>> os.path.basename(path)
'data.csv'

>>> # Get the directory name
>>> os.path.dirname(path)
'/Users/beazley/Data'

>>> # Join path components together
>>> os.path.join('tmp', 'data', os.path.basename(path))
'tmp/data/data.csv'

>>> # Expand the user's home directory
>>> path = '~/Data/data.csv'
```

```
>>> os.path.expanduser(path)
'/Users/beazley/Data/data.csv'

>>> # Split the file extension
>>> os.path.splitext(path)
('~/Data/data', '.csv')
>>>
```

èőÍèőž

åŕźäžŐäżżä¡ŢçŽĎæŰĞäžűåŘ■çŽĎæŞ■ä¡IJïijŇä¡äéČ¡åžŤèŕěä¡£çŤÍ os.path æÍąåİŮïijŇèĂŇäÿ■æŸŕä¡£çŤÍæăĞåĞĘå■ŮçňęäÿšæŞ■ä¡IJæİěæđĎéĂäèĞłåůśçŽĎäżčçăĄãĂĆçĽźåĹńæŸŕäÿžäžĘåŔŕçğżæď■æĂğèĂČèŹŚçŽĎæŮűåĂŹæŻť åžŤåęĆæ■ď'ïijŇ åŽääÿž os.path æÍąåİŮç§ěéĄŞUnixåŠŇWindowsçşżçż§äźŃéŮťçŽĎåůőåijĆåźűäÿŤèČ¡åď§åŔŕéİääIJřåď'ĎçŘĘçśżäijij Data/data.csv åŠŇ Data\data.csv è£ŹæăůçŽĎæŰĞäžűåŘ■ãĂĆ åĚűæňąïijŇä¡äçIJ§çŽĎäÿ■åžŤèŕěæţłèť źæŮűéŮť åŐżéĞ■åď'■éĂäè¡őå■ŘãĂĆéĂŽåÿÿæIJĂåě¡æŸŕçŻť æŐěä¡£

èęĄæşÍæĎŔçŽĎæŸŕ os.path è£ŸæIJL'æŻť åď'ŽçŽĎåŁ§èČ¡åIJÍè£ŹéĞŇåźűæšąæIJL'åĹŮäÿ¿åĞžæİěã åŔŕäżěæ§ěéŸĚåőŸæŰźæŰĞæąčæİěèŐůåŔŰæŻť åď'ŽäÿŐæŰĞäžűæţŃèŕŢïijŇçňęåŘůéŞ¿æŐěç■L'çŽÿåĚşçŽ

5.12 æţŃèŕŢæŰĞäžűæŸŕåŘęå■ŸåIJÍ

éŮőécŸ

ä¡ăæČşæţŃèŕŢäÿĂäÿłæŰĞäžűæĹŰçŻőå¡ŢæŸŕåŘęå■ŸåIJÍãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤÍ os.path æÍąåİŮæİěæţŃèŕŢäÿĂäÿłæŰĞäžűæĹŰçŻőå¡ŢæŸŕåŘęå■ŸåIJÍãĂĆæŕŤåęĆïijŽ

```
>>> import os
>>> os.path.exists('/etc/passwd')
True
>>> os.path.exists('/tmp/spam')
False
>>>
```

ä¡ăè£ŸèČ¡è£ŻäÿĂæ■ěæţŃèŕŢè£ŹäÿłæŰĞäžűæŮűäżĂäźĹçśżåđŃçŽĎãĂĆ åIJÍäÿŃéİćè£ŹäžŻæţŃèŕŢäÿ■ïijŇåęĆæđIJæţŃèŕŢçŽĎæŰĞäžűäÿ■å■ŸåIJÍçŽĎæŮűåĂŹïijŇçżŞæđIJéČ¡äijŽè

```
>>> # Is a regular file
>>> os.path.isfile('/etc/passwd')
True

>>> # Is a directory
>>> os.path.isdir('/etc/passwd')
```

```
False

>>> # Is a symbolic link
>>> os.path.islink('/usr/local/bin/python3')
True

>>> # Get the file linked to
>>> os.path.realpath('/usr/local/bin/python3')
'/usr/local/bin/python3.3'
>>>
```

åęĆædIJä¡ăè£ŸæČşèŐůåŔŰåĚČæŢřæ■ő(æŕŤåęĆæŰĞäżűåď ğářŔæĹŰèĂĚæŸŕä£őæŤźæŮěæIJ§)ïijŇäź
`os.path` æĺąåİŮæİěèğčåEşïijŽ

```
>>> os.path.getsize('/etc/passwd')
3669
>>> os.path.getmtime('/etc/passwd')
1272478234.0
>>> import time
>>> time.ctime(os.path.getmtime('/etc/passwd'))
'Wed Apr 28 13:10:34 2010'
>>>
```

èőÍèőž

ä¡£çŤÍ `os.path` æİěè£ŻèąŇæŰĞäżűæţŃèŕŢæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆ
åIJĺåĘŹè£ŹäžŻèĎŽæIJňæŮűïijŇåŔŕèČ¡åŤŕäÿĂéIJĂèęĄæşĺæĎŔçŽĎåřśæŸŕä¡ăéIJĂèęĄèĂČèŹŚæŰĞäżűæİČ

```
>>> os.path.getsize('/Users/guido/Desktop/foo.txt')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "/usr/local/lib/python3.3/genericpath.py", line 49, in␣
 ↪getsize
        return os.stat(filename).st_size
PermissionError: [Errno 13] Permission denied: '/Users/guido/
 ↪Desktop/foo.txt'
>>>
```

5.13 èŐůåŔŰæŰĞäżűåď źäÿ■çŽĎæŰĞäżűåĹŮèąĺ

éŮőécŸ

ä¡ăæČşèŐůåŔŰæŰĞäżűçşżçż§äÿ■æ§ŘäÿłçŻőå¡ŢäÿŃçŽĎæĽĂæIJĽæŰĞäżűåĹŮèąĺãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤÍ `os.listdir()` åĞ¡æŢřæİěèŐůåŔŰæ§ŘäÿłçŻőå¡Ţäÿ■çŽĎæŰĞäžűåĹŮèąĺïijŽ

```
import os
names = os.listdir('somedir')
```

çžŞæđIJäijŽè£ŤåŻđçŻőå¡Ţäÿ■æĽĂæIJL'æŰĞäžűåĹŮèąĺïijŇåŇĚæŃňæĽĂæIJL'æŰĞäžűïijŇå■ŘçŻőå¡Ţ åęĆæđIJä¡ăéIJĂèęĄéĂŽè£Ğæ§Řçğ■æŰźåijŔè£Ğæžď'æŢřæ■őïijŇåŔŕäžěèĂČèŹŚçżŞåŘĹ `os.path` åžŞäÿ■çŽĎäÿĂäžŻåĞ¡æŢřæİěä¡£çŤĺåĹŮèąĺæŐĺåŕijãĂĆæŕŤåęĆïijŽ

```
import os.path

# Get all regular files
names = [name for name in os.listdir('somedir')
        if os.path.isfile(os.path.join('somedir', name))]

# Get all dirs
dirnames = [name for name in os.listdir('somedir')
        if os.path.isdir(os.path.join('somedir', name))]
```

å■Ůçňęäÿšç ŽĎ `startswith()` åŠŇ `endswith()` æŰźæşŢåŕźäžŐè£ĞæžďäÿĂäÿłçŻőå¡ŢçŽĎåĘĚåőźäź§æŸŕå¿ĹæIJL'çŤĺçŽĎãĂĆæŕŤåęĆïijŽ

```
pyfiles = [name for name in os.listdir('somedir')
           if name.endswith('.py')]
```

åŕźäžŐæŰĞäžűåŘ■çŽĎåŇźéĚ■ïijŇä¡ăåŔŕèČ¡äijŽèĂČèŹŚä¡£çŤÍ `glob` æĹŰ `fnmatch` æĺąåİŮãĂĆæŕŤåęĆïijŽ

```
import glob
pyfiles = glob.glob('somedir/*.py')

from fnmatch import fnmatch
pyfiles = [name for name in os.listdir('somedir')
           if fnmatch(name, '*.py')]
```

èŐĺèŐž

èŐůåŔŰçŻőå¡Ţäÿ■çŽĎåĹŮèąĺæŸŕå¿ĹåőźæŸ§çŽĎïijŇä¡ĘæŸŕåĚűè£ŤåŻđçżŞæđIJåŔłæŸŕçŻőå¡Ţäÿ■åĉ åęĆæđIJä¡ăè£ŸæČşèŐůåŔŰåĚűäžŰçŽĎåĚČä£ąæĄŕïijŇæŕŤåęĆæŰĞäžűåđ'ğåřŔïijŇä£őæŤźæŮűéŮť'ç■Ľ'ç■ä¡ăæĹŰèőÿè£ŸéIJĂèęĄä¡£çŤĺåĹř `os.path` æĺąåİŮäÿ■çŽĎåĞ¡æŢřæĹŰçİĂ `os.stat()` åĞ¡æŢřæİěæŤűéŻĘæŢřæ■őãĂĆæŕŤåęĆïijŽ

```
# Example of getting a directory listing

import os
import os.path
import glob
```

```
pyfiles = glob.glob('*.py')

# Get file sizes and modification dates
name_sz_date = [(name, os.path.getsize(name), os.path.
↪getmtime(name))
                for name in pyfiles]
for name, size, mtime in name_sz_date:
    print(name, size, mtime)

# Alternative: Get file metadata
file_metadata = [(name, os.stat(name)) for name in pyfiles]
for name, meta in file_metadata:
    print(name, meta.st_size, meta.st_mtime)
```

æIJÄåŘŐè£ŸæIJL'äÿĂçĆźèęĄæşÍæĎŔçŽĎåřśæŸřïijŇæIJL'æŮűåĂŹåIJÍåď'ĎçŘĘæŰĞäżűåŘ■çijŰçăĄé
éĂŽåÿÿæİěèőšïijŇåĞ¡æŢř `os.listdir()` è£ŤåŻđçŽĎåőđä¡ŞåĹŮèąÍäijŽæăźæ■őçşžçż§éžŸèőď'çŽĎæŰĞä
ä¡ĘæŸřæIJL'æŮűåĂŹäź§äijŽçćřåĹřäÿĂäžŽäÿ■èČ¡æ■čåÿÿèğčçăĄçŽĎæŰĞäžűåŘ■ãĂĆ
åĚşäžŐæŰĞäżűåŘ■çŽĎåď'ĎçŘĘéŮőécŸïijŇåIJÍ5.14åŠŇ5.15åřŔèŁĆæIJL'æŻt'èŕęçżĘçŽĎèőšèğčãĂĆ

5.14 å£¡çŢěæŰĞäżűåŘ■çijŰçăĄ

éŮőécŸ

ä¡ăæČşä¡£çŤĺåŐ§ågŇæŰĞäżűåŘ■æL'ğèąŇæŰĞäžűçŽĎI/OæŞ■ä¡IJïijŇäź§åřśæŸřèŕť'æŰĞäżűåŘ■åźűæš

èğčåĘşæŰźæąĹ

éżŸèőď'æČĚåĘţäÿŃïijŇæL'ĂæIJL'çŽĎæŰĞäżűåŘ■éČ¡äijŽæăźæ■ő `sys.getfilesystemencoding()` è£ŤåŻđçŽĎæŰĞæIJñçijŰçăĄæİěçijŰçăĄæĹŰèğčçăĄãĂĆæŕŤåęĆïijŽ

```
>>> sys.getfilesystemencoding()
'utf-8'
>>>
```

åęĆæđIJåŽääÿžæ§Řçğ■åŐ§åŽää¡ăæČşå£¡çŢěè£Źçğ■çijŰçăĄïijŇåŔŕäżěä¡£çŤĺäÿĂäÿłåŐ§ågŇå■ŮèŁĆå

```
>>> # Wrte a file using a unicode filename
>>> with open('jalape\xf1o.txt', 'w') as f:
...     f.write('Spicy!')
...
6
>>> # Directory listing (decoded)
>>> import os
>>> os.listdir('.')
['jalapeÃśo.txt']
```

```
>>> # Directory listing (raw)
>>> os.listdir(b'.') # Note: byte string
[b'jalapen\xcc\x83o.txt']

>>> # Open file with raw filename
>>> with open(b'jalapen\xcc\x83o.txt') as f:
...     print(f.read())
...
Spicy!
>>>
```

æ■čåęĆä¡ăæĽĂèğĄïijŇåIJÍæIJĂåŘŐäÿd'äÿłæŞ■ä¡IJäÿ■ïijŇå¡Şä¡ăçżŹæŰĞäžűçŻÿåĚşåĞ¡æŢřåęĆ
`open()` åŠŇ `os.listdir()` äijăéĂŠå■ÙèŁĆå■ŮçňęäÿšæŮűïijŇæŰĞäžűåŘ■çŽĎåd'ĎçŘĘæŰźåijŔäijŽçĺ

èőÍèőž

éĂŽåÿÿæİěèőšïijŇä¡ăäÿ■éIJĂèęĄæŃĚå£ČæŰĞäžűåŘ■çŽĎçijŰçăĄåŠŇèğčçăĄïijŇæŹőéĂŽçŽĎæŰĞäž
ä¡ĘæŸŕïijŇæIJL'äžŻæŞ■ä¡IJçşżçż§åĚĄèőÿçŤÍæĹůéĂŽè£ĞåĄűçĎűæĹŰæĄűæĎŔæŰźåijŔåŐżåĹŻåżžåŘ■å■
è£ŹäžŻæŰĞäžűåŘ■åŔŕèČ¡äijŽçěđçğŸåIJŕäÿ■æŰ■éĆčäžŻéIJĂèęĄåd'ĎçŘĘåd'ğéĞŘæŰĞäžűçŽĎPythonçÍŃ

èŕżåŔŰçŻőå¡ŢåźűéĂŽè£ĞåŐ§ågŃæIJłèğčçăĄæŰźåijŔåd'ĎçŘĘæŰĞäžűåŘ■åŔŕäžěæIJL'æŢĹçŽĎéĄ£å
år¡çőąè£ŹæăůäijŽåÿęæİěäÿĂåőŽçŽĎçijŰçÍŃéŽ¿åžęãĂĆ

åĚşäžŐæL'Şå■řäÿ■åŔŕèğčçăĄçŽĎæŰĞäžűåŘ■ïijŇèŕůåŔĆèĂČ5.15åřŔèŁĆãĂĆ

5.15 æL'Şå■řäÿ■åŘĹæşŢçŽĎæŰĞäžűåŘ■

éŮőécŸ

ä¡ăçŽĎçÍŃåžŔèŐůåŔŰäžĘäÿĂäÿłçŻőå¡Ţäÿ■çŽĎæŰĞäžűåŘ■åĹŮèąĺïijŇä¡ĘæŸŕå¡ŞåőČèŕŢçİĂåŐżæL'Ş
åĞžçŐřäžĘ `UnicodeEncodeError` åijĆåÿÿåŠŇäÿĂæİąåěĞæĂłçŽĎæűĹæĄŕâĂŤâĂŤ
`surrogates not allowed` ãĂĆ

èğčåĘşæŰźæąĹ

å¡ŞæL'Şå■řæIJłç§ěçŽĎæŰĞäžűåŘ■æŮűïijŇä¡£çŤÍäÿŃéİćçŽĎæŰźæşŢåŔŕäžěéĄ£åĚ■è£ŹæăůçŽĎéŤŹè

```
def bad_filename(filename):
    return repr(filename)[1:-1]

try:
    print(filename)
except UnicodeEncodeError:
    print(bad_filename(filename))
```

èőÍèőž

è£ŹäÿĂåřŔèŁĆèőÍèőžçŽĎæŸŕåIJÍçijŰåEŹå£Ěéąžåď ĎçŘĘæŰĞäžűçşžçż§çŽĎçÍŃåžŔæŮűäÿĂäÿłäÿ■åd
éżŸèőď æČĚåĘţäÿŃïijŇPythonåĄĞåőŽæĽĂæIJL'æŰĞäžűåŘ■éČ¡åůšçżŔæăźæ■ő
`sys.getfilesystemencoding()` çŽĎåĂijçijŰçăĄè£ĞäžĘãĂĆ
ä¡ĘæŸŕïijŇæIJL'äÿĂäžŽæŰĞäžűçşžçż§åźűæšąæIJL'åijžåĹűèęĄæśĆè£ŹæăůåĄŽïijŇåŽăæ■ď åĚĄèőÿåĹŽåžž
è£ŹçğæČĚåĘţäÿ■åď łåÿÿèğĄïijŇä¡ĘæŸŕæĂżäijŽæIJL'äžŽçŤÍæĹůåĘŠéŹl'è£ŹæăůåĄŽæĹŰèĂĚæŸŕæŮăæĎ
åŔŕèČ¡æŸŕåIJÍäÿĂäÿłæIJL'çijžéŹůçŽĎäžčçăĄäÿ■çżŹ `open()`
åĞ¡æŢŕäijăéĂŠäžĘäÿĂäÿłäÿ■åŘĹèğĎèŇČçŽĎæŰĞäžűåŘ■)ãĂĆ

å¡ŞæĽğèąŇçśžäijij `os.listdir()` è£ŹæăůçŽĎåĞ¡æŢŕæŮűïijŇè£ŹäžŽäÿ■åŘĹèğĎèŇČçŽĎæŰĞäžűå
äÿĂæŰźéİćïijŇåőČäÿ■èČ¡äžĚäžĚåŔłæŸŕäÿćåijČè£ŹäžŽäÿ■åŘĹæăijçŽĎåŘ■å■ŮãĆèĂŇåŔęäÿĂæŰźéİćïij
Pythonåŕźè£ŹäÿłéŮőécŸçŽĎèğčåĘşæŰźæąĹæŸŕäžŐæŰĞäžűåŘ■äÿ■èŐůåŔŰæIJłèğčçăĄçŽĎå■ŮèŁĆåĂijæ
`\xhh` åźűåřĘåőČæŸăåřĎæĹŘUnicodeå■Ůçňę `\udchh` èąĺçď žçŽĎæL'ĂèřŞçŽĎ"äžčçŘĘçijŰçăĄ"ãĂĆ
äÿŃéİćäÿĂäÿłä¿Ńå■ŘæijŤçď žäžĘą¡ŞäÿĂäÿłäÿ■åŘĹæăijçŽőą¡ŢåĹŮèąĺäÿ■åŘńæIJL'äÿĂäÿłæŰĞäžűåŘ■äÿžb
1èĂŇäÿ■æŸŕUTF-8çijŰçăĄ)æŮűçŽĎæăůå■ŘïijŽ

```
>>> import os
>>> files = os.listdir('.')
>>> files
['spam.py', 'b\udce4d.txt', 'foo.txt']
>>>
```

åęĆæđIJä¡ăæIJL'äžčçăĄéIJĂèęĄæŞ■ä¡IJæŰĞäžűåŘ■æĹŰèĂĚårĘæŰĞäžűåŘ■äijăéĂŠçżŹ
`open()` è£ŹæăůçŽĎåĞ¡æŢŕïijŇäÿĂåĹĞéČ¡èČ¡æ■čåÿÿåůëä¡IJãĂĆ
åŘłæIJL'å¡Şä¡ăæČşèęĄè¿ŞåĞžæŰĞäžűåŘ■æŮűæL'■äijŽçćřåĹřäžŹéžžçČę(æŕŤåęĆæL'Şå■řè¿ŞåĞžåĹřåśŔåź
çL'źåĹńçŽĎïijŇå¡Şä¡ăæČşæL'Şå■řäÿŁéİćçŽĎæŰĞäžűåŘ■åĹŮèąĺæŮűïijŇä¡ăçŽĎçÍŃåžŔårśäijŽåť l'æžČïijŽ

```
>>> for name in files:
...     print(name)
...
spam.py
Traceback (most recent call last):
    File "<stdin>", line 2, in <module>
UnicodeEncodeError: 'utf-8' codec can't encode character '\udce4' in
position 1: surrogates not allowed
>>>
```

çÍŃåžŔåť l'æžČçŽĎåŐ§åŻăářśæŸŕå■Ůçňę `\udce4` æŸŕäÿĂäÿłéİđæşŢçŽĎUni-
codeå■Ůçňęã ĂĆ åőČåĚűåőđæŸŕäÿĂäÿłèćńçğřäÿžäžčçŘĘå■Ůçňęåŕźç ŽĎåŔŇå■Ůçňęçż ĎåŘĹçŽĎåŘŐå■ŁéČ
çŤśäžŐçijžåřŚäžĘåL'■å■ŁéČĺåĹĘïijŇåŽăæ■ď åőČæŸŕäÿłéİđæşŢçŽĎUnicodeãĂĆ
æL'Ăäžěïij ŇåŤŕäÿĂèČ¡æĹŘåŁ§è¿ŞåĞžçŽĎæŰźæşŢåŕśæŸŕå¡ŞéĄĞåĹřäÿ■åŘĹæşŢæŰĞäžűåŘ■æŮűéĞĞåŔU
æŕŤåęĆåŔŕäžěåřĘäÿŁè£ŕäžčçăĄä£őæŤźåęĆäÿŃïijŽ

```
>>> for name in files:
... try:
...     print(name)
... except UnicodeEncodeError:
...     print(bad_filename(name))
...
```

```
spam.py
b\udce4d.txt
foo.txt
>>>
```

åIJÍ bad_filename() åĞ¡æŢřäÿ■æĂŐæäůåď'Ďç¡őåŔŰåĘşäžŐä¡äèĞłåůśãĂĆ
åŔęåď'ŰäÿĂäÿłéĂL'æŃl'åřśæŸŕéĂŽè£Ğæ§Řçğ■æŰźåijŔéĞ■æŰřçijŰçăĄïijŇçď'žä¿ŃåęĆäÿŃïijŽ

```
def bad_filename(filename):
    temp = filename.encode(sys.getfilesystemencoding(), errors=
↪'surrogateescape')
    return temp.decode('latin-1')
```

èŕŚèĂĚæşÍ:

```
surrogateescape:
è£Źçğ■æŸŕPythonåIJÍçżÍåd'ğéČÍåÍĘéÍćåŘŚOSçŽĎAPIäÿ■æL'Ăä¡£çŤÍçŽĎéŤŹèŕŕåd'ĎçŘĘåŹÍïi
åőČèČ¡äżěäÿĂçğ■äijŸéŽĚçŽĎæŰźåijŔåd'ĎçŘĘçŤśæŞ■ä¡IJçşżçżŞæŔŘä¿ŽçŽĎæŢřæ■őçŽĎçijŰçăĄé
åIJÍèğčçăĄåĞžéŤŹæŮűäijŽåřĘåĞžéŤŹå■ÙèŁĆå■ŸåĆÍåÍŕäÿĂäÿłå¿ÍåřŚèćńä¡£çŤÍåÍŕçŽĎUnicode
åIJÍçijŰçăĄæŮűåřĘéĆčäžżéŽŘèŮŔåĂijåŔÍè£ŸåŐ§åŻđåŐ§åĚÍèğčçăĄåd'śèt'ěçŽĎå■ÙèŁĆåžŘåÍŮ
åőČäÿ■äżĚåŕźäžŐOS␣
↪APIéÍđåÿÿæIJL'çŤÍïijÑäź§èČ¡å¿ÍåőźæŸ§çŽĎåd'ĎçŘĘåĚűäżŰæČĚåĘţäÿŃçŽĎçijŰçăĄéŤŹèŕŕã
```

ä¡£çŤÍè£ŹäÿłçL'ÍæIJňäžğçŤ§çŽĎè¿ŞåĞžåęĆäÿŃïijŽ

```
>>> for name in files:
...     try:
...         print(name)
...     except UnicodeEncodeError:
...         print(bad_filename(name))
...
spam.py
bÃd'd.txt
foo.txt
>>>
```

è£ŹäÿĂåřŔèŁĆäÿżécŸåŔŕèČ¡äijŽèćńåď'ğéČÍåÍĘèŕżèĂĚæL'Ăå£¡çŢěãĂĆä¡ĘæŸŕåęĆæđIJä¡ăåIJÍçijŰåĘ
åřśå£Ěéążå¿ŮèĂČèŹŚåÍŕè£ŹäÿłãĂĆåŘęåÍŹä¡ăåŔŕèČ¡äijŽåIJÍæ§Řäÿłåŝ́ÍæIJńèćńåŔńåÍŕåŁđåĚñåőd'åŐżèřČ

5.16 åćđåŁăæĹŰæŤźåŔŸåůšæL'ŞåijĂæŰĞäžűçŽĎçijŰçăĄ

éŮőécŸ

ä¡ăæČşåIJÍäÿ■åĚşéŮ■äÿĂäÿłåůšæL'ŞåijĂçŽĎæŰĞäžűåL'■æŔŘäÿŃåćđåŁăæĹŰæŤźåŔŸåőČçŽĎUnico

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăæČşçżŹäÿĂäÿłäžëäžŇè£ŻåĹűæĺąåijŔæL'ŞåijĂçŽĎæŰĞäžűæůżåŁăUnicodeçijŰçăĄ/èğčçăĄ
åŔŕäžëä¡£çŤĺ `io.TextIOWrapper()` åŕźèśąåŇĚèčĚåőČãĂĆæŕŤåęĆïijŽ

```
import urllib.request
import io

u = urllib.request.urlopen('http://www.python.org')
f = io.TextIOWrapper(u, encoding='utf-8')
text = f.read()
```

åęĆædIJä¡ăæČşä£őæŤźäÿĂäÿłåůšçżŔæL'ŞåijĂçŽĎæŰĞæIJňæĺąåijŔçŽĎæŰĞäžűçŽĎçijŰçăĄæŰźåijŔ
`detach()` æŰźæşŢçğżéŹď'æŐL'åůšå■ŸåIJĺçŽĎæŰĞæIJňçijŰçăĄåśĆïijŇ
åźűä¡£çŤĺæŰřçŽĎçijŰçăĄæŰźåijŔäžčæŽ£ãĂĆäÿŇéİćæŸŕäÿĂäÿłåIJĺ `sys.stdout`
äÿŁä£őæŤźçijŰçăĄæŰźåijŔçŽĎä¿Ńå■ŘïijŽ

```
>>> import sys
>>> sys.stdout.encoding
'UTF-8'
>>> sys.stdout = io.TextIOWrapper(sys.stdout.detach(), encoding=
↪'latin-1')
>>> sys.stdout.encoding
'latin-1'
>>>
```

è£Źæăůå˛ŽåŔŕèČ¡äijŽäÿ■æŰ■ä¡ăçŽĎçżĹçńŕïijŇè£ŹéĞŇäžĚäžĚæŸŕäÿžäžĘæijŢçď'žèĂŇåůšãĂĆ

èőĺèőž

I/Oçşżçž§çŤśäÿĂçşżåĹŮçŽĎåśĆæňąædĎåżžèĂŇæĹŘãĂĆä¡ăåŔŕäžëèŕŢçİÄè£ŘèąŇäÿŃéİćè£ŹäÿłæŞ■ä

```
>>> f = open('sample.txt','w')
>>> f
<_io.TextIOWrapper name='sample.txt' mode='w' encoding='UTF-8'>
>>> f.buffer
<_io.BufferedWriter name='sample.txt'>
>>> f.buffer.raw
<_io.FileIO name='sample.txt' mode='wb'>
>>>
```

åIJĺè£Źäÿłä¿Ńå■Řäÿ■ïijŇ`io.TextIOWrapper` æŸŕäÿĂäÿłçijŰçăĄåŠŇèğčçăĄU-
nicodeçŽĎæŰĞæIJňåď'ĎçŘĘåśĆïijŇ `io.BufferedWriter`
æŸŕäÿĂäÿłåď'ĎçŘĘäžŇè£ŻåĹűæŢŕæ■őçŽĎåÿęçijŞåĘšçŽĎI/OåśĆïijŇ `io.FileIO`
æŸŕäÿĂäÿłèąĺçď'žæŞ■ä¡IJçşżçž§åžŢåśĆæŰĞäžűæŔŘè£rçňęçŽĎåŐ§ågŃæŰĞäžűãĂĆ
åćđåŁăæĹŰæŤźåŔŸæŰĞæIJňçijŰçăĄäijŽæűL'åŔŁåćđåŁăæĹŰæŤźåŔŸæIJĂäÿŁéİćçŽĎ
`io.TextIOWrapper` åśĆãĂĆ

äÿĂèĹňæİèèőšïijŇåČŔäÿŁéİćä¿Ńå■Řè£ŹæăůéĂŽè£Ğèŏ£éŮőåśđæĂğåĂijæİěçŻť'æŐěæŞ■ä¡IJäÿ■åŘŇç
ä¿ŃåęĆïijŇåęĆædIJä¡ăèŕŢçİÄä¡£çŤĺäÿŃéİćè£ŹæăůçŽĎæŁĂæIJŕæŤźåŔŸçijŰçăĄçIJŃçIJŃäijŽåŔŠçŢ§äžĂä

```
>>> f
<_io.TextIOWrapper name='sample.txt' mode='w' encoding='UTF-8'>
>>> f = io.TextIOWrapper(f.buffer, encoding='latin-1')
>>> f
<_io.TextIOWrapper name='sample.txt' encoding='latin-1'>
>>> f.write('Hello')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ValueError: I/O operation on closed file.
>>>
```

çżŞæđIJåĞžéŤŹäžĘïijŇåŻääÿžfçŽĎåŐ§ågŃåĂijåůšçżŔèćńçăť åİŔäžĘåźűåĚşéŮ■äžĘåžŢåśĆçŽĎæŰĞäz

detach() æŰźæşŢäijŽæŰ■åijĂæŰĞäżűçŽĎæIJĂéąűåśĆåźűè£ŤåŻđçňňäžŇåśĆïijŇäźŃåŘŐæIJĂéąűå

```
>>> f = open('sample.txt', 'w')
>>> f
<_io.TextIOWrapper name='sample.txt' mode='w' encoding='UTF-8'>
>>> b = f.detach()
>>> b
<_io.BufferedWriter name='sample.txt'>
>>> f.write('hello')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ValueError: underlying buffer has been detached
>>>
```

äÿĂæŮęæŰ■åijĂæIJĂéąűåśĆåŘŐïijŇä¡ăářśåŔŕäžěçżŹè£ŤåŻđçżŞæđIJæůżåŁääÿĂäÿłæŰřçŽĎæIJĂéąűå

```
>>> f = io.TextIOWrapper(b, encoding='latin-1')
>>> f
<_io.TextIOWrapper name='sample.txt' encoding='latin-1'>
>>>
```

ář¡çőąåůšçżŔåŘŚä¡ăæijŤçd'žäžĘæŤźåŔŸçijŰçăĄçŽĎæŰźæşŢïijŇ
ä¡ĘæŸŕä¡ăè£ŸåŔŕäžěåĹl'çŤĺè£Źçğ■æŁĂæIJŕæİěæŤźåŔŸæŰĞäžűèąŇåd'ĎçŘĘãĂąéŤŹèŕŕæIJžåĹűäžěåŔŁæ

```
>>> sys.stdout = io.TextIOWrapper(sys.stdout.detach(), encoding=
↪'ascii',
...                                errors='xmlcharrefreplace')
>>> print('Jalape\u00f1o')
Jalape&#241;o
>>>
```

æşÍæĎŔäÿŃæIJĂåŘŐè¿ŞåĞžäÿ■çŽĎéİđASCIIå■Ůçňę Ãś æŸŕåęĆä¡Ţèćń ñ åŔŰäžčçŽĎãĂĆ

5.17 åřĘå■ŮèŁĆåĘŹåĚěæŰĞæIJňæŰĞäžű

éŮőécŸ

ä¡ăæČşåIJÍæŰĞæIJňæÍąåijŔæL'Şåij ĂçŽĎæŰĞäžűäÿ■åĘźåĚěåŐ§ågŃçŽĎå■ŮèŁĆæŢřæ■őãĂĆ

èğčåĘşæŰźæąĹ

åřĘå■ŮèŁĆæŢřæ■őçŻť'æŐěåĘźåĚěæŰĞäžűçŽĎçijŞåĘšåŇžå■şåŔříijŇä¿ŃåęĆïijŽ

```
>>> import sys
>>> sys.stdout.write(b'Hello\n')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: must be str, not bytes
>>> sys.stdout.buffer.write(b'Hello\n')
Hello
5
>>>
```

çśżäijijçŽĎïijŇèČ¡åď'§éĂŽè£ĞèŕżåŔŰæŰĞæIJňæŰĞäžűçŽĎ `buffer` åśđæĂğæÌěèŕżåŔŰäžŇè£ŻåĹűæŢřæ■őãĂĆ

èőÍèőž

I/Oçşżçż§äžěåśĆçžğçżŞæđĎçŽĎå¡ćåijŔæđĎåżžèĂŇæĹŘãĂĆ
æŰĞæIJňæŰĞäžűæŸŕéĂŽè£ĞåIJÍäÿĂäÿłæŃěæIJL'çijŞåĘšçŽĎäžŇè£ŻåĹűæÍąåijŔæŰĞäžűäÿŁåćđåŁääÿĂäÿ
`buffer` åśđæĂğæŇĞåŘŠåŕźåžŢçŽĎåžŢåśĆæŰĞäžűãĂĆåęĆæđIJä¡äçŻť'æŐěè£éŮőåőĆçŽĎèŕÌåŕšäijŽçŻŢè

æIJňåřŔèŁĆä¿Ńå■ŘåśŢçd'žçŽĎ `sys.stdout` åŔŕèČ¡çIJŃèţůæÌěæIJL'çĆźçL'źæőŁãĂĆ
éżŸèőd'æČĚåĘţäÿŃïijÑ`sys.stdout` æĂżæŸŕäžěæŰĞæIJňæÍąåijŔæL'Şåij ĂçŽĎãĂĆ
ä¡ĘæŸŕåęĆæđIJä¡ăåIJÍåĘŹäÿĂäÿłéIJĂèęĄæL'Şå■řäžŇè£ŻåĹűæŢřæ■őåĹræăĞåĞĘè¿ŞåĞžçŽĎèĎŽæIJňçŽĎ

5.18 åřĘæŰĞäžűæŔŔè£řçňęåŇĚèčĚæĹŘæŰĞäžűåŕźèśą

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłåŕźåžŢäžŐæŞ■ä¡IJçşżçż§äÿŁäÿĂäÿłåůšæL'ŞåijĂçŽĎI/OéĂŽéĄŞ(æŕŤåęĆæŰĞäžűãĂĄç
ä¡ăæČşåřĘåőĆåŇĚèčĚæĹŘäÿĂäÿłæŻť'énŸåśĆçŽĎPythonæŰĞäžűåŕźèśąãĂĆ

èğčåĘşæŰźæąĹ

äÿĂäÿłæŰĞäžűæŔŔè£řçňęåŠŇäÿĂäÿłæL'ŞåijĂçŽĎæŹőéĂŽæŰĞäžűæŸŕäÿ■äÿĂæăůçŽĎãĂĆ
æŰĞäžűæŔŔè£řçňęäžĚäžĚæŸŕäÿĂäÿłçŤśæŞ■ä¡IJçşżçż§æŇĞåőŽçŽĎæŢť'æŢřijŇçŤĺæÌěæŇĞäžčæ§Ŕäÿłçş
åęĆæđIJä¡äçćřåůğæIJL'è£ŹäźĹäÿĂäÿłæŰĞäžűæŔŔè£řçňęijŇä¡ăåŔŕäžěéĂŽè£Ğä¡£çŤĺ
`open()` åĞ¡æŢřæÌěåřĘåĚűåŇĚèčĚäÿžäÿĂäÿłPythonçŽĎæŰĞäžűåŕźèśąãĂĆ
ä¡ăäžĚäžĚåŔłéIJĂèęĄä¡£çŤĺè£ŹäÿłæŢť'æŢřåĂijçŽĎæŰĞäžűæŔŔè£řçňęä¡IJäÿžçňňäÿĂäÿłåŔĆæŢřæÌěäžčæŽ

```
# Open a low-level file descriptor
import os
fd = os.open('somefile.txt', os.O_WRONLY | os.O_CREAT)

# Turn into a proper file
f = open(fd, 'wt')
f.write('hello world\n')
f.close()
```

å¡Şéńŷåś ÇçŽĎæŰĞäžűåŕźèśąèćńåĚşéŮ■æĹŰèĂĚçăť åİŗçŽĎæŮűåĂŹïijŇåžŢåśÇçŽĎæŰĞäžűæŔŕè£ì
åęĆæđIJè£Źäÿłåźűäÿ■æŸŕä¡ăæČşèęĄçŽĎçżŞæđIJïijŇä¡ăåŔŕäžěçżŹ open()
åĞ¡æŢřäijăéĂŠäÿĂäÿłåŔŕéĂĽ'çŽĎ colsefd=False ãĂĆæŕŤåęĆïijŽ

```
# Create a file object, but don't close underlying fd when done
f = open(fd, 'wt', closefd=False)
...
```

èőĺèőž

åIJÍUnixçşžçż§äÿ■ïijŇè£Źçğ■åŇĚèčĚæŰĞäžűæŔŕè£řçňęçŽĎæŁĂæIJŕåŔŕäžěå¿ĹæŰźä¿£çŽĎåřĘäÿĂä
åęĆçőąéĄŞãĂĄåěŮæŐěå■Ůç■Ľ'ãĂĆäÿ¿ä¿ŃæİěèőšïijŇäÿŇéİćæŸŕäÿĂäÿłæŞ■ä¡IJçőąéĄŞçŽĎä¿Ńå■ŘïijŽ

```
from socket import socket, AF_INET, SOCK_STREAM

def echo_client(client_sock, addr):
    print('Got connection from', addr)

    # Make text-mode file wrappers for socket reading/writing
    client_in = open(client_sock.fileno(), 'rt', encoding='latin-1',
                     closefd=False)

    client_out = open(client_sock.fileno(), 'wt', encoding='latin-1
↪',
                      closefd=False)

    # Echo lines back to the client using file I/O
    for line in client_in:
        client_out.write(line)
        client_out.flush()

    client_sock.close()

def echo_server(address):
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(address)
    sock.listen(1)
    while True:
        client, addr = sock.accept()
        echo_client(client, addr)
```

éIJÄèęĄéĞ■çĆźåijžèřČçŽĎäÿĂçĆźæŸŕïijŇäÿŁéİćçŽĎä¿Ńå■ŘäżĚäżĚæŸŕäÿžäžĘæijŢçď žåĘĚç¡őçŽĎ
`open()` åĞ¡æŢřçŽĎäÿĂäÿłçL’źæĂğïijŇåźűäÿŤäź§åŔłéĂĆçŢĺäžŐå§žäžŐUnixçŽĎçşżçż§ãĂĆ
åęĆæđIJä¡ăæČşåřĘäÿĂäÿłçśżæŰĞäżűæŐěåŔčä¡IJçŢĺåIJĺäÿĂäÿłåěŮæŐěå■Ůåźűåÿ˝ŇæIJŻä¡ăçŽĎäżčçăĄåŔŕä
`makefile()` æŰźæşŢãĂĆ ä¡ĘæŸŕåęĆæđIJäÿ■èĂČèŹŚåŔŕçğżæď■æĂğçŽĎèŕÍïijŇéĆčäÿŁéİćçŽĎèğčåĘşæ
`makefile()` æĂğèČ¡æŻť åě¡äÿĂçĆźãĂĆ

ä¡ăäź§åŔŕäżěä¡£çŤÍè£Źçğ■æŁĂæIJŕæİěæđĎéĂăäÿĂäÿłåĹńåŘ■ïijŇåĚĄèőÿäżěäÿ■åŘŇäžŐçňňäÿĂæňąą
ä¿ŇåęĆïijŇäÿŇéİćæijŢçď žåęĆä¡ŢåĹŻåżžäÿĂäÿłæŰĞäżűåŕźèśąïijŇåőČåĚĄèőÿä¡äè¿ŞåĞžäžŇè£ŻåĹűæŢřæ■

```
import sys
# Create a binary-mode file for stdout
bstdout = open(sys.stdout.fileno(), 'wb', closefd=False)
bstdout.write(b'Hello World\n')
bstdout.flush()
```

åř¡çőąåŔŕäżěåřĘäÿĂäÿłåůšå■ŸåIJĺçŽĎæŰĞäżűæŔŘè£řçňęåŇĚèčĚæĹŘäÿĂäÿłæ■čåÿÿçŽĎæŰĞäżűåŕźè
ä¡ĘæŸŕèęĄæşÍæĎŘçŽĎæŸŕåźűäÿ■æŸŕæL’ĂæIJL’çŽĎæŰĞäżűæÍąåijŔéČ¡èćńæŤŕæŇĄïijŇåźűäÿŤæ§ŘäžŻç
(çL’źåĹńæŸŕæűL’åŘŁåĹřéŤŹèŕŕåď ĎçŘĘãĂĄæŰĞäżűçżŞář¿æİąäžűç■L’ç■L’çŽĎæŮűåĂŹ)ãĂĆ
åIJĺäÿ■åŘŇçŽĎæŞ■ä¡IJçşżçż§äÿŁè£Źçğ■èąŇäÿžäź§æŸŕäÿ■äÿĂæăůïijŇçL’źåĹńçŽĎïijŇäÿŁéİćçŽĎä¿Ńå■Ře
æĹŠèŕťäžĘè£ŹäźĹåď ŽïijŇæĎŘæĂİåŕśæŸŕèőľä¡ääĚĚåĹĘæţŃèŕŢèĞłåůśçŽĎåőđçŐŕäžčçăĄïijŇçąőä£İåőČèČ

5.19 åĹŻåżžäÿťæŮűæŰĞäżűåŠŇæŰĞäżűåď ź

éŮőécŸ

ä¡ăéIJĂèęĄåIJĺçÍŃåžŘæL’ğèąŇæŮűåĹŻåżžäÿĂäÿłäÿťæŮűæŰĞäżűæĹŰçŻőå¡ŢïijŇåźűåÿŇæIJŻä¡£çŤÍå

èğčåĘşæŰźæąĹ

`tempfile` æÍąåİŮäÿ■æIJL’å¿Ĺåď ŽçŽĎåĞ¡æŢřåŕŕäžěåőŇæĹŘè£ŹäżżåŁąãĂĆ
äÿžäžĘåĹŻåżžäÿĂäÿłåŇ£åŘ■çŽĎäÿťæŮűæŰĞäżűïijŇåŕŕäžěä¡£çŤÍ `tempfile.TemporaryFile` ïijŽ

```
from tempfile import TemporaryFile

with TemporaryFile('w+t') as f:
    # Read/write to the file
    f.write('Hello World\n')
    f.write('Testing\n')

    # Seek back to beginning and read the data
    f.seek(0)
    data = f.read()

# Temporary file is destroyed
```

æĹŰèĂĚïijŇåęĆæđIJä¡ăåŰIJæňćïijŇä¡ăè£ŸåŕŕäžěåČŔè£ŹæăůäĄ¡£çŤÍäÿťæŮűæŰĞäżűïijŽ

```
f = TemporaryFile('w+t')
# Use the temporary file
...
f.close()
# File is destroyed
```

TemporaryFile() çŽĎçňňäÿĂäÿłåŔĆæŢřæŸŕæŰĞäžűæĺąåijŔïijŇéĂŽåÿÿæİěèőšæŰĞæIJňæĺąåijŔä
w+t ïijŇäžŇè£ŻåĹűæĺąåijŔä¡£çŤĺ w+b ãĂĆ è£ŹäÿłæĺąåijŔåŘŇæŮűæŤŕæŇĄèŕžåŠŇåĘŹæŞ■ä¡IJïijŇåIJĺè£Ź
TemporaryFile() åŔęåď Úè£ŸæŤŕæŇĄèů§åĘĚç¡őçŽĎ open()
åĞ¡æŢřäÿĂæăůçŽĎåŔĆæŢřãĂĆæŕŤåęĆïijŽ

```
with TemporaryFile('w+t', encoding='utf-8', errors='ignore') as f:
    ...
```

åIJĺåď ğåď ŽæŢřUnixçşżçż§äÿŁïijŇéĂŽè£Ğ TemporaryFile()
åĹŻåżžçŽĎæŰĞäžűéČ¡æŸŕåŇ£åŘ■çŽĎïijŇçŤŽèĞşè£đçŻőå¡ŢéČ¡æšąæIJL’ãĂĆ
åęĆæđIJä¡ăæČşæL’Şçăť è£ŹäÿłéŹŘåĹűïijŇåŔŕäżěä¡£çŤĺ NamedTemporaryFile()
æİěäžčæŻ£ãĂĆæŕŤåęĆïijŽ

```
from tempfile import NamedTemporaryFile

with NamedTemporaryFile('w+t') as f:
    print('filename is:', f.name)
    ...

# File automatically destroyed
```

è£ŹéĞŇïijŇèćńæL’ŞåijĂæŰĞäžűçŽĎ f.name åšđæĂğåŇĚåŔńäžĘèŕěäÿť æŮűæŰĞäžűçŽĎæŰĞäžűåŘ
å¡Şä¡ăéIJĂèęĄåřĘæŰĞäžűåŘ■äijăéĂŠçżŹåĚűäžŰäžčçĂąæİěæL’ŞåijĂè£ŹäÿłæŰĞäžűçŽĎæŰűåĂŹïijŇè£Źä
åŠŇ TemporaryFile() äÿĂæăůïijŇçżŞæđIJæŰĞäžűåĚşéŮ■æŮűäijŽèćńèĞłåŁĺåĹăéŹđ’æŐL’ãĂĆ
åęĆæđIJä¡ääÿ■æČşè£ŹäźĹåĄŽïijŇåŔŕäžěäijăéĂŠäÿĂäÿłåĚşéŤőå■ŮåŔĆæŢř
delte=False å■şåŔŕãĂĆæŕŤåęĆïijŽ

```
with NamedTemporaryFile('w+t', delete=False) as f:
    print('filename is:', f.name)
    ...
```

äÿžäžĘåĹŻåżžäÿĂäÿłäÿť æŮűçŽőå¡ŢïijŇåŔŕäžěä¡£çŤĺ tempfile.
TemporaryDirectory() ãĂĆæŕŤåęĆïijŽ

```
from tempfile import TemporaryDirectory

with TemporaryDirectory() as dirname:
    print('dirname is:', dirname)
    # Use the directory
    ...
# Directory and all contents destroyed
```

èőÍèőž

TemporaryFile() ãĂĄNamedTemporaryFile() åŠŇ TemporaryDirectory() åĞ¡æŢř åžŤèřěæŸřåď ĎçŘĘäÿť æŮűæŰĞäžűçŽőå¡ŢçŽĎæIJĂçőĂå■ŢçŽĎæŰź åIJÍäÿĂäÿłæŻťä¡ŐçŽĎçžğåĹńïijŇä¡ååŔŕäžěä¡£çŤÍ mkstemp() åŠŇ mkdtemp() æÌěåĹŻåžžäÿť æŮűæŰĞäžűåŠŇçŽőå¡ŢãĂĆæŕŤåęĆïijŽ

```
>>> import tempfile
>>> tempfile.mkstemp()
(3, '/var/folders/7W/7WZl5sfZEF0pljrEB1UMWE+++TI/-Tmp-/tmp7fefhv')
>>> tempfile.mkdtemp()
'/var/folders/7W/7WZl5sfZEF0pljrEB1UMWE+++TI/-Tmp-/tmp5wvcv6'
>>>
```

ä¡ĘæŸŕïijŇè£ŹäžŻåĞ¡æŢřåźűäÿ■äijŽåĄŽè£ŻäÿĂæ■ěçŽĎçőąçŘĘäžĘãĂĆ ä¿ŇåęĆïijŇåĞ¡æŢř mkstemp() äżĚäžĚåřśè£ŤåŻđäÿĂäÿłåŐ§ågŇçŽĎOSæŰĞäžűæŔŔè£řçňęïijŇä¡ãéIJĂèę åŘŇæăůä¡ãè£ŸéIJĂèęĄèĞłåůśæÿĚçŘĘè£ŹäžŻæŰĞäžűãĂĆ

éĂŽåÿÿæÌěèőšïijŇäÿť æŮűæŰĞäžűåIJÍçşżçż§éżŸèőď çŽĎä¡■ç¡őèćńåĹŻåžžïijŇæŕŤåęĆ /var/tmp æĹŰçśżäijijçŽĎåIJřæŰźãĂĆ äÿžäžĘèŐůåŔŰçIJ§åőđçŽĎä¡■ç¡őïijŇåŔŕäžěä¡£çŤÍ tempfile.gettempdir() åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
>>> tempfile.gettempdir()
'/var/folders/7W/7WZl5sfZEF0pljrEB1UMWE+++TI/-Tmp-'
>>>
```

æĽĂæIJL'åŠŇäÿť æŮűæŰĞäžűçŻÿåĚşçŽĎåĞ¡æŢřéČ¡åĚĄèőÿä¡ăéĂŽè£Ğä¡£çŤÍåĚşéŤőå■ŮåŔĆæŢř prefix ãĂĄsuffix åŠŇ dir æÌěèĞłåőŽäźĽçŽőå¡ŢäžěåŔŁåŚ¡åŘ■èğĎåĹŹãĂĆæŕŤåęĆïijŽ

```
>>> f = NamedTemporaryFile(prefix='mytemp', suffix='.txt', dir='/tmp
↪')
>>> f.name
'/tmp/mytemp8ee899.txt'
>>>
```

æIJĂåŘŐè£ŸæIJL'äÿĂçĆźïijŇår¡åŔŕèČ¡äžěæIJĂåőL'åĚÍçŽĎæŰźåijŔä¡£çŤÍ tempfile æÍąåİŮæÌěåĹŻåžžäÿť æŮűæŰĞäžűãĂĆ åŇĚæŃňäžĚçżŹå¡ŞåL'■çŤÍæĹůæŐĹæİČèő£éŮőäžěåŔŁåIJÍæŰĞäžű èęĄæşÍæĎŔçŽĎæŸŕäÿ■åŘŇçŽĎåźşåŔŕåŔŕèČ¡äijŽäÿ■äÿĂæăůãĂĆåŽăæ■ď'ä¡ăæIJĂåě¡éŸĚèŕż åőŸæŰźæŰĞæąč æÌěäžĘèğčæŻť åď ŽçŽĎçżĘèŁĆãĂĆ

5.20 äÿŐäÿšèąŇçńŕåŔčçŽĎæŢřæ■őéĂŽä£ą

éŮőécŸ

ä¡ăæČşéĂŽè£ĞäÿšèąŇçńŕåŔčèŕżåĘŹæŢřæ■őïijŇåĚÿåđŃåIJžæŹŕåřśæŸŕåŠŇäÿĂäžŽçąňäžűěő¿åď ĞæĽ'Ş

èğčåĘşæŰźæąĹ

åř¡çőąä¡ääŔŕäžěéĂŽè£Ğä¡£çŤÍPythonåĘĚç¡őçŽĎI/OæÍąåİŮæİěåőŇæĹŘè£ŹäÿłäżżåŁąïijŇä¡Ęåŕźäž
pySerialåŇĚ ãĂĆ è£ŹäÿłåŇĚçŽĎä¡£çŤÍéİđåÿÿçőĂå■ŢïijŇåĚĹåőL'èčĚpySerialïijŇä¡£çŤÍçśžäijijäÿŃéİćè£Ź

```
import serial
ser = serial.Serial('/dev/tty.usbmodem641', # Device name varies
                    baudrate=9600,
                    bytesize=8,
                    parity='N',
                    stopbits=1)
```

èő¿åd'ĞåŘ■åŕźäžŐäÿ■åŘŇçŽĎèő¿åd'ĞåŠŇæŞ■ä¡IJçşżçż§æŸŕäÿ■äÿĂæăůçŽĎãĂĆ
æŕŤåęĆïijŇåIJÍWindowsçşżçż§äÿŁïijŇä¡ääŔŕäžěä¡£çŤÍ0, 1ç■L'èąÍçd'žçŽĎäÿĂäÿłèő¿åd'ĞæİěæL'ŞåijĂéĂŽä
äÿĂæŮęçńŕåŔčæL'ŞåijĂïijŇéĆčåŕšåŔŕäžěä¡£çŤÍ `read()`ïijŇ`readline()` åŠŇ `write()`
åĞ¡æŢřèŕżåĘŹæŢřæ■őäžĘãĂĆä¿ŃåęĆïijŽ

```
ser.write(b'G1 X50 Y50\r\n')
resp = ser.readline()
```

åd'ğåd'ŽæŢřæČĚåĘţäÿŃïijŇçőĂå■ŢçŽĎäÿšåŔčéĂŽä£ąäžŐæ■d'åŔŸå¿Ůå■ĄåĹĘçőĂå■ŢãĂĆ

èőÍèőž

åř¡çőąèąÍéİćäÿŁçIJŃèţůæİěå¿ĹçőĂå■ŢïijŇåĚűåőđäÿšåŔčéĂŽä£ąæIJL'æŮűåĂŹäź§æŸŕæŇžéžżçČęçŽĎ
æŐÍè■Řä¡ää¡£çŤÍçňňäÿL'æŰźåŇĚåęĆ `pySerial` çŽĎäÿĂäÿłåŐ§åŽăæŸŕåőČæŔŘä¿ŻäžĘåŕźéńŸçžğçL'źæĂ
(æŕŤåęĆèűĚæŮűïijŇæŐğåĹűæţĄïijŇçijŞåĘšåŇžåĹůæŰřïijŇæŔąæL'Ńå■Řèőőç■L'ç■L')ãĂĆäÿ¿äÿłä¿Ńå■Řïij
`RTS-CTS` æŔąæL'Ńå■Řèőőïij Ň ä¡ääŔłéIJĂèęĄçżŹ `Serial()` äijăéĂŠäÿĂäÿł
`rtscts=True` çŽĎåŔĆæŢřå■şåŔŕãĂĆ åĚűåőŸæŰźæŰĞæąčéİđåÿÿåőŇåŰĎïijŇåŽăæ■d'æĹŚåIJÍè£ŹéĞŇa

æŮűåĹżèőřä¡ŔæL'ĂæIJL'æűL'åŔŁåĹřäÿšåŔčçŽĎI/OéČ¡æŸŕäžŇè£ŽåĹűæÍąåijŔçŽĎãĂĆåŽăæ■d'ïijŇçą
(æĹŰæIJL'æŮűåĂŹæL'ğèąŇæŰĞæIJñçŽĎçijŰçăĄ/èğčçăĄæŞ■ä¡IJ)ãĂĆ
åŔęåd'Űå¡Şä¡ăéIJĂèęĄåĹŽåžžäžŇè£ŽåĹűçijŰçăĄçŽĎæŇĞäžd'æĹŰæŢřæ■őåŇĚçŽĎæŮűåĂŹïijŇ`struct`
æÍąåİŮäź§æŸŕéİđåÿÿæIJL'çŤÍçŽĎãĂĆ

5.21 åžŔåĹŮåŇŰPythonåŕźèšą

éŮőécŸ

ä¡ääIJĂèęĄåřĘäÿĂäÿłPythonåŕźèšąåžŔåĹŮåŇŰäÿžäÿĂäÿłå■ŮèŁĆæţĄïijŇäžěä¿£åřĘåőČä£Íå■ŸåĹřäÿĂ

èğčåĘşæŰźæąĹ

åŕźäžŐåžŔåĹŮåŇŰæIJĂæŹőéĄ■çŽĎåĄŽæşŢåřśæŸŕä¡£çŤÍ `pickle`
æÍąåİŮãĂĆäÿžäžĘåřĘäÿĂäÿłåŕźèšąä£Íå■ŸåĹřäÿĂäÿłæŰĞäžűäÿ■ïijŇåŔŕäžěè£ŹæăůåĄŽïijŽ

```
import pickle

data = ... # Some Python object
f = open('somefile', 'wb')
pickle.dump(data, f)
```

äÿžäžĘåřĘäÿĂäÿłåŕźèśąè¡ňåĆĺäÿžäÿĂäÿłå■ŮçňęäÿšïijŇåŔŕäżěä¡£çŤĺ pickle.dumps() ïijŽ

```
s = pickle.dumps(data)
```

äÿžäžĘäżŐå■ŮèŁĆæţĄäÿ■æĄćåd'■äÿĂäÿłåŕźèśąïijŇä¡£çŤĺ picle.load() æĹŰ pickle.loads() åĞ¡æŢřãĂĆæŕŤåęĆïijŽ

```
# Restore from a file
f = open('somefile', 'rb')
data = pickle.load(f)

# Restore from a string
data = pickle.loads(s)
```

èőĺèőž

åŕźäžŐåd'ğåd'ŽæŢřåžŤçŤĺçĺŃåžŔæİěèőšïijŇdump() åŠŇ load() åĞ¡æŢřçŽĎä¡£çŤĺåršæŸŕä¡ăæIJL'æŢĹä¡£çŤĺ pickle æĺąåİŮæĽ'ĂéIJĂçŽĎåĚíéČĺäžĘãĂĆ åőČåŔŕéĂĆçŤĺäžŐçżİåd'ğéČĺåĹĘPythonæŢřæ■őçśżåđŃåŠŇçŤĺæĹůèĞłåőŽäźĽ'çśżçŽĎåŕźèśąåőđä¿ŃãĂĆ åęĆæđIJä¡ăçćřåĹřæ§Řäÿłåžşåŕźäżěèől'ä¡ăåIJĺæŢřæ■őåžŞäÿ■ä‡İå■Ÿ/æĄćåd'■PythonåŕźèśąæĹŰèĂĚæŸŕéĂŽ éĆčäźĹå¿ĹæIJL'åŔŕèČ¡è£ŹäÿłåžŞçŽĎåžŢåśĆåršä¡£çŤĺäžĘ pickle æĺąåİŮãĂĆ

pickle æŸŕäÿĂçğ■PythonçL'źæIJL'çŽĎèĞłæŔŔè£řçŽĎæŢřæ■őçijŰçăĄãĂĆ éĂŽè£ĞèĞłæŔŔè£řïijŇèćńåžŔåĹŮåŇŰåŘŐçŽĎæŢřæ■őåŇĚåŘńæŕŔäÿłåŕźèśąåijĂåğŃåŠŇçżŞæİ§äžěåŔŁåć åŻăæ■d'ïijŇä¡ăæŮăéIJĂæŃĚå£Čåŕźèśąèőŕå¡ŢçŽĎåőŽäźL'ïijŇåőČæĂżæŸŕèČ¡åůë¡IJãĂĆ äÿ¿äÿłä¿Ńå■ŘïijŇåęĆæđIJèęĄåd'ĎçŘĘåd'ŽäÿłåŕźèśąïijŇä¡ääŔŕäżěè£ŹæăůåĄŽïijŽ

```
>>> import pickle
>>> f = open('somedata', 'wb')
>>> pickle.dump([1, 2, 3, 4], f)
>>> pickle.dump('hello', f)
>>> pickle.dump({'Apple', 'Pear', 'Banana'}, f)
>>> f.close()
>>> f = open('somedata', 'rb')
>>> pickle.load(f)
[1, 2, 3, 4]
>>> pickle.load(f)
'hello'
>>> pickle.load(f)
{'Apple', 'Pear', 'Banana'}
>>>
```

ä¡ãè£ŸèČ¡åžŔåĹŮåŇŰåĞ¡æŢřïijŇçśżïijŇè£ŸæIJL’æŐěåŔčïijŇä¡ĘæŸŕçżŞæđIJæŢřæ■őäżĚäżĚářĘåőČä

```
>>> import math
>>> import pickle.
>>> pickle.dumps(math.cos)
b'\x80\x03cmath\ncos\nq\x00.'
>>>
```

å¡ŞæŢřæ■őåŔ■åžŔåĹŮåŇŰåŻđæİěçŽĎæŮűåĂŹïijŇäijŽåĚĹåĄĞåőŽæL’ĂæIJL’çŽĎæžŘæŢřæ■őæŮűåŀ
æĺąåİŮãĂĄçśżåŠŇåĞ¡æŢřäijŽèĞłåŁĺæŇL’éIJĂåŕijåĚèè£ŻæİěãĂĆåŕźäžŐPythonæŢřæ■őèćńäÿ■åŘŇæIJžåŹĺ
æŢřæ■őçŽĎä£Ía■ŸåŔŕèČ¡äijŽæIJL’éŮőécŸïijŇåŻääÿžæL’ĂæIJL’çŽĎæIJžåŹĺéČ¡å£Ëéążèő£éŮőåŘŇäÿĂäÿł

æşÍ

```
å■ČäÿĞäÿ■èęĄåŕźäÿ■ä£ąäżżçŽĎæŢřæ■őä¡£çŤÍpickle.load()ãĂĆ
pickleåIJÍåŁăè¡¡æŮűæIJL’äÿĂäÿłåL’ŕä¡IJçŤÍåřśæŸŕåőČäijŽèĞłåŁÍåŁăè¡¡çŻÿåžŤæÍąåİŮåź
ä¡ĘæŸŕæ§ŘäÿłåİŕäžžåęĆæđIJç§ĚéĄ§pickleçŽĎåůëä¡IJåŐ§çŘĘïijŇ
äżŰåřśåŔŕäżěåĹżåżžäÿĂäÿłæĄűæĎŔçŽĎæŢřæ■őåŕijèĞt’PythonæL’ğèąŇéŽŔæĎŔæŇĞåőŽçŽĎçşżçż
åŽăæ■d’ïijŇäÿĂåőŽèęĄä£ÌèŕĄpickleåŘłåIJÍçŻÿäžŠäźŃéŮt’åŔŕäżěèőd’èŕĄåŕźæŰźçŽĎèğčæđŘ
```

æIJL’äžŻçśżåđŃçŽĎåŕźèśąæŸŕäÿ■èČ¡èćńåžŔåĹŮåŇŰçŽĎãĂĆè£ŹäžŻéĂŽåÿÿæŸŕéĆčäžŻä¿ÌèţŰåđ’Űé
æŕŤåęĆæL’ŞåijĂçŽĎæŰĞäżűïijŇç¡ŚçżIJè£đæŐěïijŇçž£çÍŃïijŇè£ŻçÍŃïijŇæăĹåÿğç■L’ç■L’ãĂĆ
çŤÍæĹůèĞłåőŽäźL’çśżåŔŕäżěéĂŽè£ĞæŔŘä¿Ż __getstate__()
åŠŇ __setstate__() æŰźæşŢæİěçżŢè£Ğè£ŹäžŻéŹŘåĹűãĂĆ
åęĆæđIJåőŽäźL’äžĘè£Źäÿd’äÿłæŰźæşŢïijŇpickle.dump()
åřśäijŽèřČçŤÍ __getstate__() èŐůåŔŰåžŔåĹŮåŇŰçŽĎåŕźèśąãĂĆ
çśżäijijçŽĎïijŇ__setstate__() åIJÍåŔ■åžŔåĹŮåŇŰæŮűèćńèřČçŤÍãĂĆäÿžäžĘæijŢçd’žè£Źäÿłåůëä¡IJåĆ
äÿŇéİćæŸŕäÿĂäÿłåIJÍåĘĚéČÍåőŽäźL’äžĘäÿĂäÿłçž£çÍŃä¡Ęäż■çĎűåŔŕäżěåžŔåĹŮåŇŰåŠŇåŔ■åžŔåĹŮåŇŰç

```
# countdown.py
import time
import threading

class Countdown:
    def __init__(self, n):
        self.n = n
        self.thr = threading.Thread(target=self.run)
        self.thr.daemon = True
        self.thr.start()

    def run(self):
        while self.n > 0:
            print('T-minus', self.n)
            self.n -= 1
            time.sleep(5)

    def __getstate__(self):
        return self.n

    def __setstate__(self, n):
        self.__init__(n)
```

èŕŢçİĂè£Řèą Ňäÿ Ńéİćç ŽĎåžŔåĹŮåŇŰèŕŢéłŇäžčçă Ąïij Ž

```
>>> import countdown
>>> c = countdown.Countdown(30)
>>> T-minus 30
T-minus 29
T-minus 28
...

>>> # After a few moments
>>> f = open('cstate.p', 'wb')
>>> import pickle
>>> pickle.dump(c, f)
>>> f.close()
```

çĎűåŘŐéĂĂåĞžPythonèğčæđŘåŹĺåźűéĞ■åŘŕåŘŐåĘ■èŕŢéłŇäÿŃïijŽ

```
>>> f = open('cstate.p', 'rb')
>>> pickle.load(f)
countdown.Countdown object at 0x10069e2d0>
T-minus 19
T-minus 18
...
```

ä¡ăåŔŕäžěçIJŃåĹřçž£çÍŃåŔĹåěĞè£źèĹńçŽĎéĞ■çŤ§äžĘïijŇäžŐä¡ăçńňäÿĂæňąåžŔåĹŮåŇŰåőČçŽĎåIJ

pickle åŕźäžŐåd'ğåđŃçŽĎæŢřæ■őçżŞæđĎæŕŤåęĆä¡£çŤÍ array æĹŰ numpy æĺąåİŮåĹŻåžžçŽĎäžŇè£ŻåĹűæŢřçżĎæŢĹçŐĞåźűäÿ■æŸŕäÿĂäÿłéńŸæŢĹçŽĎçijŰçăĄæŰźåijŔãĂĆ åęĆæđIJä¡ăéIJĂèęĄçğżåŁĺåd'ğéĞŔçŽĎæŢřçżĎæŢřæ■őïijŇä¡ăæIJĂåě¡æŸŕåĚĹåIJĺäÿĂäÿłæŰĞäźűäÿ■åřĘåĚ (éIJĂèęĄçňňäÿL'æŰźåžŞçŽĎæŤŕæŇĄ)ãĂĆ

çŤśäžŐ pickle æŸŕPythonçL'źæIJL'çŽĎåźűäÿŤéŹĎçİĂåIJĺæžŘçăĄäÿŁïijŇæL'ĂæIJL'åęĆæđIJéIJĂèę ä¿ŃåęĆïijŃåęĆæđIJæžŘçăĄåŔŸåŁĺäžĘïijŇä¡ăæL'ĂæIJL'çŽĎå■ŸåĆĺæŢřæ■őåŔŕèČ¡äijŽèćńçăt'åİŔåźűäÿŤåŔ åİęçŹ¡æİèèőšïijŇåŕźäžŐåIJĺæŢřæ■őåžŞåŠŇå■ŸæąćæŰĞäźűäÿ■å■ŸåĆĺæŢřæ■őæŮűïijŇä¡ăæIJĂåě¡ä¡£çŤĺæŻ è£ŹäžŽçijŰçăĄæäijåijŔæŻt'æăĞåĞĘïijŇåŔŕäžěèćńäÿ■åŘŇçŽĎèŕ■èĺĂæŤŕæŇĄïijŇåźűäÿŤäź§èČ¡å¿Ĺåě¡çŽĎ

æIJĂåŘŐäÿĂçĆźèęĄæşĺæĐŘçŽĎæŸŕ pickle æIJL'åd'ğéĞŔçŽĎéĚ■ç¡őéĂL'éąźåŠŇäÿĂäžŽæčŸæL'Ń åŕźäžŐæIJĂåÿÿèğĄçŽĎä¡£çŤĺåIJžæŹŕïijŇä¡ăäÿ■éIJĂèęĄåŐżæŃĚå£Čè£ŹäÿłïijŇä¡ĘæŸŕåęĆæđIJä¡ăèęĄåIJĺ æIJĂåě¡åŐżæ§ěéŸĚäÿĂäÿŃ åőŸæŰźæŰĞæąč ãĂĆ

çňňåĚ■çńäïijŽæŢřæ■őçijŰçăĄåŠŇåd'ĐçŘĘ

è£ŹäÿĂçńääÿżèęĄèőĺèőžä¡£çŤĺPythonåd'ĐçŘĘåŘĐçğ■äÿ■åŘŇæŰźåijŔçijŰçăĄçŽĐæŢřæ■őïijŇærŤåę åŠŇæŢřæ■őçżŞæđĐéĆčäÿĂçńääÿ■åŘŇçŽĐæŸŕïijŃè£Źçńääÿ■äijŽèőĺèőžçL'źæőŁçŽĐçőŮæşŢéŮőécŸïijŃè

Contents:

6.1 èŕžåĘŹCSVæŢřæ■ő

éŮőécŸ

ä¡ăæČşèŕżåĘŹäÿĂäÿłCSVæăijåijŔçŽĎæŰĞäžűãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐåď˘gåď˘ŽæŢřçŽĎCSVæăijåijŔçŽĎæŢřæ■őèŕżåĘŹéŮőécŸïijŇéČ¡åŔŕäžěä¡£çŤĺ csv åžŞãĂĆ ä¿ŃåęĆïijŽåĄĞèő¿ä¡ăåIJĺäÿĂäÿłåŘ■åŔństocks.csvæŰĞäžűäÿ■æIJL˘äÿĂäžŻèĆąçĕĺåÿĆåIJžæŢï

```
Symbol,Price,Date,Time,Change,Volume
"AA",39.48,"6/11/2007","9:36am",-0.18,181800
"AIG",71.38,"6/11/2007","9:36am",-0.15,195500
"AXP",62.58,"6/11/2007","9:36am",-0.46,935000
"BA",98.31,"6/11/2007","9:36am",+0.12,104800
"C",53.08,"6/11/2007","9:36am",-0.25,360900
"CAT",78.29,"6/11/2007","9:36am",-0.23,225400
```

äÿŃéIćåŘŚä¡ăåśŢçd˘žåęĆä¡ŢåřĘè£ŹäžŻæŢřæ■őèŕżåŔŰäÿžäÿĂäÿłåĚČçżĎçŽĎåžŔåĹŮïijŽ

```
import csv
with open('stocks.csv') as f:
    f_csv = csv.reader(f)
    headers = next(f_csv)
    for row in f_csv:
        # Process row
        ...
```

åIJĺäÿŁéIćçŽĎäžčçăĄäÿ■ïijŇ row äijŽæŸŕäÿĂäÿłåĹŮèąĺãĂĆåŻăæ■d˘ïijŇäÿžäžĘè©£éŮőæ§Řäÿłå■Ůæč row[0] è©£éŮőSymbolïijŇ row[4] è©£éŮőChangeãĂĆ

çŤśäžŐè£Źçğ■äÿŃæăĞèő£éŮőéĂŽåÿÿäijŽåijŢètůæůůæůĘïijŇä¡ăåŔŕäžěèĂČèŹŚä¡£çŤĺåŚ¡åŘ■åĚČçżĎå

```
from collections import namedtuple
with open('stock.csv') as f:
    f_csv = csv.reader(f)
    headings = next(f_csv)
    Row = namedtuple('Row', headings)
    for r in f_csv:
        row = Row(*r)
        # Process row
        ...
```

åőČåĚĄèőÿä¡ăä¡£çŤĺåĹŮåŘ■åęĆ row.Symbol åŠŇ row.Change äžčæŽ£äÿŃæăĞèő£éŮőãĂĆ éIJĂèęĄæşĺæĎŔçŽĎæŸŕè£ŹäÿłåŔłæIJL˘åIJĺåĹŮåŘ■æŸŕåŘĹæşŢçŽĎPythonæă ä¡ăåŔŕèČ¡éIJĂèęĄä£őæŤźäÿŃåŐ§ågŃçŽĎåĹŮåŘ■(åęĆåřĘéIđæăĞèŕĘçňęå■ŮçňęæŽ£æ■ćæĹŘäÿŃåĹŠçž£äź

åŔęåď˘ŰäÿĂäÿłéĂL˘æŃľ˘åřśæŸŕåřĘæŢřæ■őèŕżåŔŰåĹřäÿĂäÿłå■ŮåĚÿåžŔåĹŮäÿ■åŐżãĂĆåŔŕäžěè£ŹæăĂ

```
import csv
with open('stocks.csv') as f:
```

```
    f_csv = csv.DictReader(f)
    for row in f_csv:
        # process row
        ...
```

åIJÍè£ŹäýłçL'ĹæIJñäý■ïijŇä¡ăåŔŕäžěä¡£çŤĺåĹŮåŘ■åŐżèő£éŮőæŕŔäÿĂèąŇçŽĎæŢřæ■őäžĘãĂĆæŕŤåęĆ
æĹŰèĂĚ `row['Change']`

äÿžäžĘåĘŹåĚěCSVæŢřæ■őïijŇä¡ääż■çĎűåŔŕäžěä¡£çŤĺcsvæĺąåİŮïijŇäÿ■è£Ğè£ŹæŮűåĂŹåĚĹåĹŻåžžäý
`writer` åŕźèśąãĂĆä¿ŃåęĆ:

```
headers = ['Symbol','Price','Date','Time','Change','Volume']
rows = [('AA', 39.48, '6/11/2007', '9:36am', -0.18, 181800),
         ('AIG', 71.38, '6/11/2007', '9:36am', -0.15, 195500),
         ('AXP', 62.58, '6/11/2007', '9:36am', -0.46, 935000),
       ]

with open('stocks.csv','w') as f:
    f_csv = csv.writer(f)
    f_csv.writerow(headers)
    f_csv.writerows(rows)
```

åęĆædIJä¡ăæIJL'äýĂäýłå■ŮåĚÿåžŔåĹŮçŽĎæŢřæ■őïijŇåŔŕäžěåČŔè£ŹæăůåĄŽïijŽ

```
headers = ['Symbol', 'Price', 'Date', 'Time', 'Change', 'Volume']
rows = [{'Symbol':'AA', 'Price':39.48, 'Date':'6/11/2007',
        'Time':'9:36am', 'Change':-0.18, 'Volume':181800},
        {'Symbol':'AIG', 'Price': 71.38, 'Date':'6/11/2007',
        'Time':'9:36am', 'Change':-0.15, 'Volume': 195500},
        {'Symbol':'AXP', 'Price': 62.58, 'Date':'6/11/2007',
        'Time':'9:36am', 'Change':-0.46, 'Volume': 935000},
        ]

with open('stocks.csv','w') as f:
    f_csv = csv.DictWriter(f, headers)
    f_csv.writeheader()
    f_csv.writerows(rows)
```

èőÍèőž

ä¡ăåžŤèŕěæĂżæŸŕäijŸåĚĹéĂL'æŃľcsvæĺąåİŮåĹĘåL'šæĹŰèğčædŘCSVæŢřæ■őãĂĆä¿ŃåęĆïijŇä¡ăåŔŕ

```
with open('stocks.csv') as f:
for line in f:
    row = line.split(',')
    # process row
    ...
```

ä¡£çŤĺè£Źçğ■æŰźåijŔçŽĎäÿĂäýłçijžçĆźåřśæŸŕä¡ääż■çĎűéIJĂèęĄåŐżåď'ĎçŘĘäÿĂäžŽæčŸæL'ŃçŽĎç

æŕŤåęĆïijŇåęĆæđIJæ§ŘäžŻå■Ůæőţ̊åĂijèćńåijŢåŔůåŇĚåŻt’ïijŇä¡ăäÿ■å¿Ůäÿ■åŐżéŹđ’è£ŹäžŻåijŢåŔůãĂĆ
åŘęåđ’ŰïijŇåęĆæđIJäÿĂäÿłèćńåijŢåŔůåŇĚåŻt’çŽĎå■Ůæőţçćřåůğå̊ŕńæIJL’äÿĂäÿłéĂŮåŔůïijŇéĆčäźĹçĺŃåž

éżŸèőđ’æČĚåĘţäÿŃïijŇcsv åžŞåŔŕèŕĘåĹńMicrosoft Ex-
celæL’Ăä¡£çŤĺçŽĎCSVçijŰçăĄèğĎåĹŹãĂĆ è£ŹæĹŰèőÿäź§æŸŕæIJĂåÿÿèğĄçŽĎå¡ćåijŔïijŇåźűäÿŤäź§äijŽ
çĎűèĂŇïijŇåęĆæđIJä¡ăæ§ěçIJŃcsvçŽĎæŰĞæąčïijŇåřśäijŽåŔŠçŐřæIJL’å¿Ĺåđ’Žçğ■æŰźæşŢåřĘåőČåžŤçŤĺ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăæČşèŕżåŔŰäžětabåĹĘåL’šçŽĎæŢŕæ■őïijŇåŔŕäžěè£ŹæăůåĄŽïijŽ

```
# Example of reading tab-separated values
with open('stock.tsv') as f:
    f_tsv = csv.reader(f, delimiter='\t')
    for row in f_tsv:
        # Process row
        ...
```

åęĆæđIJä¡ăæ■čåIJĺèŕżåŔŰCSVæŢŕæ■őåźűåřĘåőČäžňè¡ňæ■ćäÿžåŠ¡åŘ■åĚČçżĎïijŇéIJĂèęĄæşĺæĎŔåŕź
ä¿ŃåęĆïijŇäÿĂäÿłCSVæăijåijŔæŰĞäžűæIJL’äÿĂäÿłåŇĚåŘńéİđæşŢæăĞèŕĘçňęçŽĎåĹŮåđ’t’èąŇïijŇçśžäijijž

```
StreetÂăAddress,Num-Premises,Latitude,Longitude 5412ÂăNÂăCLARK,10,
↪41.980262,-87.668452
```

è£ŹæăůæIJĂçżĹäijŽåŕijèĞt’åIJĺåĹŻåžžäÿĂäÿłåŚ¡åŘ■åĚČçżĎæŮűäžğçŤ§äÿĂäÿł
`ValueError` åijĆåÿÿèĂŇåđ’śèt’ěãĂĆ äÿžäžĘèğčåĘşè£ŹéŮőécŸïijŇä¡ääŔŕèČ¡äÿ■å¿Ůäÿ■åĚĹåŐżä£őæ■čå
ä¿ŃåęĆïijŇåŔŕäžěåČŔäÿŃéİcè£ŹæăůåIJĺéİđæşŢæăĞèŕĘçňęäÿŁä¡£çŤĺäÿĂäÿłæ■čåĹŹèąĺè¿¿åijŔæŻ£æ■ćïijŽ

```
import re
with open('stock.csv') as f:
    f_csv = csv.reader(f)
    headers = [ re.sub('[^a-zA-Z_]', '_', h) for h in next(f_csv) ]
    Row = namedtuple('Row', headers)
    for r in f_csv:
        row = Row(*r)
        # Process row
        ...
```

è£ŸæIJL’éĞ■èęĄçŽĎäÿĂçĆźéIJĂèęĄåijžèřČçŽĎæŸŕïijŇcsväžğçŤ§çŽĎæŢŕæ■őéČ¡æŸŕå■Ůçňęäÿšçśżå
åęĆæđIJä¡ăéIJĂèęĄåĄŽè£ŹæăůçŽĎçśžåđŃè¡ňæ■ćïijŇä¡ää£ËéąžèĞłåůśæL’ŃåŁĺåŐżåőđçŐřãĂĆ
äÿŃéİcæŸŕäÿĂäÿłåIJĺCSVæŢŕæ■őäÿŁæL’ğèąŇåĚűäžŰçśžåđŃè¡ňæ■ćçŽĎä¿Ńå■ŘïijŽ

```
col_types = [str, float, str, str, float, int]
with open('stocks.csv') as f:
    f_csv = csv.reader(f)
    headers = next(f_csv)
    for row in f_csv:
        # Apply conversions to the row items
        row = tuple(convert(value) for convert, value in zip(col_
↪types, row))
        ...
```

åŘęåđ’ŰïijŇäÿŃéİcæŸŕäÿĂäÿłè¡ňæ■ćå■ŮåĚÿäÿ■çL’źåőŽå■Ůæőţ̊çŽĎä¿Ńå■ŘïijŽ

```
print('Reading as dicts with type conversion')
field_types = [ ('Price', float),
                ('Change', float),
                ('Volume', int) ]

with open('stocks.csv') as f:
    for row in csv.DictReader(f):
        row.update((key, conversion(row[key]))
                   for key, conversion in field_types)
        print(row)
```

éĂŽåÿÿæÌèèőšïijÑä¡ãåŔŕèČ¡åźűäÿ■æČşè£ĞåďŽåŐżèĂČèŹŚè£ŹäžŽè¡ñæ■ćéŮőécŸãĂĆ
åIJĺåőđéŹĚæČĚåĘţäÿ■ïijÑCSVæŰĞäżűéČ¡æĹŰåďŽæĹŰåřŚæIJL'äžŽçijžåď śçŽĎæŢřæ■őïijÑècńçăt'åİŔçŽĎ
åŽäæ■d'ïijÑéŹď'éİđä¡ãçŽĎæŢřæ■őçąőåőđæIJL'ä£ÌéŽIJæŸŕåĞĘçąőæŮäèŕŕçŽĎïijÑåŘęåĹŹä¡ãå£ĚéążèĂČèŹ

æIJĂåŘŐïijÑåęĆæđIJä¡ãèŕżåŔŰCSVæŢřæ■őçŽĎçŻőçŽĎæŸŕåĄŽæŢřæ■őåĹĘæđŘåŠÑçż§èőąçŽĎèŕİïij
ä¡ãåŔŕèČ¡éIJĂèęĄçIJÑäÿĂçIJÑ `Pandas` åÑĚãĂĆPandas
åÑĚåŘńäžĘäÿĂäÿłéİđåÿÿæŰźä¿£çŽĎåĞ¡æŢřåŔń `pandas.read_csv()`
ïijÑ åőČåŔŕäžěåŁäè¡¡CSVæŢřæ■őåĹřäÿĂäÿł `DataFrame` åŕźèśąäÿ■åŐżãĂĆ
çĎűåŘŐåĹl'çŤÍè£Źäÿłåŕźèśąä¡ãåřśåŔŕäžěçŤ$æĹŘåŘĎçğ■å¡ćåijŔçŽĎçż§èőąãĂĄè£ĞæżđæŢřæ■őäžěåŘŁæI
åIJĺ6.13åřŔèŁĆäÿ■äijŽæIJL'è£ŹæăůäÿĂäÿłä¿Ńå■ŘãĂĆ

6.2 èŕżåĘŹJSONæŢřæ■ő

éŮőécŸ

ä¡ãæČşèŕżåĘŹJSON(JavaScript Object Notation)çijŰçăĄæäijåijŔçŽĎæŢřæ■őãĂĆ

èğčåĘşæŰźæąĹ

`json` æĺąåİŮæŔŘä¿ŻäžĘäÿĂçğ■å¿ĹçőĂå■ŢçŽĎæŰźåijŔæİěçijŰçăĄåŠÑèğčçăĄJSONæŢřæ■őãĂĆ
åĚűäÿ■äÿđ'äÿłäÿżèęĄçŽĎåĞ¡æŢřæŸŕ `json.dumps()` åŠÑ `json.loads()`
ïijÑ èęĄæŕŤåĚűäžŮåžŔåĹŮåŇŰåĞ¡æŢřåžŞåęCpickleçŽĎæŐěåŘčåřŚå¿Ůåď'ŽãĂĆ
äÿŃéİćæijŤçď'žåęĆä¡ŢåřĘäÿĂäÿłPythonæŢřæ■őçżŞæđĎè¡ñæ■ćäÿžJSONïijŽ

```
import json

data = {
    'name' : 'ACME',
    'shares' : 100,
    'price' : 542.23
}

json_str = json.dumps(data)
```

äÿŃéİćæijŤçď'žåęĆä¡ŢåřĘäÿĂäÿłJSONçijŰçăĄçŽĎå■Ůçñęäÿšè¡ñæ■ćåŻđäÿĂäÿłPythonæŢřæ■őçżŞæđÌ

```
data = json.loads(json_str)
```

åęĆæđIJä¡ăèęĄåď'ĎçŘĘçŽĎæŸŕæŰĞäžűèĂŇäÿ■æŸŕå■ŮçňęäÿšïijŇä¡ăåŔŕäžěä¡£çŤĺ
`json.dump()` åŠŇ `json.load()` æİěçijŰçăĄåŠŇèğčçăĄJSONæŢřæ■őãĂĆä¿ŃåęĆïijŽ

```
# Writing JSON data
with open('data.json', 'w') as f:
    json.dump(data, f)

# Reading data back
with open('data.json', 'r') as f:
    data = json.load(f)
```

èőĺèőž

JSONçijŰçăĄæŤŕæŇĄçŽĎå§žæIJňæŢřæ■őçśžåđŃäÿž `None` ïijŇ `bool` ïijŇ `int` ïijŇ `float` åŠŇ `str` ïijŇ äžěåŔŁåŇĚåŘńè£ŹäžŻçśžåđŃæŢřæ■őçŽĎlistsïijŇtuplesåŠŇdictionariesãĂĆ åŕźäžŐdictionariesïijŇkeyséIJĂèęĄæŸŕå■ŮçňęäÿšçśžåđŃ(å■ŮåĚÿäÿ■äżżä¡Ţéİđåđ■ŮçňęäÿšçśžåđŃçŽĎkeyåI äÿžäžĘéĄţå¿ŁJSONèğĎèŇČïijŇä¡ăåžŤèŕěåŔŁçijŰçăĄPythonçŽĎlistsåŠŇdictionariesãĂĆ èĂŇäÿŤïijŇåIJÍwebåžŢçŤÍçÍŃåžŔäÿ■ïijŇéąűåśĆåŕźèśąèćńçijŰçăĄäÿžäÿĂäÿłå■ŮåĚÿæŸŕäÿĂäÿłæăĞåĞĘåA

JSONçijŰçăĄçŽĎæăijåijŔåŕźäžŐPythonèŕ■æşŢèĂŇåůšåĞääźŐæŸŕåőŇåĚÍäÿĂæăůçŽĎïijŇéŹď'äžĘäÿĂ æŕŤåęĆïijŇTrueäijŽèćńæŸĂăřĎäÿžtrueïijŇFalseèćńæŸĂăřĎäÿž-falseïijŇèĂŇNoneäijŽèćńæŸĂăřĎäÿžnullãĂĆ äÿŃéİćæŸŕäÿĂäÿłą¿Ńå■ŘïijŇæijŤçď'žäžĘçijŰçăĄåŘŐçŽĎå■

```
>>> json.dumps(False)
'false'
>>> d = {'a': True,
...      'b': 'Hello',
...      'c': None}
>>> json.dumps(d)
'{"b": "Hello", "c": null, "a": true}'
>>>
```

åęĆæđIJä¡ăèŕŢçİĂåŐżæčĂæ§ěJSONèğčçăĄåŘŐçŽĎæŢřæ■őïijŇä¡ăéĂŽåÿÿå¿ĹéŽ¿éĂŽè£ĞçőĂå■ŢçŽĺ çĽźåĹńæŸŕå¡ŞæŢřæ■őçŽĎåţŇåěŰçżŞæđĎåśĆæňąå¿ĹæůśæĹŰèĂĚåŇĚåŘńåď'ğéĞŔçŽĎå■ŮæőţæŰűãĂĆ äÿžäžĘèğčåĘşè£ŹäÿłéŮőéćŸïijŇåŔŕäžěèĂČèŹŚä¡£çŤĺpprintæĺąåİŰçŽĎ `pprint()` åĞ¡æŢřæİěäžčæŻ£æŹőéĂŽçŽĎ `print()` åĞ¡æŢřãĂĆ åőČäijŽæŇĽçĚğkeyçŽĎå■Ůæí■éąžäžŔåźűäžěäÿĂçğ■æŻť'åŁăç¿ÕèğĆçŽĎæŰźåijŔè¿ŞåĞžãĂĆ äÿŃéİćæŸŕäÿĂäÿłæijŤçď'žåęĆä¡ŢæijĆäžőçŽĎæĽŞå■ŕè¿ŞåĞžTwitteräÿŁæŘIJçť'ćçżŞæđIJçŽĎä¿Ńå■ŘïijŽ

```
>>> from urllib.request import urlopen
>>> import json
>>> u = urlopen('http://search.twitter.com/search.json?q=python&
 ↪rpp=5')
>>> resp = json.loads(u.read().decode('utf-8'))
>>> from pprint import pprint
>>> pprint(resp)
{'completed_in': 0.074,
```

```
'max_id': 264043230692245504,
'max_id_str': '264043230692245504',
'next_page': '?page=2&max_id=264043230692245504&q=python&rpp=5',
'page': 1,
'query': 'python',
'refresh_url': '?since_id=264043230692245504&q=python',
'results': [{'created_at': 'Thu, 01 Nov 2012 16:36:26 +0000',
             'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:14 +0000',
             'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:13 +0000',
             'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:07 +0000',
             'from_user': ...
             }
             {'created_at': 'Thu, 01 Nov 2012 16:36:04 +0000',
             'from_user': ...
             }],
'results_per_page': 5,
'since_id': 0,
'since_id_str': '0'}
>>>
```

äÿĂèĹñæİèèőšïijŇJSONèğčçăĄäijŽæăźæ■őæŔŘä¿ŻçŽĎæŢřæ■őåĹŻåžždictsæĹŰlistsãĂĆåęĆæđIJä¡ăæČşèęĄåĹŻåžžåĚűäžŰçśżåđŃçŽĎåŕźèśąïijŇåŘŕäžěçżŹ `json.loads()` äijăéĂŠobject_pairs_hookæĹŰobject_hookåŘĆæŢřãĂĆä¿ŃåęĆïijŇäÿŃéİćæŸŕæijŤçd'žåęĆä¡Ţèğčçă ĄJSONæŢřæ■őåźűåIJĺäÿĂäÿłOrderedDictäÿ■ä£İçŢŹåĚűéąžåžŔ

```
>>> s = '{"name": "ACME", "shares": 50, "price": 490.1}'
>>> from collections import OrderedDict
>>> data = json.loads(s, object_pairs_hook=OrderedDict)
>>> data
OrderedDict([('name', 'ACME'), ('shares', 50), ('price', 490.1)])
>>>
```

äÿŃéİćæŸŕåęĆä¡ŢåřĘäÿĂäÿłJSONå■ŮåĚÿè¡ňæ■ćäÿžäÿĂäÿłPythonåŕźèśąä¿Ńå■ŘïijŽ

```
>>> class JSONObject:
...     def __init__(self, d):
...         self.__dict__ = d
...
>>>
>>> data = json.loads(s, object_hook=JSONObject)
>>> data.name
'ACME'
>>> data.shares
50
```

```
>>> data.price
490.1
>>>
```

æIJĂåŘŐäÿĂäÿłä¿Ńå■Řäÿ■ïijŇJSONèğčçăĄåŘŐçŽĎå■ŮåĚÿä¡IJäÿžäÿĂäÿłå■Ţäÿłå
__init__() ãĂĆ çĎűåŘŐïijŇä¡ăåřśåŔŕäżěéŽŔå£ČæĽĂæňšçŽĎä¡£çŤĺåőČäžĘïijŇærŤåęĆä¡IJäÿžäÿĂäÿłå

åIJÍçijŰçăĄJSONçŽĎæŮűåĂŹïijŇè£ŸæIJL'äÿĂäžŻéĂĽ'éąźå¿ĹæIJL'çŤĺãĂĆ
åęĆædIJä¡ăæČşèŐůå¿ŮæijĆäžőçŽĎæăijåijŔåŇŰå■Ůçňęäÿšå ŘŐè¿ŞåĞžïijŇåŔŕäżěä¡£çŤĺ
json.dumps() çŽĎindentåŔĆæŢřãĂĆ åőČäijŽä¡£å¿Ůè¿ŞåĞžåŠŇpprint()åĞ¡æŢřæŢĹædIJçśäijijãĂĆærĹ

```
>>> print(json.dumps(data))
{"price": 542.23, "name": "ACME", "shares": 100}
>>> print(json.dumps(data, indent=4))
{
    "price": 542.23,
    "name": "ACME",
    "shares": 100
}
>>>
```

åŕźèśąåőđä¿ŃéĂŽåÿÿåźűäÿ■æŸŕJSONåŔŕåžŔåĹŮåŇŰçŽĎãĂĆä¿ŃåęĆïijŽ

```
>>> class Point:
...     def __init__(self, x, y):
...         self.x = x
...         self.y = y
...
>>> p = Point(2, 3)
>>> json.dumps(p)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "/usr/local/lib/python3.3/json/__init__.py", line 226, in␣
↪dumps
        return _default_encoder.encode(obj)
    File "/usr/local/lib/python3.3/json/encoder.py", line 187, in␣
↪encode
        chunks = self.iterencode(o, _one_shot=True)
    File "/usr/local/lib/python3.3/json/encoder.py", line 245, in␣
↪iterencode
        return _iterencode(o, 0)
    File "/usr/local/lib/python3.3/json/encoder.py", line 169, in␣
↪default
        raise TypeError(repr(o) + " is not JSON serializable")
TypeError: <__main__.Point object at 0x1006f2650> is not JSON␣
↪serializable
>>>
```

åęĆædIJä¡ăæČşåžŔåĹŮåŇŰåŕźèśąåőđä¿ŃïijŇä¡ăåŔŕäżěæŔŘä¿ŻäÿĂäÿłåĞ¡æŢřijŇåőČçŽĎè¿ŞåĚěæŸŕ

```
def serialize_instance(obj):
    d = { '__classname__' : type(obj).__name__ }
    d.update(vars(obj))
    return d
```

åęĆædIJä¡ăæČşåŔ■è£ĞæİěèŐůåŔŰè£Źäÿłåőđä¿ŃïijŇåŔŕäžěè£ŹæăůåĄŽïijŽ

```
# Dictionary mapping names to known classes
classes = {
    'Point' : Point
}

def unserialize_object(d):
    clsname = d.pop('__classname__', None)
    if clsname:
        cls = classes[clsname]
        obj = cls.__new__(cls) # Make instance without calling __
↪init__
        for key, value in d.items():
            setattr(obj, key, value)
        return obj
    else:
        return d
```

äÿŃéİćæŸŕåęĆä¡Ţä¡£çŤÍè£ŹäžŻåĞ¡æŢřçŽĎä¿Ńå■ŘïijŽ

```
>>> p = Point(2,3)
>>> s = json.dumps(p, default=serialize_instance)
>>> s
'{"__classname__": "Point", "y": 3, "x": 2}'
>>> a = json.loads(s, object_hook=unserialize_object)
>>> a
<__main__.Point object at 0x1017577d0>
>>> a.x
2
>>> a.y
3
>>>
```

json æĺąåİŮè£ŸæIJL'å¿Ĺåd'ŽåĚűäžŰéĂL'éąźæİěæŐğåĹűæŻt'ä¡ŐçžğåĹńçŽĎæŢřå■ŮãĂĄçL'źæőŁåĂi
åŔŕäžěåŔĆèĂČåőŸæŰźæŰĞæąčèŐůåŔŰæŻt'åd'ŽçżĘèŁĆãĂĆ

6.3 èğčædŘçőĂå■ŢçŽĎXMLæŢřæ■ő

éŮőécŸ

ä¡ăæČşäžŐäÿĂäÿłçőĂå■ŢçŽĎXMLæŰĞæąčäÿ■æŔŘåŔŰæŢřæ■őãĂĆ

èğčåĘşæŰźæąĹ

åŘŕäžěä¡£çŤÍ xml.etree.ElementTree æĺąåİŮäžŐçőĂå■ŢçŽĎXMLæŰĞæąčäÿ■æŘŘåŘŰæŢřæ■őãĂĆ äÿžäžĘæijŤçď žïijŇåĄĞèő¿ä¡ăæČşèğčæđŘPlanet PythonäÿŁçŽĎRSSæžŘãĂĆäÿŃéİćæŸŕçŻÿåžŤçŽĎäžčçăĄïijŽ

```
from urllib.request import urlopen
from xml.etree.ElementTree import parse

# Download the RSS feed and parse it
u = urlopen('http://planet.python.org/rss20.xml')
doc = parse(u)

# Extract and output tags of interest
for item in doc.iterfind('channel/item'):
    title = item.findtext('title')
    date = item.findtext('pubDate')
    link = item.findtext('link')

    print(title)
    print(date)
    print(link)
    print()
```

è£ŘèąŇäÿŁéİćçŽĎäžčçăĄïijŇè¿ŞåĞžçżŞæđIJçśżäijijè£ŹæăůïijŽ

```
Steve Holden: Python for Data Analysis
Mon, 19 Nov 2012 02:13:51 +0000
http://holdenweb.blogspot.com/2012/11/python-for-data-analysis.html

Vasudev Ram: The Python Data model (for v2 and v3)
Sun, 18 Nov 2012 22:06:47 +0000
http://jugad2.blogspot.com/2012/11/the-python-data-model.html

Python Diary: Been playing around with Object Databases
Sun, 18 Nov 2012 20:40:29 +0000
http://www.pythondiary.com/blog/Nov.18,2012/been-...-object-
 ↪databases.html

Vasudev Ram: Wakari, Scientific Python in the cloud
Sun, 18 Nov 2012 20:19:41 +0000
http://jugad2.blogspot.com/2012/11/wakari-scientific-python-in-
 ↪cloud.html

Jesse Jiryu Davis: Toro: synchronization primitives for Tornado␣
 ↪coroutines
Sun, 18 Nov 2012 20:17:49 +0000
http://feedproxy.google.com/~r/EmptysquarePython/~3/_DOZT2Kd0hQ/
```

å¿ĹæŸ¿çĎűïijŇåęĆæđIJä¡ăæČşåĄŽè£ŻäÿĂæ■ěçŽĎåď ĎçŘĘïijŇä¡ăéIJĂèęĄæŻ£æ■ć `print()` èŕ■åŘěæİěåőŇæĹŘåĚűäžŰæIJL'èűčçŽĎäžŇãĂĆ

èőÍèőž

åIJÍå¿Ĺåď ŽåžŤçŤÍçÍŃåžŔäÿ■åď ĎçŘĘXMLçijŰçăĄæăijåijŔçŽĎæŢřæ■őæŸŕå¿Ĺåÿÿè ğĄçŽĎãĂĆ äÿ■äžĚåŻääÿžXMLåIJÍInternetäÿŁéÍćåůšçżŔèćńáź£æşŽåžŤçŤÍäžÓæŢřæ■őäžď æ■ćïijŇ åŘŇæŮűåőČäź§æŸŕäÿĂçğ■å■ŸåĆÍåžŤçŤÍçÍŃåžŔæŢřæ■őçŽĎåÿÿçŤÍæăijåijŔ(æŕŤåęĆå■Ůåď ĎçŘĘïijŇé§§ä æŐěäÿŃæÍěçŽĎèőÍèőžäijŽåĚĹåĄĞåőŽèŕżèĂĚåůšçżŔåŕźXMLå§žçąĂæŕŤè¿ČçĘ§æĆL’äžĘãĂĆ

åIJÍå¿Ĺåď ŽæČĚåĘţäÿŃïijŇå¡Şä¡£çŤÍXMLæÍěäžĚäžĚå■ŸåĆÍæŢřæ■őçŽĎæŮűåĂŹïijŇåŕźåžŤçŽĎæŰĞ ä¿ŃåęĆïijŇäÿŁéÍćä¿Ńå■Řäÿ■çŽĎRSSèőćéŸĚæžŘçśżäijijäžŐäÿŃéÍćçŽĎæäijåijŔïijŽ

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:dc="http://purl.org/dc/elements/1.1/">
    <channel>
        <title>Planet Python</title>
        <link>http://planet.python.org/</link>
        <language>en</language>
        <description>Planet Python - http://planet.python.org/</
↪description>
        <item>
            <title>Steve Holden: Python for Data Analysis</title>
            <guid>http://holdenweb.blogspot.com/...-data-analysis.
↪html</guid>
            <link>http://holdenweb.blogspot.com/...-data-analysis.
↪html</link>
            <description>...</description>
            <pubDate>Mon, 19 Nov 2012 02:13:51 +0000</pubDate>
        </item>
        <item>
            <title>Vasudev Ram: The Python Data model (for v2 and␣
↪v3)</title>
            <guid>http://jugad2.blogspot.com/...-data-model.html</
↪guid>
            <link>http://jugad2.blogspot.com/...-data-model.html</
↪link>
            <description>...</description>
            <pubDate>Sun, 18 Nov 2012 22:06:47 +0000</pubDate>
        </item>
        <item>
            <title>Python Diary: Been playing around with Object␣
↪Databases</title>
            <guid>http://www.pythondiary.com/...-object-databases.
↪html</guid>
            <link>http://www.pythondiary.com/...-object-databases.
↪html</link>
            <description>...</description>
            <pubDate>Sun, 18 Nov 2012 20:40:29 +0000</pubDate>
        </item>
        ...
    </channel>
</rss>
```

`xml.etree.ElementTree.parse()` åĞ¡æŢřè ğčæđŘæŢt'äÿłXMLæŰĞæąčåźűåřĘåĚűè¡ňæ■ćæĹŘ
çĎűåŘŐïijŇä¡ăářśèČ¡ä¡£çŤÍ `find()` ãĂĄ`iterfind()` åŠŇ
`findtext()` ç■L'æŰźæşŢæİěæŘIJçt'ćçL'źåőŽçŽĎXMLåĚČçt'äážĘãĂĆ
è£ŹäžŻåĞ¡æŢřçŽĎåŔĆæŢřářśæŸŕæ§Řäÿłæ ŇĞåőŽçŽĎæăĞç■¿åŘ■ïijŇä¿ŇåęĆ `channel/item` æĹŰ `title` ãĂĆ

æŕŔæňąæŇĞåőŽæ§Řäÿłæă Gç■¿æŮűïijŇä¡ăéIJĂèęĄéĄ■åŐĘæŢt'äÿłæŰĞæąčçżŞædĎãĂĆæŕŔæňąæŘIJ
åŘŇæăůïijŇæŕŔæňąæŞ■ä¡IJæĽĂæŇĞåőŽçŽĎæăĞç■¿åŘ■äź§æŸŕè ţůåğŃåĚČçt'ăçŽĎçŽÿářźèůřå¿ĎãĂĆ
ä¿ŇåęĆïijŇæĽ'ğèąŇ `doc.iterfind('channel/item')`
æİěæŘIJçt'ćæĽĂæIJL'åIJÍ `channel` åĚČçt'ääÿŃéÍćçŽĎ `item` åĚČçt'ăãĂĆ
`doc` äžčèąÍæŰĞæąčçŽĎæIJĂéąűåśĆ(äź§åřśæŸŕçňňäÿĂçžğçŽĎ `rss` åĚČçt'ă)ãĂĆ
çĎűåŘŐæŐěäÿŃæİěçŽĎèřČçŤÍ `item.findtext()` äijŽäžŐåůšæĽ¿åĹřçŽĎ `item`
åĚČçt'ăä¡■ç¡őåijĂåğŃæŘIJçt'ćãĂĆ

`ElementTree` æÍąåİŮäÿ■çŽĎæŕŔäÿłåĚČçt'ăæIJL'äÿĂäžŻéĞ■èęĄçŽĎåśđæĂğåŠŇæŰźæşŢïijŇåIJÍèğč
`tag` åśđæĂğåŃĚåŘńäžĘæăĞç■¿çŽĎåŘ■å■ŮïijŇ`text` åśđæĂğåŃĚåŘńäžĘåĘĚéČÍçŽĎæŰĞæIJñïijŇèĂŇ
`get()` æŰźæşŢèČ¡èŐůåŔŰåśđæĂğåĂijãĂĆä¿ŇåęĆïijŽ

```
>>> doc
<xml.etree.ElementTree.ElementTree object at 0x101339510>
>>> e = doc.find('channel/title')
>>> e
<Element 'title' at 0x10135b310>
>>> e.tag
'title'
>>> e.text
'Planet Python'
>>> e.get('some_attribute')
>>>
```

æIJL'äÿĂçĆźèęĄåijžèřČçŽĎæŸŕ `xml.etree.ElementTree` åźűäÿ■æŸŕXM-
LèğčæđŘçŽĎåŤŕäÿĂæŰźæşŢãĂĆ åŕźäžŐæŻt'énŸçžğçŽĎåžŤçŤÍçÍŃåžŔïijŇä¡äéIJĂèęĄèĂČèŹŚä¡£çŤÍ
`lxml` ãĂĆ åőČä¡£çŤÍäžĘåŠŇElementTreeåŘŇæăůçŽĎçijŰçÍŃæŐěåŔčïijŇåŻăæ■d'äÿŁéÍćçŽĎä¿Ńå■ŘåŘŇa
ä¡ääŔłéIJĂèęĄářĘåĹŽåijĂåğŃçŽĎimportèŕ■åŘěæ■ćæĹŘ `from lxml.etree import parse` åřśèąŇäžĘãĂĆ `lxml` åőŇåĚÍéĄţå¿ŁXMLæăĞåĞĘïijŇåźűäÿŤéĂ§åžęäź§éİđåÿÿå£ńïijŇåŘŇæŮűè£Ÿa

6.4 åćđéĞŔåijŔèğčæđŘåd'ğåđŃXMLæŰĞäžű

éŮőécŸ

ä¡ääČşä¡£çŤÍår¡åŔŕèČ¡ářŞçŽĎåĘĚå■ŸäžŐäÿĂäÿłèűĚåd'ğçŽĎXMLæŰĞæąčäÿ■æŔŘåŔŰæŢřæ■őãĂĆ

èğčåĘşæŰźæąĹ

äžżä¡ŢæŰűåĂŹåŔłèęĄä¡ăéĄĞåĹřåćđéĞŔåijŔçŽĎæŢřæ■őåd'ĎçŘĘæŮűïijŇçňňäÿĂæŮűéŮt'ářśåžŤèŕěa
äÿŃéÍćæŸŕäÿĂäÿłå¿ĹçőĂå■ŢçŽĎåĞ¡æŢřïijŇåŔłä¡£çŤÍå¿ĹářŞçŽĎåĘĚå■ŸářśèČ¡åćđéĞŔåijŔçŽĎåd'ĎçŘĘä

```
from xml.etree.ElementTree import iterparse

def parse_and_remove(filename, path):
    path_parts = path.split('/')
    doc = iterparse(filename, ('start', 'end'))
    # Skip the root element
    next(doc)

    tag_stack = []
    elem_stack = []
    for event, elem in doc:
        if event == 'start':
            tag_stack.append(elem.tag)
            elem_stack.append(elem)
        elif event == 'end':
            if tag_stack == path_parts:
                yield elem
                elem_stack[-2].remove(elem)
            try:
                tag_stack.pop()
                elem_stack.pop()
            except IndexError:
                pass
```

äÿžäžĘæţŃèŕŢè£ŹäÿłåĞ¡æŢřïijŇä¡äéIJĂèęĄåĚĹæIJL’äÿĂäÿłåď ğåđŃçŽĎXMLæŰĞäżűãĂĆ éĂŽåÿÿä¡ăåŘŕäžěåIJĺæŤ£åžIJç¡ŚçńŹæĹŰåĚňåĚśæŢřæ■ős¡ŚçńŹäÿŁæL’¿åĹřè£Źæăůç ŽĎæŰĞäżűãĂĆ ä¿ŇåęĆïijŇä¡ăåŘŕäžěäÿŃè¡¡XMLæăijåijŘçŽĎèŁİåŁăåŞěå§ŐåÿĆéĄŞèůŕåİŚæť ijæŢřæ■őåžŞãĂĆ åIJĺåĘŹè£ŹæIJňäźęçŽĎæŮűåĂŹïijŇäÿŃè¡¡æŰĞäżűåůšçżŘåŇĚåŘńèűĚè£Ğ100,000èąŇæŢřæ■őïijŇçijŰçăĄ

```
<response>
    <row>
        <row ...>
            <creation_date>2012-11-18T00:00:00</creation_date>
            <status>Completed</status>
            <completion_date>2012-11-18T00:00:00</completion_date>
            <service_request_number>12-01906549</service_request_
↪number>
            <type_of_service_request>Pot Hole in Street</type_of_
↪service_request>
            <current_activity>Final Outcome</current_activity>
            <most_recent_action>CDOT Street Cut ... Outcome</most_
↪recent_action>
            <street_address>4714 S TALMAN AVE</street_address>
            <zip>60632</zip>
            <x_coordinate>1159494.68618856</x_coordinate>
            <y_coordinate>1873313.83503384</y_coordinate>
            <ward>14</ward>
            <police_district>9</police_district>
            <community_area>58</community_area>
            <latitude>41.808090232127896</latitude>
```

```
            <longitude>-87.69053684711305</longitude>
            <location latitude="41.808090232127896"
            longitude="-87.69053684711305" />
        </row>
        <row ...>
            <creation_date>2012-11-18T00:00:00</creation_date>
            <status>Completed</status>
            <completion_date>2012-11-18T00:00:00</completion_date>
            <service_request_number>12-01906695</service_request_
↪number>
            <type_of_service_request>Pot Hole in Street</type_of_
↪service_request>
            <current_activity>Final Outcome</current_activity>
            <most_recent_action>CDOT Street Cut ... Outcome</most_
↪recent_action>
            <street_address>3510 W NORTH AVE</street_address>
            <zip>60647</zip>
            <x_coordinate>1152732.14127696</x_coordinate>
            <y_coordinate>1910409.38979075</y_coordinate>
            <ward>26</ward>
            <police_district>14</police_district>
            <community_area>23</community_area>
            <latitude>41.91002084292946</latitude>
            <longitude>-87.71435952353961</longitude>
            <location latitude="41.91002084292946"
            longitude="-87.71435952353961" />
        </row>
    </row>
</response>
```

åĄĞèő¿ä¡ăæČşåĘŹäÿĂäÿłèĎŽæIJňæİěæŇL’çĔğåİŚæť’ijæŁěåŚŁæŢřéĞŔæŐŠåĹŮéĆőçijŰůŔůçăĄãĂĆä

```
from xml.etree.ElementTree import parse
from collections import Counter

potholes_by_zip = Counter()

doc = parse('potholes.xml')
for pothole in doc.iterfind('row/row'):
    potholes_by_zip[pothole.findtext('zip')] += 1
for zipcode, num in potholes_by_zip.most_common():
    print(zipcode, num)
```

è£ŹäÿłèĎŽæIJňåŤŕäÿĂçŽĎéŮőécŸæŸŕåőČäijŽåĚĹåřĘæŢt’äÿłXMLæŰĞäžűåŁăè¡¡åĹŕåĘĚå■Ÿäÿ■çĎű
åIJĺæĹŚçŽĎæIJžåŹĺäÿŁïijŇäÿžäžĘè£ŘèąŇè£ŹäÿłçĺŃåžŔéIJĂèęĄçŤĺåĹř450MBåůęåŔşçŽĎåĘĚå■Ÿçl’žéŮt’ă
åęĆæđIJä¡£çŤĺåęĆäÿŃäžčçăĄïijŇçĺŃåžŔåŔłéIJĂèęĄä£őæŤźäÿĂçĆźçĆźïijŽ

```
from collections import Counter

potholes_by_zip = Counter()
```

```
data = parse_and_remove('potholes.xml', 'row/row')
for pothole in data:
    potholes_by_zip[pothole.findtext('zip')] += 1
for zipcode, num in potholes_by_zip.most_common():
    print(zipcode, num)
```

çžŞædIJæŸŕïijŽè£ŹäÿłçL'ĹæIJňçŽĎäžčçăĄè£ŘèąŇæŮűåŔłéIJĂèęĄ7MBçŽĎåĘĚå■Ÿ–åď'ğåď'ğèŁĆçžęäžĘåĘĚå■ŸètĎæžŘãĂĆ

èőÍèőž

è£ŹäÿĂèŁĆçŽĎæŁĂæIJŕäijŽä¿ÌètŰ `ElementTree` æĺąåİŮäÿ■çŽĎäÿď'äÿłæăÿå£ČåŁ§èČ¡ãĂĆ çňňäÿĂïijŇiterparse() æŰźæşŢåĚĄèőÿåŕźXMLæŰĞæąčè£ŻèąŇåćđéĞŔæŞ■ä¡IJãĂĆ ä¡£çŤĺæŮűïijŇä¡ăéIJĂèęĄæŔŘä¿ŻæŰĞäžűåŘ■åŠŇäÿĂäÿłåŇĚåŘńäÿŃéİćäÿĂçğ■æĹŰåď'Žçğ■çśžåđŃçŽĎä `start`, `end`, `start-ns` åŠŇ `end-ns` ãĂĆ çŤś `iterparse()` åĹŻåžžçŽĎè£■äžčåŹĺäijŽäžğçŤ§å¡ćåęĆ `(event, elem)` çŽĎåĚČçżĎïijŇ åĚűäÿ■ `event` æŸŕäÿŁè£ŕäžŃäžűåĹŮèąĺäÿ■çŽĎæ§ŘäÿĂäÿłïijŇèĂŇ `elem` æŸŕçŻÿåžŤçŽĎXMLåĚČçť'ããĂĆä¿ŃåęĆïijŽ

```
>>> data = iterparse('potholes.xml',('start','end'))
>>> next(data)
('start', <Element 'response' at 0x100771d60>)
>>> next(data)
('start', <Element 'row' at 0x100771e68>)
>>> next(data)
('start', <Element 'row' at 0x100771fc8>)
>>> next(data)
('start', <Element 'creation_date' at 0x100771f18>)
>>> next(data)
('end', <Element 'creation_date' at 0x100771f18>)
>>> next(data)
('start', <Element 'status' at 0x1006a7f18>)
>>> next(data)
('end', <Element 'status' at 0x1006a7f18>)
>>>
```

`start` äžŃäžűåIJÍæ§ŘäÿłåĚČçť'ăçňňäÿĂæňąèćńåĹŻåžžåźűäÿŤè£ŸæšąæIJL'èćńæŔŠåĚèåĚűäžŰæŢřæ■èĂŇ `end` äžŃäžűåIJÍæ§ŘäÿłåĚČçť'ăåůšçżŔåőŇæĹŘæŮűèćńåĹŻåžžãĂĆ åř¡çőąæšąæIJL'åIJÍä¿Ńå■Řäÿ■æijŤçď'žïijŇ `start-ns` åŠŇ `end-ns` äžŃäžűèćńçŤÍæİěåď'ĎçŘĘXMLæŰĞæąčåŚ¡åŘ■çl'žéŮť'çŽĎåčřæŸŐãĂĆ

è£ŹæIJňèŁĆä¿Ńå■Řäÿ■ïijŇ `start` åŠŇ `end` äžŃäžűèćńçŤÍæİěçőąçŘĘåĚČçť'ăåŠŇæăĞç■¿æăĹãĂĆ æăĹäžčèąÍäžĘæŰĞæąčèćńèğčæđŘæŰűçŽĎåśĆæňąçżŞæđĎïijŇ è£ŸèćńçŤÍæİěåĹď'æŰ■æ§ŘäÿłåĚČçť'ăæŸŕåŘęåŇźéĚ■äijäçżŹåĞ¡æŢř `parse_and_remove()` çŽĎèůŕå¿ĎãĂĆ åęĆæđIJåŇźéĚ■ïijŇářśåĹl'çŤÍ `yield` èŕ■åŘěåŘŚèřČçŤÍèĂĚè£ŤåŻđè£ŹäÿłåĚČçť'ããĂĆ

åIJÍ `yield` äźŃåŘŐçŽĎäÿŃéİćè£Źäÿłèŕ■åŘěæL'■æŸŕä¡£å¿ŮçÍŃåžŔå■ăçŤÍæđĄåřŚåĘĚå■ŸçŽĎElemer

```
elem_stack[-2].remove(elem)
```

è£Źäÿłèŕ■åŔëä¡£å¿ÛäźŃåL’■çŤś yield äžğçŤ§çŽĎåĚČçt’ääżŐåőČçŽĎçĹűèŁĆçĆźäÿ■åĹäéŹd’æŐL’ãA
åĄĞèő¿åůšçżŔæšąæIJL’åĚűåőČçŽĎåIJřæŰźåijŢçŤĺè£ŹäÿłåĚČçt’ääžĘïijŇéĆčäźLè£ŹäÿłåĚČçt’ääřšèćńéŤÄæ

åŕźèŁĆçĆźçŽĎè£■äžčåijŔèğčædŘåŠŇåĹäéŹd’çŽĎæIJÄçżĹæŢĹædIJåřśæŸŕäÿÄäÿłåIJĺæŰĞæąčäÿŁéńŸ
æŰĞæąčæăŚçżŞædĎäżŐåğŃèĞłçżĹæšąèćńåőŇæŢt’çŽĎåĹŽåžžè£ĞãĂĆår¡çőąåęĆæ■d’ïijŇè£ŸæŸŕèČ¡éĂŽă

è£Źçğ■æŰźæąĹçŽĎäÿżèęĄçijžéŹůåřśæŸŕåőČçŽĎè£ŘèąŇæĂğèČ¡äžĘãĂĆ
æĹŚèĞłåůśæţŃèŕŢçŽĎçżŞædIJæŸŕïijŇèŕżåŔŰæŢt’äÿłæŰĞæąčåĹřåĘĚå■ŸäÿçŽĎçL’ĹæIJñçŽĎè£ŘèąŇéĂ§
ä¡ĘæŸŕåőČå■t’ä¡£çŤĺäžĘèúĚè£ĞåŘŐèĂĚ60åĂ■çŽĎåĘĚå■ŸãĂĆ
åŻăæ■d’ïijŇåęĆædIJä¡ăæŻt’åĚşå£ČåĘĚå■Ÿä¡£çŤĺéĞŔçŽĎèŕÏïijŇéĆčäźĹáćđéĞŔåijŔçŽĎçL’ĹæIJñåőŇèČIJă

6.5 åřĘå■ŮåĚÿè¡ňæ■ćäÿžXML

éŮőécŸ

ä¡ăæČşä¡£çŤĺäÿĂäÿłPythonå■ŮåĚÿå■ŸåĆĺæŢřæ■őïijŇáźűåřĘåőČè¡ňæ■ćæĹŘXMLæăijåijŔãĂĆ

èğčåĘşæŰźæąĹ

åř¡çőą xml.etree.ElementTree åžŞéĂŽåÿÿçŤĺæİěåĄŽèğčædŘåůěä¡IJïijŇåĚűåődåőČäź§åŔŕäżěåĹ
ä¿ŇåęĆïijŇèĂČèŹŠåęĆäÿŃè£ŹäÿłåĞ¡æŢřïijŽ

```
from xml.etree.ElementTree import Element

def dict_to_xml(tag, d):
'''
Turn a simple dict of key/value pairs into XML
'''
elem = Element(tag)
for key, val in d.items():
    child = Element(key)
    child.text = str(val)
    elem.append(child)
return elem
```

äÿŃéİćæŸŕäÿÄäÿłä¡£çŤĺä¿Ńå■ŘïijŽ

```
>>> s = { 'name': 'GOOG', 'shares': 100, 'price':490.1 }
>>> e = dict_to_xml('stock', s)
>>> e
<Element 'stock' at 0x1004b64c8>
>>>
```

è¡ňæ■ćçżŞædIJæŸŕäÿÄäÿł Element åődä¿ŃãĂĆårźäžŐI/OæŞ■ä¡IJïijŇä¡£çŤĺ xml.
etree.ElementTree äÿ■çŽĎ tostring() åĞ¡æŢřå¿ĹåőźæŸŞåřśèČ¡ářĘåőČè¡ňæ■ćæĹŘäÿÄäÿłå■ŮèŁ

```
>>> from xml.etree.ElementTree import tostring
>>> tostring(e)
b'<stock><price>490.1</price><shares>100</shares><name>GOOG</name></
 ↪stock>'
>>>
```

åęĆædIJä¡ăæČşçżŹæ§ŘäÿłåĚČçť'ăæůżåŁăåśđæĂğåĂijïijŇåŔŕäżěä¡£çŤÍ `set()` æŰźæşŢïijŽ

```
>>> e.set('_id','1234')
>>> tostring(e)
b'<stock _id="1234"><price>490.1</price><shares>100</shares><name>
 ↪GOOG</name>
</stock>'
>>>
```

åęĆædIJä¡ăè£ŸæČşä£ÌæŇĄåĚČçť'ăçŽĎéąžåžŔïijŇåŔŕäżěèĂČèŹŚædĎéĂăäÿĂäÿł `OrderedDict` æÌěäžčæŻ£äÿĂäÿłæŹőéĂŽçŽĎå■ŮåĚÿãĂĆèŕůåŔĆèĂČ1.7åřŔèŁĆãĂĆ

èőÍèőž

å¡ŞåĹŻåżžXMLçŽĎæŮűåĂŹïijŇä¡ăèćńéŹŘåĹűåŔłèČ¡ædĎéĂăå■ŮçňęäÿšçśżåđŃçŽĎåĂijãĂĆä¿ŃåęĆï

```
def dict_to_xml_str(tag, d):
    '''
    Turn a simple dict of key/value pairs into XML
    '''
    parts = ['<{}>'.format(tag)]
    for key, val in d.items():
        parts.append('<{0}>{1}</{0}>'.format(key,val))
    parts.append('</{}>'.format(tag))
    return ''.join(parts)
```

éŮőécŸæŸŕåęĆædIJä¡ăæL'ŃåŁÍçŽĎåŐżædĎéĂăçŽĎæŮűåĂŹåŔŕèČ¡äijŽçćřåĹřäÿĂäžŽéžżçČęãĂĆä¿Ń

```
>>> d = { 'name' : '<spam>' }

>>> # String creation
>>> dict_to_xml_str('item',d)
'<item><name><spam></name></item>'

>>> # Proper XML creation
>>> e = dict_to_xml('item',d)
>>> tostring(e)
b'<item><name>&lt;spam&gt;</name></item>'
>>>
```

æşÍæĎŔåĹřçÍŃåžŔçŽĎåŘŐéÍćéĆčäÿłä¿Ńå■Řäÿ■ïijŇå■Ůçňę '<' åŠŇ '>' èćńæŻ£æ■ćæĹŘäžĘ `<` åŠŇ `>`

äÿŃéÌćäżĚä¿ŻåŔĆèĂČïijŇåęĆæđIJä¡ăéIJĂèęĄæĽŃåŁÍåŐżè¡ňæ■ćè£ŹäžŻå■Ůçňę¨ijŇåŔŕäžěä¡£çŤÍ `xml.sax.saxutils` äÿ■çŽĎ `escape()` åŠŇ `unescape()` åĞ¡æŢřãĂĆä¿ŃåęĆïijŽ

```
>>> from xml.sax.saxutils import escape, unescape
>>> escape('<spam>')
'&lt;spam&gt;'
>>> unescape(_)
'<spam>'
>>>
```

éŹđ'äžĘèČ¡åĹŻåżžæ■čçąőçŽĎè¿ŞåĞžåđ'Ű¨ijŇè£ŸæIJL'åŔęåđ'ŰäÿĂäÿłåŐ§åŻäæŐÍè■Řä¡ääĹŻåżž `Element` åőđä¿ŃèĂŇäÿ■æŸŕå■Ůçňę¨äÿšïijŇ éĆčářśæŸŕä¡£çŤÍå■Ůçňę¨äÿšçżĎåŘĹæđĎéĂääÿĂäÿłæŻť'åđ'ğç èĂŇ `Element` åőđä¿ŃåŔŕäžěä¨ä■çŤÍèĂČèŹŚèğčæđŘXMLæŰĞæIJńçŽĎæČĚåĘţäÿŃéĂŽè£Ğåđ'Žçğ■æŰźäi äź§ářśæŸŕèŕť'ïijŇä¡ääŔŕäžěåIJÍäÿĂäÿłéńŸçžğæŢřæ■őçżŞæđĎäÿŁåőŇæĹŘä¡ăæĽ'ĂæIJL'çŽĎæŞ■ä¡IJïijŇåźű

6.6 èğčæđŘåŠŇä£őæŤźXML

éŮőécŸ

ä¡ăæČşèŕžåŔŰäÿĂäÿłXMLæŰĞæąčïijŇåŕźåőČæIJĂäÿĂäžŻä£őæŤźïijŇçĎűåŘŐårĘçżŞæđIJåĘŹåŻđXM

èğčåĘşæŰźæąĹ

ä¡£çŤÍ `xml.etree.ElementTree` æÍąåİŮåŔŕäžěå¿ĹåőźæŸŞçŽĎåđ'ĎçŘĘè£ŹäžŻäżżåŁąãĂĆ çňňäÿĂæ■ěæŸŕäžěéĂŽåÿÿçŽĎæŰźåijŔæİèèğčæđŘè£ŹäÿłæŰĞæąčãĂĆä¿ŃåęĆïijŇåĄĞèő¿ä¡ăæIJL'äÿĂäÿłå `pred.xml` çŽĎæŰĞæąčïijŇçśżäijijäÿŃéÌćè£ŹæăůïijŽ

```
<?xml version="1.0"?>
<stop>
    <id>14791</id>
    <nm>Clark &amp; Balmoral</nm>
    <sri>
        <rt>22</rt>
        <d>North Bound</d>
        <dd>North Bound</dd>
    </sri>
    <cr>22</cr>
    <pre>
        <pt>5 MIN</pt>
        <fd>Howard</fd>
        <v>1378</v>
        <rn>22</rn>
    </pre>
    <pre>
        <pt>15 MIN</pt>
        <fd>Howard</fd>
        <v>1867</v>
```

```
        <rn>22</rn>
    </pre>
</stop>
```

äÿŃéÌćæŸŕäÿĂäÿłåĹľçŤÍ ElementTree æÌěèŕžåŔŰè£ŹäÿłæŰĞæąčåźűåŕžåőČåĄŽäÿĂäžŽä£őæŤźçŽ

```
>>> from xml.etree.ElementTree import parse, Element
>>> doc = parse('pred.xml')
>>> root = doc.getroot()
>>> root
<Element 'stop' at 0x100770cb0>

>>> # Remove a few elements
>>> root.remove(root.find('sri'))
>>> root.remove(root.find('cr'))
>>> # Insert a new element after <nm>...</nm>
>>> root.getchildren().index(root.find('nm'))
1
>>> e = Element('spam')
>>> e.text = 'This is a test'
>>> root.insert(2, e)

>>> # Write back to a file
>>> doc.write('newpred.xml', xml_declaration=True)
>>>
```

åď ĎçŘEçżŞæđIJæŸŕäÿĂäÿłåČŔäÿŃéÌćè£ŹæăůæŰřçŽĎXMLæŰĞäžűïijŽ

```
<?xml version='1.0' encoding='us-ascii'?>
<stop>
    <id>14791</id>
    <nm>Clark &amp;amp; Balmoral</nm>
    <spam>This is a test</spam>
    <pre>
        <pt>5 MIN</pt>
        <fd>Howard</fd>
        <v>1378</v>
        <rn>22</rn>
    </pre>
    <pre>
        <pt>15 MIN</pt>
        <fd>Howard</fd>
        <v>1867</v>
        <rn>22</rn>
    </pre>
</stop>
```

èőÍèőž

ä£őæŤźäÿĂäÿłXMLæŰĞæąčçżŞædĎæŸŕå¿Ĺåőźæŷ§çŽĎïijŇä¡ĘæŸŕä¡ăå£ĚéążçĽćèőřçŽĎæŸŕæĽĂæĺ
åřĘåőČä¡IJäÿžäÿĂäÿłåĹŮèąÍæİěåď ĎçŘĘãĂĆä¿ŃåęĆïijŇåęĆædIJä¡ăåĹăéŹď æ§Räÿłåĕčçť ăïijŇéĂŽè£ĞèřČ
`remove()` æŰźæşŢäżŐåőČçŽĎçŻť æŐěçĹűèŁĆçĆźäÿ■åĹăéŹď ãĂĆ
åęĆædIJä¡ăæŔŠåĚěæĹŰåćđåŁăæŰřçŽĎåĚČçť ăïijŇä¡ăåŘŇæăůä¡£çŤĺçĹűèŁĆçĆźåĚČçť ăçŽĎ
`insert()` åŠŇ `append()` æŰźæşŢãĂĆ è£ŸèČ¡åŕźåĚČçť ää¡£çŤĺçť ćåijŢåŠŇåĹĞçĽĞæŞ■ä¡IJïijŇæŕŤåęĆ
`element[i]` æĹŰ `element[i:j]`

åęĆædIJä¡ăéIJĂèęĄåĹŻåżžæŰřçŽĎåĚČçť ăïijŇåŔŕäžěä¡£çŤĺæIJňèŁĆæŰźæąĹäÿ■æijŤçď žçŽĎ
`Element` çśżãĂĆæĹŠäžňåIJĺ6.5åřŔèŁĆåůšçżŔèŕęçżĘèőĺèőžè£ĞäžĘãĂĆ

6.7 åĹľçŤĺåŚ¡åŘ■çľžéŮťèğčædŘXMLæŰĞæąč

éŮőécŸ

ä¡ăæČşèğčædŘæ§ŘäÿłXMLæŰĞæąčïijŇæŰĞæąčäÿ■ä¡£çŤĺäžĘXMLåŚ¡åŘ■çľžéŮťãĂĆ

èğčåĘşæŰźæąĹ

èĂČèŹŠäÿŃéİćè£Źäÿłä¡£çŤĺäžĘåŚ¡åŘ■çľžéŮť çŽĎæŰĞæąčïijŽ

```
<?xml version="1.0" encoding="utf-8"?>
<top>
    <author>David Beazley</author>
    <content>
        <html xmlns="http://www.w3.org/1999/xhtml">
            <head>
                <title>Hello World</title>
            </head>
            <body>
                <h1>Hello World!</h1>
            </body>
        </html>
    </content>
</top>
```

åęĆædIJä¡ăèğčædŘè£ŹäÿłæŰĞæąčåźűæĽğèąŇæŹőéĂŽçŽĎæ§ěèŕćïijŇä¡ăäijŽåŔŚçŐřè£Źäÿłåźűäÿ■æŸ

```
>>> # Some queries that work
>>> doc.findtext('author')
'David Beazley'
>>> doc.find('content')
<Element 'content' at 0x100776ec0>
>>> # A query involving a namespace (doesn't work)
>>> doc.find('content/html')
>>> # Works if fully qualified
>>> doc.find('content/{http://www.w3.org/1999/xhtml}html')
```

```
<Element '{http://www.w3.org/1999/xhtml}html' at 0x1007767e0>
>>> # Doesn't work
>>> doc.findtext('content/{http://www.w3.org/1999/xhtml}html/head/
↪title')
>>> # Fully qualified
>>> doc.findtext('content/{http://www.w3.org/1999/xhtml}html/'
... '{http://www.w3.org/1999/xhtml}head/{http://www.w3.org/1999/
↪xhtml}title')
'Hello World'
>>>
```

ä¡ăåŔŕäżěéĂŽè£ĞåřĘåŚ¡åŘ■çľ žéŮť åď ĎçŘĘéĂżè¿ŚåŇĚèčĚäÿžäÿĂäÿłåůěåĚůçśżæİěçőĂåŃŰè£Źäÿłè£

```python
class XMLNamespaces:
    def __init__(self, **kwargs):
        self.namespaces = {}
        for name, uri in kwargs.items():
            self.register(name, uri)
    def register(self, name, uri):
        self.namespaces[name] = '{'+uri+'}'
    def __call__(self, path):
        return path.format_map(self.namespaces)
```

éĂŽè£ĞäÿŃéİćçŽĎæŰźåijŔä¡£çŤĺè£Źäÿłçśżïij Ž

```
>>> ns = XMLNamespaces(html='http://www.w3.org/1999/xhtml')
>>> doc.find(ns('content/{html}html'))
<Element '{http://www.w3.org/1999/xhtml}html' at 0x1007767e0>
>>> doc.findtext(ns('content/{html}html/{html}head/{html}title'))
'Hello World'
>>>
```

èőĺèőž

èğčæđŘåŘńæIJL'åŚ¡åŘ■çľ žéŮť çŽĎXMLæŰĞæąčäijŽæŕŤè¿ČçźĄçŘŘãĂĆ äÿŁéİćçŽĎ `XMLNamespaces` äżĚäżĚæŸŕåĚĄèőÿä¡ăä¡£çŤĺçijľ çŢěåŘ■äżčæŻ£åőŇæŢť çŽĎURIåřĘåĚűåŔŸå¿Ůçĺ■å¿őç

å¿Ĺäÿ■åźÿçŽĎæŸŕïijŇåIJĺå§žæIJňçŽĎ `ElementTree` èğčæđŘäÿ■æšąæIJL'äżżä¡ŢéĂŤå¿ĎèŐůåŔŰåŚ¡åŘ■çľ žéŮť çŽĎä£ąæĄŕãĂĆ ä¡ĘæŸŕïijŇåęĆæđIJä¡ăä¡£çŤĺ `iterparse()` åĞ¡æŢřçŽĎèŕİåřśåŔŕäżěèŐůåŔŰæŻť åď ŽåĚşäžŐåŚ¡åŘ■çľ žé

```
>>> from xml.etree.ElementTree import iterparse
>>> for evt, elem in iterparse('ns2.xml', ('end', 'start-ns', 'end-
↪ns')):
... print(evt, elem)
...
end <Element 'author' at 0x10110de10>
start-ns ('', 'http://www.w3.org/1999/xhtml')
end <Element '{http://www.w3.org/1999/xhtml}title' at 0x1011131b0>
```

```
end <Element '{http://www.w3.org/1999/xhtml}head' at 0x1011130a8>
end <Element '{http://www.w3.org/1999/xhtml}h1' at 0x101113310>
end <Element '{http://www.w3.org/1999/xhtml}body' at 0x101113260>
end <Element '{http://www.w3.org/1999/xhtml}html' at 0x10110df70>
end-ns None
end <Element 'content' at 0x10110de68>
end <Element 'top' at 0x10110dd60>
>>> elem # This is the topmost element
<Element 'top' at 0x10110dd60>
>>>
```

æIJĂåŘŐäÿĂçĆźïijŇåęĆæđIJä¡ăèęĄåď ĎçŘEçŽĎXMLæŰĞæIJňéŹď äžEèęĄä¡£çŤĺåĹřåĔűäżŰéńŸçžğ
åžžèőőä¡ăæIJĂåě¡æŸŕä¡£çŤÍ lxml åĞ¡æŢřåžŞæÍěäžčæŻ£ ElementTree ãĂĆ
ä¿ŃåęĆïijŇlxml åŕźåĹľçŤÍDTDélŇèŕĄæŰĞæąčãĂĄæŻť åě¡çŽĎXPathæŤŕæŇĄåŠŇäÿĂäžŻåĔűäżŰéńŸçžğ
è£ŹäÿĂåřŔèŁĆåĔűåőđåŔłæŸŕæŢŹä¡ääęĆä¡Ţèőľ XMLèğčæđŘçÍ■å¿őçőĂå■ŢäÿĂçĆźãĂĆ

6.8 äÿŐåĚşçşžåđŃæŢřæ■őåžŞçŽĎäžď äžŠ

éŮőécŸ

ä¡ăæČşåIJÍåĚşçşžåđŃæŢřæ■őåžŞäÿ■æ§ěèŕćãĂĄåćđåŁăæĹŰåĹáéŹď èőřå¡ŢãĂĆ

èğčåĘşæŰźæąĹ

Pythonäÿ■èąÍçď žåď ŽèąŇæŢřæ■őçŽĎæăĞåĞĘæŰźåijŔæŸŕäÿĂäÿłçŤśåĚČçżĎæđĎæĹŘçŽĎåžŔåĹŮãĂ

```
stocks = [
    ('GOOG', 100, 490.1),
    ('AAPL', 50, 545.75),
    ('FB', 150, 7.45),
    ('HPQ', 75, 33.2),
]
```

ä¿Íæ■őPEP249ïijŇéĂŽè£Ğè£Źçğ■å¡ćåijŔæŔŘä¿ŻæŢřæ■őïijŇ
åŘŕäžěå¿ĹåőźæŸ§çŽĎä¡£çŤÍPythonæăĞåĞĘæŢřæ■őåžŞAPIåŠŇåĚşçşžåđŃæŢřæ■őåžŞè£ŻèąŇäžď äžŠãĂĆ
æĽĂæIJĽæŢřæ■őåžŞäÿŁçŽĎæŞ■ä¡IJéČ¡éĂŽè£ĞSQLæ§ěèŕćèŕ■åŘěæÍěåőŇæĹŘãĂĆæŕŔäÿĂèąŇè¿ŞåĚěè¿Ş

äÿžäžĘæijŤçď žèŕťæŸŐïijŇä¡ääŘŕäžěä¡£çŤÍPythonæăĞåĞĘåžŞäÿ■çŽĎ sqlite3
æĺąåİŮãĂĆ åęĆæđIJä¡ää¡£çŤÍçŽĎæŸŕäÿĂäÿłäÿ■åŘŇçŽĎæŢřæ■őåžŞ(æŕŤåęĆMySqlãĂĄPostgresqlæĹŰèĂ
è£Ÿå¿ŮåőĽèčĚçŻÿåžŤçŽĎçňňäÿĽæŰźæĺąåİŮæÍěæŘŔä¿ŻæŤŕæŇĄãĂĆ
äÿ■è£ĞçŻÿåžŤçŽĎçijŰçÍŃæŐěåŘčåĞääźŐéČ¡æŸŕäÿĂæăůçŽĎïijŇéŹď äžĘäÿĂçĆźçĆźçŻĘå¿őåůőåĹńåď Ůã

çňňäÿĂæ■ěæŸŕè£đæŐěåĹřæŢřæ■őåžŞãĂĆéĂŽåÿÿä¡ăèęĄæĽğèąŇ connect()
åĞ¡æŢřïijŇ çżŹåőČæŔŘä¿ŻäÿĂäžŻæŢřæ■őåžŞåŘ■ãĂĄäÿżæIJžãĂĄçŤÍæĹůåŘ■ãĂĄåŕĘçăĄåŠŇåĚűäżŰå£Ě

```
>>> import sqlite3
>>> db = sqlite3.connect('database.db')
>>>
```

äÿžäžĘåď ĎçŘĘæŢřæ■őïijŇäÿŃäÿĂæ■ěä¡ăéIJĂèęĄåĹŻåžžäÿĂäÿłæÿÿæăĞãĂĆ
äÿĂæŮęä¡ăæIJL’äžĘæÿÿæăĞïijŇéĆčäźĹä¡ăářšåŘŕäžěæL’ğèąŇSQLæ§ěèŕćèŕ■åŘěäžĘãĂĆærŤåęĆïijŽ

```
>>> c = db.cursor()
>>> c.execute('create table portfolio (symbol text, shares integer,␣
↪price real)')
<sqlite3.Cursor object at 0x10067a730>
>>> db.commit()
>>>
```

äÿžäžĘåŘŚæŢřæ■őåžŞèąĺäÿ■æŔŠåĚěåď Žæİąèőŕå¡ŢïijŇä¡£çŤĺçśżäijijäÿŃéİćè£ŹæăůçŽĎèŕ■åŘěïijŽ

```
>>> c.executemany('insert into portfolio values (?,?,?)', stocks)
<sqlite3.Cursor object at 0x10067a730>
>>> db.commit()
>>>
```

äÿžäžĘæL’ğèąŇæ§Řäÿłæ§ěèŕćïijŇä¡£çŤĺåČŔäÿŃéİćè£ŹæăůçŽĎèŕ■åŘěïijŽ

```
>>> for row in db.execute('select * from portfolio'):
...     print(row)
...
('GOOG', 100, 490.1)
('AAPL', 50, 545.75)
('FB', 150, 7.45)
('HPQ', 75, 33.2)
>>>
```

åęĆædIJä¡ăæČşæŐěåŔŮçŤĺæĹůè¿ŞåĚěä¡IJäÿžåŔĆæŢřæİěæL’ğèąŇæ§ěèŕćæŞ■ä¡IJïijŇå£Ěéążçąőä£İä¡ăă

```
>>> min_price = 100
>>> for row in db.execute('select * from portfolio where price >= ?
↪',
                          (min_price,)):
...     print(row)
...
('GOOG', 100, 490.1)
('AAPL', 50, 545.75)
>>>
```

èőÍèőž

åIJÍæŕŤè¿Čä¡ŐçŽĎçžğåĹńäÿŁåŠŇæŢřæ■őåžŞäžď äžŠæŸréİđåÿÿçőĂå■ŢçŽĎãĂĆ
ä¡ăåŘłéIJĂæŘŘä¿ŻSQLèŕ■åŘěåźűěřĆçŤĺçŻÿåžŤçŽĎæĺąåİŮåřšåŘŕäžěæŻťæŰřæĹŰæŘŘåŘŰæŢřæ■őäžĘãĂ
èŹ¡èŕťåęĆæ■ď ïijŇè£ŸæŸŕæIJL’äÿĂäžŻæŕŤè¿ČæčŸæL’ŃçŽĎçżĘèŁĆéŮőécŸéIJĂèęĄä¡ăéĂŘäÿłåĹŮåĞžåĆ

äÿĂäÿłéŽ¿çĆźæŸŕæŢřæ■őåžŞäÿ■çŽĎæŢřæ■őåŠŇPythonçśżåđŃçŻťæŐěçŽĎæŸäářĎãĂĆ
åŕźäžŐæŮěæIJ§çśżåđŃïijŇéĂŽåÿÿåŘŕäžěä¡£çŤĺ `datetime` æĺąåİŮäÿ■çŽĎ `datetime`
åőđä¿ŃïijŇ æĹŰèĂĚåŘŕèČ¡æŸŕ `time` æĺąåİŮäÿ■çŽĎçşżçż§æŮűéŮťæĹşãĂĆ
åŕźäžŐæŢřå■ŮçśżåđŃïijŇçL’źåĹńæŸŕä¡£çŤĺåĽřářŘæŢřçŽĎéĞŚèđ■æŢřæ■őïijŇåŘŕäžěçŤĺ

decimal æÍąåİŮäÿ■çŽĎ Decimal åőđä¿ŃæİěèąÍçď'žãĂĆ
äÿ■åźÿçŽĎæŸŕïijŇåŕźäžŐäÿ■åŘŇçŽĎæŢřæ■őåžŞèĂŇèÍĂåĚůä¡ŞæŸăářĐèğĎåĹŹæŸŕäÿ■äÿĂæăůçŽĎïijŇä¡

åŔęåď'ŰäÿĂäÿłæŻt'åŁăåď'■æİĆçŽĎéŰőécŸåřśæŸŕSQLèŕ■åŔěå■Ůçňęäÿšç ŽĎæđĎéĂããĂĆ
ä¡ăå■ČäÿĞäÿ■èęĄä¡£çŤÍPythonå■ŮçňęäÿšæăijåijŔåŇŰæŞ■ä¡IJçňę(åęĆ%)æĹŰèĂĚ
.format() æŰźæşŢæİěåĹŻåžžè£ŹæăůçŽĎå■Ůçňęäÿšã ĂĆ
åęĆæđIJäijăéĂŠçżŹè£ŹäžŻæăijåijŔåŇŰæŞ■ä¡IJçňęçŽĎåĂijæİěèĞłäžŐçŤÍæĹůçŽĎè¿ŞåĚěïijŇéĆčäźĹä¡ăçŽ
http://xkcd.com/327)ãĂĆ æ§èěŕćèŕ■åŔëäÿ■çŽĎéĂŽéĚ■çňę ?
æŇĞçď'žåŘŐåŔřæŢřæ■őåžŞä¡£çŤÍåőČèĞłåůśçŽĎå■ŮçňęäÿšæŻ£æ■ćæIJžåĹűïijŇè£ŹæăůæŻt'åŁăçŽĎåőL'å

äÿ■åźÿçŽĎæŸŕïijŇäÿ■åŘŇçŽĎæŢřæ■őåžŞåŘŐåŔřåŕźäžŐéĂŽéĚ■çňęçŽĎä¡£çŤÍæŸŕäÿ■äÿĂæăůçŽĎãĂ
? æĹŰ %s ïijŇ è£ŸæIJL'åĚűäžŰäÿĂäžŻä¡£çŤÍäžĘäÿ■åŘŇçŽĎçňęåŘůïijŇærŤåęĆ:0æĹŰ:1æİěæŇĞçď'žåŔĆ
åŘŇæăůçŽĎïijŇä¡ăè£ŸæŸŕå¿ŮåŐżåŔĆèĂČä¡ăä¡£çŤÍçŽĎæŢřæ■őåžŞæĺąåİŮçŻÿåžŤçŽĎæŰĞæąčãĂĆ
äÿĂäÿłæŢřæ■őåžŞæĺąåİŮçŽĎ paramstyle åśđæĂğåŇĚåŘńäžĘåŔĆæŢřåijŢçŤÍéčŐæäijçŽĎä£ąæĄřãĂĆ

åŕźäžŐçőÅå■ŢçŽĎæŢřæ■őåžŞæŢřæ■őçŽĎèŕżåĘŹéŮőécŸïijŇä¡£çŤÍæŢřæ■őåžŞAPIéĂŽåÿÿéİđåÿÿçőĂ
åęĆæđIJä¡ăèęĄåď'ĎçŘĘæŻt'åŁăåď'■æİĆçŽĎéŮőécŸïijŇåżžèőőä¡ăä¡£çŤÍæŻt'åŁäéńŸçžğçŽĎæŐěåŔčïijŇæí
çśżäijij SQLAlchemy è£ŹæăůçŽĎåžŞåĚĄèőÿä¡ăä¡£çŤÍPythonçśżæİěèąÍçď'žäÿĂäÿłæŢřæ■őåžŞèąÍïijŇ
åźűäÿŤèČ¡åIJÍéŽŘèŮŔåžŢåśCSQLçŽĎæČĚåĘţäÿŃåőđçŐřåŘĎçğ■æŢřæ■őåžŞçŽĎæŞ■ä¡IJãĂĆ

6.9 çijŰçăĄåŠŇèğčçăĄå■ĄåĚ■è£ŻåĹűæŢř

éŮőécŸ

ä¡ăæČşåřĘäÿĂäÿłå■ĄåĚ■è£ŻåĹűå■Ůçňęäÿšèğčçă ĄæĹŘäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšæĹŰèĂĚåřĘäÿĂäÿłå

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăåŔłæŸŕçőÅå■ŢçŽĎèğčçăĄæĹŰçijŰçăĄäÿĂäÿłå■ĄåĚ■è£ŻåĹűçŽĎåŐ§ågŇå■ŮçňęäÿšïijŇă
æĺąåİŮãĂĆä¿ŃåęĆïijŽ

```
>>> # Initial byte string
>>> s = b'hello'
>>> # Encode as hex
>>> import binascii
>>> h = binascii.b2a_hex(s)
>>> h
b'68656c6c6f'
>>> # Decode back to bytes
>>> binascii.a2b_hex(h)
b'hello'
>>>
```

çśżäijijçŽĎåŁ§èČ¡åŘŇæăůåŔŕäžěåIJÍ base64 æĺąåİŮäÿ■æL'¿åĹřãĂĆä¿ŃåęĆïijŽ

```
>>> import base64
>>> h = base64.b16encode(s)
>>> h
b'68656C6C6F'
```

```
>>> base64.b16decode(h)
b'hello'
>>>
```

èőÍèőž

åď ğéČÍåĹĘæČĚåĘţäÿŃïijŇéĂŽè£Ğä¡£çŤÍäÿŁè£řçŽĎåĞ¡æŢřæİěè¡ňæ■ćå■ĄåĚ■è£ŻåĹűæŸřå¿ĹçőĂå■Ţ
äÿŁéİćäÿď çğ■æŁĂæIJŕçŽĎäÿżèęĄäÿ■åŘŇåIJÍäžŐåď ğářŔåĘŹçŽĎåď ĎçŘĘãĂĆ
åĞ¡æŢř `base64.b16decode()` åŠŇ `base64.b16encode()`
åŔłèČ¡æŞ■ä¡IJåď ğåĘŹå¡ćåijŔçŽĎå■ĄåĚ■è£ŻåĹűå■Ůæŕ■ïijŇ èĂŇ `binascii`
æĺąåİŮäÿ■çŽĎåĞ¡æŢřåď ğářŔåĘŹéČ¡èČ¡åď ĎçŘĘãĂĆ

è£ŸæIJL’äÿĂçĆźéIJĂèęĄæşĺæĎŔçŽĎæŸŕçijŰçăĄåĞ¡æŢřæL’Ăäžğç Ť§çŽĎè¿ŞåĞžæĂżæŸŕäÿĂäÿłå■Ů
åęĆæđIJæČşåijžåĹűäžéUnicodeå¡ćåijŔè¿ŞåĞžïijŇä¡ăéIJĂèęĄåćđåŁääÿĂäÿłéćİåď ŰçŽĎçŢŇéİćæ■ěélď ãĂĆ

```
>>> h = base64.b16encode(s)
>>> print(h)
b'68656C6C6F'
>>> print(h.decode('ascii'))
68656C6C6F
>>>
```

åIJÍèğčçăĄå■ĄåĚ■è£ŻåĹűæŢřæŮűïijŇåĞ¡æŢř `b16decode()`
åŠŇ `a2b_hex()` åŔŕäžěæŐěåŔŮå■ŮèŁĆæĹŰunicodeå■Ůçňęäÿšã ĂĆ
ä¡ĘæŸŕïijŇunicodeå■Ůçňęäÿšå£ĚéążäžĚäžĚåŔłåŇĚåŘńASCIIçijŰçăĄçŽĎå■ĄåĚ■è£ŻåĹűæŢřã ĂĆ

6.10 çijŰçăĄèğčçăĄBase64æŢřæ■ő

éŮőécŸ

ä¡ăéIJĂèęĄä¡£çŤÍBase64æăijåijŔèğčçăĄæĹŰçijŰçăĄäžŇè£ŻåĹűæŢřæ■õã ĂĆ

èğčåĘşæŰźæąĹ

`base64` æĺąåİŮäÿ■æIJL’äÿď äÿłåĞ¡æŢř `b64encode()` and `b64decode()`
åŔŕäžěåÿőä¡ăèğčåĘşè£ŹäÿłéŮőécŸãĂĆä¿ŃåęĆ;

```
>>> # Some byte data
>>> s = b'hello'
>>> import base64

>>> # Encode as Base64
>>> a = base64.b64encode(s)
>>> a
b'aGVsbG8='
```

```
>>> # Decode from Base64
>>> base64.b64decode(a)
b'hello'
>>>
```

èőÍèőž

Base64çijŰçăĄäžËäžĚçŤÍäžŐéİćåŘŚå■ŮèŁĆçŽĎæŢřæ■őæŕŤåęĆå■ŮèŁĆå■Ůçňęäÿšåš
æ■d'åd'ŰïijŇçijŰçăĄåd'ĎçŘĘçŽĎè¿ŞåĞžçżŞæđIJæĂżæŸŕäÿĂäÿłå■ŮèŁĆå■Ůçňęäÿšã
åęĆæđIJä¡ăæČşæůůåŘĹä¡£çŤÍBase64çijŰçăĄçŽĎæŢřæ■őåŠŇUnicodeæŰĞæIJňïijŇä¡ăå£Ëéążæůżå

```
>>> a = base64.b64encode(s).decode('ascii')
>>> a
'aGVsbG8='
>>>
```

å¡Şèğčçă Base64çŽĎæŮűåĂŹïijŇå■ŮèŁĆå■Ůçňęäÿšå
ä¡ĘæŸŕïijŇUnicodeå■Ůçňęäÿšå

6.11 èŕżåĘŹäžŇè£ŻåĹűæŢřçżĎæŢřæ■ő

éŮőécŸ

ä¡ăæČşèŕżåĘŹäÿĂäÿłäžŇè£ŻåĹűæŢřçżĎçŽĎçżŞæđĎåŇŰæŢřæ■őåĹřPythonåĚČçżĎäÿ■ãĂĆ

èğčåĘşæŰźæąĹ

åŘŕäżěä¡£çŤÍ struct æĺąåİŮåd'ĎçŘĘäžŇè£ŻåĹűæŢřæ■őãĂĆ
äÿŇéİćæŸŕäÿĂæőţçd'žä¿Ńäżčçă ĄåřĘäÿĂäÿłPythonåĚČçżĎåĹŮèąĺåĘŹåĚěäÿĂäÿłäžŇè£ŻåĹűæŰĞäžűïijŇå
struct åřĘæŕŘäÿłåĚČçżĎçijŰçăĄäÿžäÿĂäÿłçżŞæđĎä¡ŞãĂĆ

```
from struct import Struct
def write_records(records, format, f):
    '''
    Write a sequence of tuples to a binary file of structures.
    '''
    record_struct = Struct(format)
    for r in records:
        f.write(record_struct.pack(*r))

# Example
if __name__ == '__main__':
    records = [ (1, 2.3, 4.5),
                (6, 7.8, 9.0),
                (12, 13.4, 56.7) ]
```

```
    with open('data.b', 'wb') as f:
        write_records(records, '<idd', f)
```

æIJL'å¿Ĺåd'Žçğ■æŰźæşŢæİèèŕżåŔŰè£ŹäÿłæŰĞäżűåźűè£ŤåŻđäÿĂäÿłåĚČçżĎåĹŮèąĺãĂĆ éęŰåĚĹïijŇåęĆæđIJä¡ăæL'ŞçőŮäžěåİŮçŽĎå¡ćåijŔåćđéĞŔèŕżåŔŰæŰĞäżűïijŇä¡ääŔŕäżěè£ŹæăůåĄŽïijŽ

```
from struct import Struct

def read_records(format, f):
    record_struct = Struct(format)
    chunks = iter(lambda: f.read(record_struct.size), b'')
    return (record_struct.unpack(chunk) for chunk in chunks)

# Example
if __name__ == '__main__':
    with open('data.b','rb') as f:
        for rec in read_records('<idd', f):
            # Process rec
            ...
```

åęĆæđIJä¡ăæČşåřĘæŢť'äÿłæŰĞäżűäÿĂæñąæĂğèŕżåŔŰåĹřäÿĂäÿłå■ŮèŁĆå■Ůçňęäÿšäÿ■ïijŇçĎűåŘŐåĽ

```
from struct import Struct

def unpack_records(format, data):
    record_struct = Struct(format)
    return (record_struct.unpack_from(data, offset)
            for offset in range(0, len(data), record_struct.size))

# Example
if __name__ == '__main__':
    with open('data.b', 'rb') as f:
        data = f.read()
    for rec in unpack_records('<idd', data):
        # Process rec
        ...
```

äÿd'çğ■æČĚåĘţäÿŇçŽĎçżŞæđIJéČ¡æŸŕäÿĂäÿłåŔŕè£ŤåŻđçŤĺæİèåĹŻåżžèŕěæŰĞäżűçŽĎåŐ§åğŃåĚČçż

èőĺèőž

åŕźäžŐéIJĂèęĄçijŰçăĄåŠŇèğčçăĄäžŇè£ŻåĹűæŢřæ■őçŽĎçĺŃåžŔèĂŇèĺĂïijŇéĂŽåÿÿäijŽä¡£çŤĺ `struct` æĺąåİŮãĂĆ äÿžäžĘåčřæŸŐäÿĂäÿłæŰřçŽĎçżŞæđĎä¡ŞïijŇåŕłéIJĂèęĄåČŔè£ŹæăůåĹŻåżžäÿĂäÿł `Struct` åőđä¿Ńå■şåŔŕïijŽ

```
# Little endian 32-bit integer, two double precision floats
record_struct = Struct('<idd')
```

çżŞæđĎä¡ŞéĂŽåÿÿäijŽä¡£çŤĺäÿĂäžŻçżŞæđĎçăĄåĂiji, d, fç■L' [åŔĆèĂČ PythonæŰĞæąč]ãĂĆ è£ŹäžŻäžčçăĄåĹĘåĹńäžčèąĺæ§ŘäÿłçL'źåőŽçŽĎäžŇè£ŻåĹűæŢřæ■őçśžåđŃåęĆ32ä¡■a

çňňäÿĂäÿłå■Ůçňę < æŇĞåőŽäžĘå■ÚèŁĆéąžåžŔãĂĆåIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇåőČèąÍçd'ž"ä¡Őä¡■åIJÍåL'■"ã
æŽť æŤźè£Źäÿłå■Ůçňęäÿž > èąÍçd'žéńŸä¡■åIJÍåL'■ïijŇæĹŰèĂĚæŸŕ !
èąÍçd'žç¡ŚçżIJå■ÚèŁĆéąžåžŔãĂĆ

äžğçŤ§çŽĎ `Struct` åőđä¿ŃæIJL'å¿Ĺåd'Žåśđæ Ăğå ŠŇæŰźæşŢçŤÍæİěæŞ■ä¡IJçŻÿåžŤçśżåđŃçŽĎçżŞæđ
`size` åśđæĂğåŇĚåŘńäžĘçżŞæđĎçŽĎå■ÚèŁĆæŢřïijŇè£ŹåIJÍI/OæŞ■ä¡IJæŮűéİđåÿÿæIJL'çŤÍãĂĆ
`pack()` åŠŇ `unpack()` æŰźæşŢèćńçŤÍæİěæL'ŞåŇĚåŠŇèğčåŇĚæŢřæ■őãĂĆæŕŤåęĆïijŽ

```
>>> from struct import Struct
>>> record_struct = Struct('<idd')
>>> record_struct.size
20
>>> record_struct.pack(1, 2.0, 3.0)
b
 ↪'\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00@\x00\x00\x00\x00\x00\x00\x08@
 ↪'
>>> record_struct.unpack(_)
(1, 2.0, 3.0)
>>>
```

æIJL'æŮűåĂŹä¡àè£ŸäijŽçIJŃåĹř `pack()` åŠŇ `unpack()`
æŞ■ä¡IJäžěæĺąåİŮçžğåĹńåĞ¡æŢřèćńèřČçŤÍïijŇçśżäijijäÿŃéİćè£Źæăůïij Ž

```
>>> import struct
>>> struct.pack('<idd', 1, 2.0, 3.0)
b
 ↪'\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00@\x00\x00\x00\x00\x00\x00\x08@
 ↪'
>>> struct.unpack('<idd', _)
(1, 2.0, 3.0)
>>>
```

è£ŹæăůåŔŕäžěåůëä¡IJïijŇä¡ĘæŸŕæĎ§èğL'æšąæIJL'åőđä¿ŃæŰźæşŢéĆčäźĹäijŸéŻĚïijŇçL'źåĹńæŸŕåIJÍà
éĂŽè£ĞåĹŽåžžäÿĂäÿł `Struct` åőđä¿ŃïijŇæäijåijŔäžčçăĄåŔłäijŽæŇĞåőŽäÿĂæňąåźűäÿŤæL'ĂæIJL'çŽĎæ
è£ŹæăůäÿĂæİěäžčçăĄçžť'æŁd'åŕśåŔŸå¿ŮæŽť'åŁăçőĂå■ŢäžĘ(åŽääÿžä¡ääŔłéIJĂèęĄæŤźåŔŸäÿĂåd'Đäžčçă

èŕżåŔŰäžŇè£ŽåĹűçżŞæđĎçŽĎäžčçăĄèęĄçŤÍåĹřäÿĂäžŻéİđåÿÿæIJL'èűčèĂŇäijŸç¿ŐçŽĎçijŰçÍŃæŁĂ
åIJÍåĞ¡æŢřãĂĂ`read_records` äÿ■ïijŇ`iter()` èćńçŤÍæİěåĹŽåžžäÿĂäÿłè£ŤåŽđåŽžåőŽåd'ğåřŔæŢřæ■őå
è£ŹäÿłèŁ■äžčåŹÍäijŽäÿ■æŰ■çŽĎèřČçŤÍäÿĂäÿłçŤÍæĹůæŔŘä¿ŻçŽĎåŔŕèřČçŤÍåŕźèśą(æŕŤåęĆ
`lambda: f.read(record_struct.size)`)ïijŇ çŽť'åĹřåőČè£ŤåŽđäÿĂäÿłçL'źæőŁçŽĎåĂij(åęĆb''

```
>>> f = open('data.b', 'rb')
>>> chunks = iter(lambda: f.read(20), b'')
>>> chunks
<callable_iterator object at 0x10069e6d0>
>>> for chk in chunks:
...     print(chk)
...
b'\x01\x00\x00\x00ffffff\x02@\x00\x00\x00\x00\x00\x00\x12@'
b'\x06\x00\x00\x00333333\x1f@\x00\x00\x00\x00\x00\x00"@'
b'\x0c\x00\x00\x00\xcd\xcc\xcc\xcc\xcc\xcc*@\x9a\x99\x99\x99\x99YL@'
>>>
```

åęĆä¡ăæĽĂèğĄïijŇåĹŻåžžäÿĂäÿłåŔřè£■äżčářźèśąçŽĎäÿĂäÿłåŐ§åŽăæŸřåőČèČ¡åĚĄèőÿä¡£çŤĺäÿĂäÿ
åęĆæđIJä¡ääÿ■ä¡£çŤĺè£Źçğ■æŁĂæIJřïijŇéĆčäźĹäžčçăĄåŔřèČ¡äijŽåČŔäÿŇéİćè£Źæăůïijž

```
def read_records(format, f):
    record_struct = Struct(format)
    while True:
        chk = f.read(record_struct.size)
        if chk == b'':
            break
        yield record_struct.unpack(chk)
```

åIJĺåĞ¡æŢř `unpack_records()` äÿ■ä¡£çŤĺäžĘåŔęåđ'ŰäÿĂçğ■æŰźæşŢ `unpack_from()` ãĂĆ `unpack_from()` åŕźäžŐäżŐäÿĂäÿłåđ'ğđŇäžŇè£ŻåĹűæŢřçżĎäÿ■æŔŘåŔŰäžŇè
åŽääÿžåőČäÿ■äijŽäžğçŤ§äżża¡ŢçŽĎäÿť'æŮűåŕźèśąæĹŰèĂĚè£Żèą̃ŇåĘĚå■Ÿåđ'■åĹűæŞ■ä¡IJãĂĆ
ä¡ăåŘłéIJĂèęĄçżŹåőČäÿĂäÿłå■ŮèŁĆå■Ůçňęäÿš(æĹŰæŢřçżĎ)åŠŇäÿĂäÿłå■ŮèŁĆåĄŔçğżéĞŔïijŇåőČäijŽa

åęĆæđIJä¡ăä¡£çŤĺ `unpack()` æİěäžčæŻ£ `unpack_from()` ïijŇ
ä¡ăéIJĂèęĄä£őæŤźäžčçăĄæİěæđĎéĂăåđ'ğéĞŔçŽĎåřŔçŽĎåĹĞçL'Ğäžěåŕŀè£ŻèąŇåĄŔçğżéĞŔçŽĎèőąçőŮ

```
def unpack_records(format, data):
    record_struct = Struct(format)
    return (record_struct.unpack(data[offset:offset + record_struct.
↪size])
            for offset in range(0, len(data), record_struct.size))
```

è£Źçğ■æŰźæąĹéŹđ'äžĘäžčçăĄçIJŃäÿŁåŐżå¿Ĺåđ'■æİĆåđ'ŰïijŇè£Ÿå¿ŮåĄŽå¿Ĺåđ'Žéćİåđ'ŰçŽĎåůěä¡
åđ'■åĹűæŢřæ■őäžěåŔŁæđĎéĂăåřŔçŽĎåĹĞçL'Ğåŕźèśąã̃ĆĆ åęĆæđIJä¡ääĞĘåđ'ĞäżŐèŕżåŔŰåĹřçŽĎäÿĂäÿłå
äijŽèąĺçŐřçŽĎæŻť'åĞžèL'šãĂĆ

åIJĺèğčåŇĚçŽĎæŮűåĂŹïijŇ`collections` æĺąåİŮäÿ■çŽĎåŚ¡åŘ■åĚČçżĎåŕźèśąæĹŰèőÿæŸřä¡ăæČşè
åőČåŘŕäžěèőľ'ä¡ăçżŹè£ŤåŽđåĚČçżĎèő¿ç¡őåśđæĂğåŘ■çğřãĂĆä¿ŃåęĆïijž

```
from collections import namedtuple

Record = namedtuple('Record', ['kind','x','y'])

with open('data.p', 'rb') as f:
    records = (Record(*r) for r in read_records('<idd', f))

for r in records:
    print(r.kind, r.x, r.y)
```

åęĆæđIJä¡ăçŽĎçÍŃåžŔéIJĂèęĄåđ'ĎçŘĘåđ'ğéĞŔçŽĎäžŇè£ŻåĹűæŢřæ■őïijŇä¡ăæIJĂåě¡ä¡£çŤĺ
`numpy` æĺąåİŮãĂĆ ä¿ŃåęĆïijŇä¡ääŔŕäžěårĘäÿĂäÿłäžŇè£ŻåĹűæŢřæ■õèŕżåŔŰåĹřäÿĂäÿłçżŞæđĎåŇŰæŢřçż

```
>>> import numpy as np
>>> f = open('data.b', 'rb')
>>> records = np.fromfile(f, dtype='<i,<d,<d')
>>> records
array([(1, 2.3, 4.5), (6, 7.8, 9.0), (12, 13.4, 56.7)],
```

```
dtype=[('f0', '<i4'), ('f1', '<f8'), ('f2', '<f8')])
>>> records[0]
(1, 2.3, 4.5)
>>> records[1]
(6, 7.8, 9.0)
>>>
```

æIJÄåŘŐæŔŘäÿĂçĆźïijŇåęĆæđIJä¡ăéIJÄèęĄäżŐåůšç§ěçŽĎæŰĞäžűæäijåijŔ(åęĆåŻ¿çL'Ğæäijåij
åĚĹæčÄæ§ěçIJŃçIJŃPythonæŸŕäÿ■æŸŕåůšçżŔæŔŘä¿ŻäžEçŐřå■ŸçŽĎæĺąåİŮãĂĆåŻääÿžäÿ■åĹřäÿĞäÿ■å¿

6.12 èŕżåŔŰåţŇåěŮåŠŇåŔŕåŔŸéŢ£äžŇè£ŻåĹűæŢřæ■ő

éŮőécŸ

ä¡ăéIJÄèęĄèŕżåŔŰåŇĚåŘńåţŇåěŮæĹŰèĂĚåŔŕåŔŸéŢ£èőřå¡ŢéŻĘåŘĹçŽĎåď■æİĆäžŇè£ŻåĹűæäijåijŔ

èğčåĘşæŰźæąĹ

`struct` æĺąåİŮåŔŕèćńçŤĺæİěçijŰçăĄ/èğčçăĄåĞääźŐæL'ĂæIJL'çśżåđŃçŽĎäžŇè£ŻåĹűçŽĎæŢřæ■őçż
æİěèąĺçď'žäÿĂäÿłçżĎæĹŘäÿĂçşżåĹŮåď'Žè¿źå¡ćçŽĎçĆźçŽĎéŻĘåŘĹïijŽ

```
polys = [
    [ (1.0, 2.5), (3.5, 4.0), (2.5, 1.5) ],
    [ (7.0, 1.2), (5.1, 3.0), (0.5, 7.5), (0.8, 9.0) ],
    [ (3.4, 6.3), (1.2, 0.5), (4.6, 9.2) ],
]
```

çŐřåIJĺåĄĞèő¿è£ŹäÿłæŢřæ■őèćńçijŰçăĄåĹřäÿĂäÿłäżěäÿŇåĹŮåď't'éČĺåijĂåğŃçŽĎäžŇè£ŻåĹűæŰĞäż

```
+------+--------+-----------------------------------+
|Byte  | Type   |  Description                      |
+======+========+===================================+
|0     | int    |  æŰĞäżűäżčçăĄïijĹ0x1234ïijŇåřŔçńŕïijL'          |
+------+--------+-----------------------------------+
|4     | double |  x çŽĎæIJÄåřŔåĂijïijĹåřŔçńŕïijL'                |
+------+--------+-----------------------------------+
|12    | double |  y çŽĎæIJÄåřŔåĂijïijĹåřŔçńŕïijL'                |
+------+--------+-----------------------------------+
|20    | double |  x çŽĎæIJÄåd'ğåĂijïijĹåřŔçńŕïijL'                |
+------+--------+-----------------------------------+
|28    | double |  y çŽĎæIJÄåd'ğåĂijïijĹåřŔçńŕïijL'                |
+------+--------+-----------------------------------+
|36    | int    |  äÿL'èğŠå¡ćæŢřéĞŔïijĹåřŔçńŕïijL'                |
+------+--------+-----------------------------------+
```

çt'ğèů§çİÄåď't'éČĺæŸŕäÿĂçşżåĹŮçŽĎåď'Žè¿źå¡ćèőřå¡ŢïijŇçijŰçăĄæăijåijŔåęĆäÿŇïijŽ

```
+------+--------+-----------------------------------------+
|Byte  | Type   |  Description                            |
+======+========+=========================================+
|0     | int    |  èőřå¡ŢéŢ£åžęïijĹNå■ŮèŁĆïijL’                |
 ↪  |
+------+--------+-----------------------------------------+
|4-N   | Points |  (X,Y) åİŘæăĞïijŇäżěætőçĆźæŢřèąÍçd’ž       |
 ↪    |
+------+--------+-----------------------------------------+
```

äÿžäžĘåĘŹè£ŹæăůçŽĎæŰĞäżűïijŇä¡ääŔŕäžěä¡£çŤĺåęĆäÿŃçŽĎPythonäżčçăĄïijŽ

```
import struct
import itertools

def write_polys(filename, polys):
    # Determine bounding box
    flattened = list(itertools.chain(*polys))
    min_x = min(x for x, y in flattened)
    max_x = max(x for x, y in flattened)
    min_y = min(y for x, y in flattened)
    max_y = max(y for x, y in flattened)
    with open(filename, 'wb') as f:
        f.write(struct.pack('<iddddi', 0x1234,
                            min_x, min_y,
                            max_x, max_y,
                            len(polys)))
        for poly in polys:
            size = len(poly) * struct.calcsize('<dd')
            f.write(struct.pack('<i', size + 4))
            for pt in poly:
                f.write(struct.pack('<dd', *pt))
```

åřĘæŢřæ■őèŕžåŔŮåŻđæİěçŽĎæŮűůåĂŹïijŇåŔŕäžěåĹľçŤĺåĞ¡æŢř `struct.unpack()` ïijŇäžčçăĄå¿ĹçŻÿäijijïijŇå§žæIJňářśæŸŕäÿŁéİćåĘŹæŞ■ä¡IJçŽĎéĂĘåžŔãĂĆåęĆäÿŃïijŽ

```
def read_polys(filename):
    with open(filename, 'rb') as f:
        # Read the header
        header = f.read(40)
        file_code, min_x, min_y, max_x, max_y, num_polys = \
            struct.unpack('<iddddi', header)
        polys = []
        for n in range(num_polys):
            pbytes, = struct.unpack('<i', f.read(4))
            poly = []
            for m in range(pbytes // 16):
                pt = struct.unpack('<dd', f.read(16))
                poly.append(pt)
            polys.append(poly)
    return polys
```

åř¡çőąè£ŹäÿłäžčçăĄåŔŕäžěåůěä¡IJïijŇä¡ĘæŸŕéĞŇéİćæůåæİĆäžĘå¿Ĺåď Žèŕžå ŔŰãĂĄèğčåŇĚæŢřæ■őçž
éĆčæIJłåĚ■äź§åď łçźĄæİĆäžĘçĆźãĂĆåŻäæ■ď å¿ĹæŸ¿çĎűåžŤèŕěæIJL åŔęäÿĂçğ■èğčåĘşæŰźæşŢåŔŕäžěçő

åIJÍæIJňåřŔèŁĆæŐěäÿŃæİěçŽĎéČĺåĹĘïijŇæĹŚäijŽéĂŘæ■ěæijŤçď žäÿĂäÿłæŻť åŁääijŸçğĂçŽĎèğčæć
çŻőæăĞæŸŕåŔŕäžěçżŹçÍŃåžŔåŚŸæŔŘä¿ŻäÿĂäÿłéńŸçžğçŽĎæŰĞäžűæăijåijŔåŇŰæŰźæşŢïijŇåźűçőĂåŇŰ
æIJňåřŔèŁĆæŐěäÿŃæİěçŽĎéČĺåĹĘäžčçăĄåžŤèŕěæŸŕæŢť æIJňäźęäÿ■æIJĂåď■æİĆæIJĂéńŸçžğçŽĎä¿Ńå■İ
äÿĂåőŻèęĄäżŤçżĘçŽĎéŸĚèŕżæĹŚäžňçŽĎèőÍèőžéČĺåĹĘïijŇåŔęåď Űäź§èęĄåŔĆèĂČäÿŃåĚűäžŰçńäèŁĆåĘ

éęŰåĚĹïijŇå¡Şèŕżå ŔŰå■ŮèŁĆæŢřæ■őçŽĎæŮűåĂŹïijŇéĂŽåÿÿåIJÍæŰĞäžűåijĂåğŃéČĺåĹĘäijŽåŇĚåŘ
åř¡çőąstructæĺąåİŮåŔŕäžěèğčåŇĚè£ŹäžŻæŢřæ■őåĹřäÿĂäÿłåĚČçżĎäÿ■åŐżïijŇåŔęåď ŰäÿĂçğ■èąÍçď žè£ŹçğĘ
åřšåČŔäÿŃéİćè£ŹæăůïijŽ

```
import struct

class StructField:
    '''
    Descriptor representing a simple structure field
    '''
    def __init__(self, format, offset):
        self.format = format
        self.offset = offset
    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            r = struct.unpack_from(self.format, instance._buffer,␣
↪self.offset)
            return r[0] if len(r) == 1 else r

class Structure:
    def __init__(self, bytedata):
        self._buffer = memoryview(bytedata)
```

è£ŹéĞŇæĹŚäžňä¡£çŤĺäžĘäÿĂäÿłæŔŘè£řåŹĺæİěèąÍçď žæŕŔäÿłçżŞæđĎå■ŮæőţïijŇæŕŔäÿłæŔŘè£řåŹĺåŇ
å■ŸåĆĺåIJĺåĘĚéČĺçŽĎåĘĚå■ŸçijŞåĘšäÿ■ãĂĆåIJÍ `__get__()` æŰźæşŢäÿ■ïijŇstruct.
`unpack_from()` åĞ¡æŢřèćńçŤĺæİěäžŐçijŞåĘšäÿ■èğčåŇĚäÿĂäÿłåĂijïijŇçIJĄåŐżäžĘéćİåď ŰçŽĎåĹĘçĽČ

`Structure` çśžåřśæŸŕäÿĂäÿłå§žçąĂçśżïijŇæŐěåŘŮå■ŮèŁĆæŢřæ■őåźűå■ŸåĆĺåIJĺåĘĚéČĺçŽĎåĘĚå■
`StructField` æŔŘè£řåŹĺä¡£çŤĺãĂĆ è£ŹéĞŇä¡£çŤĺäžĘ `memoryview()`
ïijŇæĹŚäžňäijŽåIJĺåŘŐéİćèŕęçżĘèőšèğčåőĆæŸŕçŤĺæİěåźšåŸŻçŽĎãĂĆ

ä¡£çŤĺè£ŹäÿłäžčçăĄïijŇä¡ăçŐřåIJĺåřśèČ¡åőŽäźĽäÿĂäÿłéńŸåśĆæňąçŽĎçżŞæđĎåŕźèśąæİěèąÍçď žäÿŁéİ

```
class PolyHeader(Structure):
    file_code = StructField('<i', 0)
    min_x = StructField('<d', 4)
    min_y = StructField('<d', 12)
    max_x = StructField('<d', 20)
    max_y = StructField('<d', 28)
    num_polys = StructField('<i', 36)
```

äÿŃéÌćçŽĎä¿Ńå■ŘåĹl’çŤĺè£ŹäÿłçśżæİěèřżåŔŰäźŃåL’■æĹŚäžňåĘŹåĚěçŽĎåď’Žè¿źå¡ćæŢřæ■őçŽĎåď’t’

```
>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader(f.read(40))
>>> phead.file_code == 0x1234
True
>>> phead.min_x
0.5
>>> phead.min_y
0.5
>>> phead.max_x
7.0
>>> phead.max_y
9.2
>>> phead.num_polys
3
>>>
```

è£Źäÿłå¿ĹæIJL’èűčïijŇäÿ■è£Ğè£Źçğ■æŰźåijŔè£ŸæŸŕæIJL’äÿĂäžŽçČęäžžçŽĎåIJřæŰźãĂĆéęŰåĚĹïijŇ
ä¡ĘæŸŕè£Źäÿłäžčçăąè£ŸæŸŕæIJL’çĆźèĞČèĆ£ïijŇè£ŸéIJĂèęąä¡£çŤĺèĂĚæŇĞåőŽå¿Ĺåď’ŽåžŢåśĆçŽĎçżĘ
`StructField`ïijŇæŇĞåőŽåĄŕçğžéĞŔç■L’)ãĂĆ åŕęåď’ŰïijŇè£ŤåŽđçŽĎçżŞæđIJçśżåŘŇæăůçąőåőđäÿĂä

äžżä¡ŢæŮűåĂŹåŔłèęąä¡ăéĄĞåĹřäžĘåČŔè£Źæăůå ĘŮä¡ŹçŽĎçśżåőŽäźL’ïijŇä¡ăåžŤèŕěèĂČèŹŚäÿŃä¡£ç
åĚČçśżæIJL’äÿĂäÿłçL’źæĂğåřśæŸŕåőČèČ¡åď’§èćńçŤĺæİěåąńåĚĚèőÿåď’Žä¡ŐåśĆçŽĎåőđçŐřçżĘèŁĆïijŇäžŐ
äÿŃéÌćæĹŚæİěäÿ¿äÿłä¿Ńå■ŘïijŇä¡£çŤĺåĚČçśżçĺ■å¿őæŤźéĂääÿŃæĹŚäžňçŽĎ `Structure`
çśżïijŽ

```
class StructureMeta(type):
    '''
    Metaclass that automatically creates StructField descriptors
    '''
    def __init__(self, clsname, bases, clsdict):
        fields = getattr(self, '_fields_', [])
        byte_order = ''
        offset = 0
        for format, fieldname in fields:
            if format.startswith(('<','>','!','@')):
                byte_order = format[0]
                format = format[1:]
            format = byte_order + format
            setattr(self, fieldname, StructField(format, offset))
            offset += struct.calcsize(format)
        setattr(self, 'struct_size', offset)

class Structure(metaclass=StructureMeta):
    def __init__(self, bytedata):
        self._buffer = bytedata

    @classmethod
    def from_file(cls, f):
        return cls(f.read(cls.struct_size))
```

ä¡£çŤĺæŰřçŽĎ `Structure` çśżïijŇä¡ååŔŕäžěåČŔäÿŃéİćè£ŹæăůåőŽäźĽäÿĂäÿłçżŞæđĎïijŽ

```
class PolyHeader(Structure):
    _fields_ = [
        ('<i', 'file_code'),
        ('d', 'min_x'),
        ('d', 'min_y'),
        ('d', 'max_x'),
        ('d', 'max_y'),
        ('i', 'num_polys')
    ]
```

æ■čåęĆä¡ăæĽĂèğĄïijŇè£ŹæăůåĘŹåřśçőĂå■Ţåď˙ŽäžĘãĂĆæĹŚäžňæůżåŁăçŽĎçśżæŰźæşŢ `from_file()` èőľ˙æĹŚäžňåIJĺäÿ■éIJĂèęĄç§ěéĄŞäżżä¡ŢæŢřæ■őçŽĎåď˙ğářŔåŠŇçżŞæđĎçŽĎæČĚåĘţäÿŃă

```
>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader.from_file(f)
>>> phead.file_code == 0x1234
True
>>> phead.min_x
0.5
>>> phead.min_y
0.5
>>> phead.max_x
7.0
>>> phead.max_y
9.2
>>> phead.num_polys
3
>>>
```

äÿĂæŮęä¡ăăijĂåğŃä¡£çŤĺäžĘåĚČçśżïijŇä¡ăářśåŔŕäžěèőľ˙åőČåŔŸå¿ŮæŻť˙åŁăæŹžèČ¡ãĂĆä¿ŃåęĆïijŇå äÿŃéİćæŸŕářźåĽ■éİćåĚČçśżçŽĎäÿĂäÿłářŔçŽĎæŤźè£ŻïijŇæŔŘä¿ŻäžĘäÿĂäÿłæŰřçŽĎè¿ĚåŁľ˙æŔŘè£řåŹĺa

```
class NestedStruct:
    '''
    Descriptor representing a nested structure
    '''
    def __init__(self, name, struct_type, offset):
        self.name = name
        self.struct_type = struct_type
        self.offset = offset

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            data = instance._buffer[self.offset:
                                    self.offset+self.struct_type.struct_
↪size]
            result = self.struct_type(data)
```

```
            # Save resulting structure back on instance to avoid
            # further recomputation of this step
            setattr(instance, self.name, result)
            return result

class StructureMeta(type):
    '''
    Metaclass that automatically creates StructField descriptors
    '''
    def __init__(self, clsname, bases, clsdict):
        fields = getattr(self, '_fields_', [])
        byte_order = ''
        offset = 0
        for format, fieldname in fields:
            if isinstance(format, StructureMeta):
                setattr(self, fieldname,
                        NestedStruct(fieldname, format, offset))
                offset += format.struct_size
            else:
                if format.startswith(('<','>','!','@')):
                    byte_order = format[0]
                    format = format[1:]
                format = byte_order + format
                setattr(self, fieldname, StructField(format,␣
↪offset))
                offset += struct.calcsize(format)
        setattr(self, 'struct_size', offset)
```

åIJÍè£Źæőţäžčçă Ąäÿ■ïijÑNestedStruct æŔŕè£řåŹÍèćńçŤÍæİěåŔäåŁäåŔęåd'ŰäÿĂäÿłåőŽäźL'åIJÍæ§
åőČéĂŽè£ĞářĘåŐ§ågŃåĘÉå■ŸçijŞåĘšè£ŻèąŇåĹĞçL'ĞæŞ■ä¡IJåŘŐåőđä¿ŃåŇŰçżŹåőŽçŽĎçżŞæđĎçśžåđŇ
æL'Ăäžěè£Źçğ■åĹĞçL'ĞæŞ■ä¡IJäÿ■äijŽåijŢåŔŚäżżä¡ŢçŽĎéćİåd'ŰçŽĎåĘĚå■Ÿåd'■åĹűãĂĆçŻÿåŔ■ïijŇåőČ
åŔęåd'ŰïijÑäÿžäžĘéŸšæ■ćéĞ■åd'■åőđä¿ŃåŇŰïijŇéĂŽè£Ğä¡£çŤĺåŠŇ8.10ářŔèŁĆåŘŇæăůçŽĎæŁĂæIJřïijŇ

ä¡£çŤĺè£ŹäÿłæŰřçŽĎä£őæ■čçL'ĹïijÑä¡ăářśåŔŕäžěåČŔäÿŃéİćè£ŹæăůçijŰåĘŹïijŽ

```
class Point(Structure):
    _fields_ = [
        ('<d', 'x'),
        ('d', 'y')
    ]

class PolyHeader(Structure):
    _fields_ = [
        ('<i', 'file_code'),
        (Point, 'min'), # nested struct
        (Point, 'max'), # nested struct
        ('i', 'num_polys')
    ]
```

äžd'äžžæČŁèőűçŽĎæŸŕïijŇåőČäź§èČ¡æŇL'çĚğécĎæIJ§çŽĎæ■čåÿÿåůěä¡IJïijŇæĹŚäžňåőđéŹĚæŞ■ä¡IJ

```
>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader.from_file(f)
>>> phead.file_code == 0x1234
True
>>> phead.min # Nested structure
<__main__.Point object at 0x1006a48d0>
>>> phead.min.x
0.5
>>> phead.min.y
0.5
>>> phead.max.x
7.0
>>> phead.max.y
9.2
>>> phead.num_polys
3
>>>
```

åĹřçŽőåL’■äÿžæ■ćïijŇäÿĂäÿłåď’ĎçŘĘåőŽéŢ£èőřå¡ŢçŽĎæąĘæđűåůšçżŔåĘŹåě¡äžĘãĂĆä¡ĘæŸŕåęĆæđI
ærŤåęĆïijŇåď’Žè¿źå¡ćæŰĞäžűåŇĚåŘńåŔŸéŢ£çŽĎéČĺåĹĘãĂĆ

äÿĂçğ■æŰźæąĹæŸŕåĘŹäÿĂäÿłçśżæİěèąÍçď’žå■ŮèŁĆæŢřæ■őïijŇåŘŇæŮűåĘŹäÿĂäÿłåůěåĚůåĞ¡æŢřæl

```
class SizedRecord:
    def __init__(self, bytedata):
        self._buffer = memoryview(bytedata)

    @classmethod
    def from_file(cls, f, size_fmt, includes_size=True):
        sz_nbytes = struct.calcsize(size_fmt)
        sz_bytes = f.read(sz_nbytes)
        sz, = struct.unpack(size_fmt, sz_bytes)
        buf = f.read(sz - includes_size * sz_nbytes)
        return cls(buf)

    def iter_as(self, code):
        if isinstance(code, str):
            s = struct.Struct(code)
            for off in range(0, len(self._buffer), s.size):
                yield s.unpack_from(self._buffer, off)
        elif isinstance(code, StructureMeta):
            size = code.struct_size
            for off in range(0, len(self._buffer), size):
                data = self._buffer[off:off+size]
                yield code(data)
```

çśżæŰźæşŢ `SizedRecord.from_file()` æŸŕäÿĂäÿłåůěåĚůïijŇçŤĺæİěäżŐäÿĂäÿłæŰĞäžűäÿ■èŕżå
è£Źäź§æŸŕå¿Ĺåď’ŽæŰĞäžűæăijåijŔåÿÿçŤĺçŽĎæŰźåijŔãĂĆä¡IJäÿžè¿ŞåĚëïijŇåőĆæŐěåŔŮäÿĂäÿłåŇĚåŘńå
åŘŕéĂL’çŽĎ `includes_size` åŔĆæŢřæŇĞåőŽäžĘå■ŮèŁĆæŢřæŸŕåŘęåŇĚåŘńåď’ť éČĺåď’ğářŔãĂĆ
äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘæŢŹä¡äæĂŐæåůä¡£çŤĺäżŐåď’Žè¿źå¡ćæŰĞäžűäÿ■èŕżåŔŰå■ŢçŃńçŽĎåď’Žè¿źå¡ća

```
>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader.from_file(f)
>>> phead.num_polys
3
>>> polydata = [ SizedRecord.from_file(f, '<i')
...              for n in range(phead.num_polys) ]
>>> polydata
[<__main__.SizedRecord object at 0x1006a4d50>,
<__main__.SizedRecord object at 0x1006a4f50>,
<__main__.SizedRecord object at 0x10070da90>]
>>>
```

åŘŕäżěçIJŃåĞžïijŇSizedRecordåőđä¿ŃçŽĎåĘĚåőźè£ŸæšąæIJL'èćńèğčæđŘåĞžæİěãĂĆ
åŘŕäżěä¡£çŤÍiter_as()æŰźæşŢæİěè¿¿åĹřçŻőçŽĎïijŇè£ŹäÿłæŰźæşŢæŐěåŘŮäÿĂäÿłçżŞæđĎæăijåijŔåł
Structureçśżä¡IJäÿžè¿ŞåĚěãĂĆè£ŹæăůåŕŘåŘŕäżěå¿ĹçĄţæť żçŽĎåŐżèğčæđŘæŢřæ■őïijŇä¿ŃåęĆïijŽ

```
>>> for n, poly in enumerate(polydata):
...     print('Polygon', n)
...     for p in poly.iter_as('<dd'):
...         print(p)
...
Polygon 0
(1.0, 2.5)
(3.5, 4.0)
(2.5, 1.5)
Polygon 1
(7.0, 1.2)
(5.1, 3.0)
(0.5, 7.5)
(0.8, 9.0)
Polygon 2
(3.4, 6.3)
(1.2, 0.5)
(4.6, 9.2)
>>>

>>> for n, poly in enumerate(polydata):
...     print('Polygon', n)
...     for p in poly.iter_as(Point):
...         print(p.x, p.y)
...
Polygon 0
1.0 2.5
3.5 4.0
2.5 1.5
Polygon 1
7.0 1.2
5.1 3.0
0.5 7.5
0.8 9.0
```

```
Polygon 2
3.4 6.3
1.2 0.5
4.6 9.2
>>>
```

åřĘæL’ĂæIJL’è£ŹäžŻçżŞåŘĹèţůæİěïijŇäÿŃéİćæŸŕäÿĂäÿł `read_polys()` åĞ¡æŢřçŽĎåŔęåď’ŰäÿĂäÿła£őæ■čçL’ĹïijŽ

```
class Point(Structure):
    _fields_ = [
        ('<d', 'x'),
        ('d', 'y')
    ]

class PolyHeader(Structure):
    _fields_ = [
        ('<i', 'file_code'),
        (Point, 'min'),
        (Point, 'max'),
        ('i', 'num_polys')
    ]

def read_polys(filename):
    polys = []
    with open(filename, 'rb') as f:
        phead = PolyHeader.from_file(f)
        for n in range(phead.num_polys):
            rec = SizedRecord.from_file(f, '<i')
            poly = [ (p.x, p.y) for p in rec.iter_as(Point) ]
            polys.append(poly)
    return polys
```

èőÍèőž

è£ŹäÿĂèŁĆåŘŚä¡ăåśŢçd’žäžĘèőÿåd’ŽéńŸçžğçŽĎçijŰçÍŃæŁĂæIJŕïijŇåŇĚæŃňæŔŔè£ŕåŹÍïijŇåžűè£§è
çĎűèĂŇïijŇåőČäžňéČ¡äÿžäžĘåŘŇäÿĂäÿłçL’źåőŽçŽĎçŽőæăĞæIJ■åŁąãĂĆ

äÿŁéİćçŽĎåőđçŐřçŽĎäÿĂäÿłäÿżèęĄçL’źå¿ĄæŸŕåőČæŸŕå§žäžŐæĞŠèğčåŇĚçŽĎæĂİæČşãĂĆå¡ŞäÿĂä
`Structure` åőđä¿ŃèćńåĹŻåžžæŮűïijŇ `__init__()` äžĚäżĚåŔłæŸŕåĹŻåžžäÿĂäÿłå■ŮèŁĆæŢřæ■őçŽĎå
çL’źåĹńçŽĎïijŇè£ŹæŮűåĂŹåźűæšąæIJL’äżżä¡ŢçŽĎèğčåŇĚæĹŰèĂĚåĚűäżŰäÿŐçżŞæđĎçŻÿåĚşçŽĎæŞ■ä¡
è£ŹæăůåĄŽçŽĎäÿĂäÿłåŁĺæIJžæŸŕä¡ăåŔŕèČ¡äżĚäżĚåŔłåŕźäÿĂäÿłå■ŮèŁĆèőřå¡ŢçŽĎæ§ŘäÿĂåřŔéČĺåĹĘæ

äÿžäžĘåőđçŐřæĞŠèğčåŇĚåŠŇæL’ŞåŇĚïijŇéIJĂèęĄä¡£çŤÍ
`StructField` æŔŔè£ŕåŹÍçźżãĂĆ çŤÍæĹůåIJÍ `_fields_`
äÿ■åĹŮåĞžæİěçŽĎæŕŘäÿłåšđæĂğéČ¡äijŽèćńè¡ňåŇŰæĹŘäÿĂäÿł `StructField`
æŔŔè£ŕåŹÍïijŇ åőČåřĘçŻÿåĚşçżŞæđĎæăijåijŔçăĄåŠŇåĄŘçğżåĂijä£Ìå■ŸåĹřå■ŸåĆÍçijŞå■Ÿäÿ■ãĂĆåĚČçś
`StructureMeta` åIJÍåď’ŽäÿłçżŞæđĎçźèćńåőŽäźL’æŮűèĞłåŁÍåĹŽåžžäžĘè£ŹäžŽæŔŔè£ŕåŹÍãĂĆ
æĹŚäžňä¡£çŤÍåĚČçźżçŽĎäÿĂäÿłäÿżèęĄåŐ§åŽăæŸŕåőČä¡£å¿ŮçŤÍæĹůéİđåÿÿæŰźä¿£çŽĎéĂŽè£ĞäÿĂäÿłér

StructureMeta çŽĎäÿĂäÿłå¿Ĺå¿ő åęŹçŽĎåIJřæŰźářśæŸŕåőČäijŽåŻžåőŽå■ŮèŁĆæŢřæ■őéąžåžŔãĂ
äź§ářśæŸŕèŕťïijŇåęĆæđIJäżżæĎŔçŽĎåśđæĂğæŇĞåőŽäžĘäÿĂäÿłå■ŮèŁĆéąžåžŔ(<èąĺçď žä¡Öä¡■äijŸåĚĹ
æĹŰèĂĚ >èąĺçď žéńŸä¡■äijŸåĚĹ)ïijŇ éĆčåŘŐéİćæĽĂæIJL’å■Ůæőţç ŽĎéąžåžŔéČ¡äżěè£ŹäÿłéąžåžŔäÿžåĞĚ
æŕŤåęĆïijŇä¡ăåŔŕèČ¡æIJL’äÿĂäžŻæŕŤè¿Čåď ■æİĆçŽĎçżŞæđĎïijŇářśåČŔäÿŃéİćè£Źæăůïij Ž

```
class ShapeFile(Structure):
    _fields_ = [ ('>i', 'file_code'), # Big endian
        ('20s', 'unused'),
        ('i', 'file_length'),
        ('<i', 'version'), # Little endian
        ('i', 'shape_type'),
        ('d', 'min_x'),
        ('d', 'min_y'),
        ('d', 'max_x'),
        ('d', 'max_y'),
        ('d', 'min_z'),
        ('d', 'max_z'),
        ('d', 'min_m'),
        ('d', 'max_m') ]
```

äźŃåL’■æĹŚäžňæŔŘåĹřè£ĞïijŇmemoryview() çŽĎä¡£çŤĺåŕäžěåÿőåŁl’æĹŚäžňéĄ£åĚ■åĘĚå■ŸçŽĎ
å¡ŞçżŞæđĎå■ŸåIJĺåţŇåěŮçŽĎæŮűåĂŹïijŇmemoryviews åŕäžěåŔäåŁäåŘŇäÿĂåĘĚå■ŸåŇžå§§äÿŁåőŽä
è£ŹäÿłçL’źæĂğæŕŤè¿Čå¿őåęŹïijŇä¡ĘæŸŕåőČåĚşæşĺçŽĎæŸŕåĘĚå■ŸègĘåŻ¿äÿŐæŹőéĂŽå■ŮèŁĆæŢřçżĎçŽ
åęĆæđIJä¡ăåIJĺäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšæĹŰå■ŮèŁĆæŢřçżĎäÿŁæL’ğèąŇåĹĞçL’ĞæŞ■ä¡IJïijŇä¡äéĂŽåÿÿä
èĂŇåĘĚå■ŸègĘåŻ¿åĹĞçL’Ğäÿ■æŸŕè£ŹæăůçŽĎïijŇåőČäžĚäžĚæŸŕåIJĺåůšå■ŸåIJĺçŽĎåĘĚå■ŸäÿŁéİćåŘäă

è£ŸæIJL’å¿Ĺåď ŽçŻÿåĚşçŽĎçńäèŁĆåŕäžěåÿőåŁl’æĹŚäžňæL’l’åśŢè£ŹéĞŇèőÍèőžçŽĎæŰźæąĹãĂĆ
åŔĆèĂČ8.13åřŔèŁĆä¡£çŤĺæŔŔè£řåŹĺæđĎåžžäÿĂäÿłçśžåđŃçşżçż§ãĂĆ
8.10åřŔèŁĆæIJL’æŻt’åď ŽåĚşäžŐåžűè£§èőąçőŮåśđæĂğåĂijçŽĎèőÍèőžïijŇåźűäÿŤèů§NestedStructæŔŔè£řå
9.19åřŔèŁĆæIJL’äÿĂäÿłä¡£çŤĺåĚČçśżæİěåĹİåğŃåŇŰçśżæĹŘåŚŸçŽĎä¿Ńå■ŘïijŇåŠŇ
StructureMeta çśźéİđåÿÿçŻÿäijijãĂĆ PythonçŽĎ ctypes
æžŘçăĄåŘŇæăůäź§å¿ĹæIJL’èűčïijŇåőČæŔŔä¿ŻäžĘåŕźåőŽäźL’æŢřæ■őçżŞæđĎãĂĄæŢřæ■őçżŞæđĎåţŇåěŮ

6.13 æŢřæ■őçŽĎçt’ŕåŁääÿŐçż§èőąæŞ■ä¡IJ

éŮőécŸ

ä¡ăéIJĂèęĄåď ĎçŘĘäÿĂäÿłå¿Ĺåď ğçŽĎæŢřæ■őéŻĘåźűéIJĂèęĄèőąçőŮæŢřæ■őæĂżåŠŇæĹŰåĚűäžŰçż

èğčåĘşæŰźæąĹ

åŕźäžŐäżżä¡Ţæű L’åŔŁåĹřçż§èőąãĂĄæŮűéŮt’åžŔåĹŮäžěåŔŁåĚűäžŰçŻÿåĚşæŁĂæIJŕçŽĎæŢřæ■őåĹĘ
PandasåžŞ ãĂĆ

äÿžäžĘèőľä¡ăåĚĹä¡ŞéłŇäÿŃïijŇäÿŃéİćæŸŕäÿĂäÿłä¡£çŤĺPandasæİěåĹĘæđŘèŁİåŁăåŞěå§ŐåÿĆçŽĎ
èĂĄéijăåŠŇåŢőé¡£çśżåŁĺçL’l’æŢřæ■őåžŞ çŽĎä¿Ńå■ŘãĂĆ åIJĺæĹŚåĘŹè£ŹçŕĞæŰĞçńäçŽĎæŮűåĂŹïijŇè£Ź

```
>>> import pandas

>>> # Read a CSV file, skipping last line
>>> rats = pandas.read_csv('rats.csv', skip_footer=1)
>>> rats
<class 'pandas.core.frame.DataFrame'>
Int64Index: 74055 entries, 0 to 74054
Data columns:
Creation Date 74055 non-null values
Status 74055 non-null values
Completion Date 72154 non-null values
Service Request Number 74055 non-null values
Type of Service Request 74055 non-null values
Number of Premises Baited 65804 non-null values
Number of Premises with Garbage 65600 non-null values
Number of Premises with Rats 65752 non-null values
Current Activity 66041 non-null values
Most Recent Action 66023 non-null values
Street Address 74055 non-null values
ZIP Code 73584 non-null values
X Coordinate 74043 non-null values
Y Coordinate 74043 non-null values
Ward 74044 non-null values
Police District 74044 non-null values
Community Area 74044 non-null values
Latitude 74043 non-null values
Longitude 74043 non-null values
Location 74043 non-null values
dtypes: float64(11), object(9)

>>> # Investigate range of values for a certain field
>>> rats['Current Activity'].unique()
array([nan, Dispatch Crew, Request Sanitation Inspector],␣
 ↪dtype=object)
>>> # Filter the data
>>> crew_dispatched = rats[rats['Current Activity'] == 'Dispatch␣
 ↪Crew']
>>> len(crew_dispatched)
65676
>>>

>>> # Find 10 most rat-infested ZIP codes in Chicago
>>> crew_dispatched['ZIP Code'].value_counts()[:10]
60647 3837
60618 3530
60614 3284
60629 3251
60636 2801
60657 2465
60641 2238
```

```
60609 2206
60651 2152
60632 2071
>>>

>>> # Group by completion date
>>> dates = crew_dispatched.groupby('Completion Date')
<pandas.core.groupby.DataFrameGroupBy object at 0x10d0a2a10>
>>> len(dates)
472
>>>

>>> # Determine counts on each day
>>> date_counts = dates.size()
>>> date_counts[0:10]
Completion Date
01/03/2011 4
01/03/2012 125
01/04/2011 54
01/04/2012 38
01/05/2011 78
01/05/2012 100
01/06/2011 100
01/06/2012 58
01/07/2011 1
01/09/2012 12
>>>

>>> # Sort the counts
>>> date_counts.sort()
>>> date_counts[-10:]
Completion Date
10/12/2012 313
10/21/2011 314
09/20/2011 316
10/26/2011 319
02/22/2011 325
10/26/2012 333
03/17/2011 336
10/13/2011 378
10/14/2011 391
10/07/2011 457
>>>
```

åŮřïijŇçIJŃæăůå■Ř2011åźť10æIJĹ7æŮěářźèĂĄéijääżňæİěèřť'æŸŕäÿłå¿Ĺå£ŹçćŇçŽĎæŮěå■ŘåŢŁïijĄ

èőÍèőž

PandasæŸŕäÿĂäÿłæŃěæIJL'å¿Ĺåď'ŽçL'źæĂğçŽĎåď'ğåđŃåĞ¡æŢřåžŞïijŇæĹŠåIJÍè£ŹéĞŇäÿ■åŔŕèČ¡äżŇ
ä¡ĘæŸŕåŔłèęĄä¡ăéIJĂèęĄåŐżåĹĘæđŘåď'ğåđŃæŢřæ■őéŽĘåŘĹãĂĄåŕźæŢřæ■őåĹĘçżĎãĂĄèőąçőŮåŘĎçğ■

çňňäÿČçńăïijŽåĞ¡æŢř

ä¡£çŤÍ def èŕ■åŘěåőŽäźL'åĞ¡æŢřæŸŕæL'ĂæIJL'çÍŃåžŘçŽĎå§žçąĂãĂĆ
æIJňçńăçŽĎçŻőæăĞæŸŕèőšèğčäÿĂäžŻæŻť'åŁăéńŸçžğåŠŇäÿ■åÿÿèğĄçŽĎåĞ¡æŢřåőŽäźL'äÿŐä¡£çŤÍæĺąåijŘ
æűL'åŘŁåĹřçŽĎåĘĚåőźåŇĚæŃňéżŸèőď'åŔĆæŢřãĂĄäżżæĎŔæŢřéĞŔåŔĆæŢřãĂĄåijžåĹűåĚşéŤőå■ŮåŔĆ
åŘęåď'ŰïijŇäÿĂäžŽéńŸçžğçŽĎæŐğåĹűæţĄåŠŇåĹl'çŤÍåŻđèřČåĞ¡æŢřäijăéĂŠæŢřæ■őçŽĎæŁĂæIJŕåIJÍè£Ź

Contents:

7.1 åŘŕæŐěåŘŮäżżæĎŔæŢřéĞŔåŔĆæŢřçŽĎåĞ¡æŢř

éŮőécŸ

ä¡ăæČşæđĎéĂääÿĂäÿłåŘŕæŐěåŘŮäżżæĎŔæŢřéĞŔåŔĆæŢřçŽĎåĞ¡æŢřãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘèČ¡èől'äÿĂäÿłåĞ¡æŢřæŐěåŘŮäżżæĎŔæŢřéĞŔçŽĎä¡■ç¡őåŔĆæŢřijŇåŘŕäžěä¡£çŤÍäÿĂäÿł*åŔĆ

```
def avg(first, *rest):
    return (first + sum(rest)) / (1 + len(rest))

# Sample use
avg(1, 2) # 1.5
avg(1, 2, 3, 4) # 2.5
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijÑrestæŸŕçŤśæL'ĂæIJL'åĚűäžŰä¡■ç¡őåŔĆæŢřçżĎæĹŘçŽĎåĚČçżĎãĂĆçĎűåŘĆ

äÿžäžĘæŐěåŘŮäżżæĎŔæŢřéĞŔçŽĎåĚşéŤőå■ŮåŔĆæŢřijŇä¡£çŤÍäÿĂäÿłäžě**åijĂåď't'çŽĎåŔĆæŢřã

```
import html

def make_element(name, value, **attrs):
    keyvals = [' %s="%s"' % item for item in attrs.items()]
    attr_str = ''.join(keyvals)
    element = '<{name}{attrs}>{value}</{name}>'.format(
                name=name,
                attrs=attr_str,
                value=html.escape(value))
    return element
```

```
# Example
# Creates '<item size="large" quantity="6">Albatross</item>'
make_element('item', 'Albatross', size='large', quantity=6)

# Creates '<p>&lt;spam&gt;</p>'
make_element('p', '<spam>')
```

åIJÍè£ŹéĞÑïijÑattrsæŸŕäÿĂäÿłåÑĚåŘńæĽĂæIJL'èćńäijååĚěè£ŻæİěçŽĎåĚşéŤőå■ŮåŘĆæŢřçŽĎå■ŮåĚ

åęĆædIJä¡ăè£ŸåÿÑæIJŻæ§ŘäÿłåĞ¡æŢřèČ¡åŘÑæŮűæŐěåŔŮäżżæĎŔæŢřéĞŘçŽĎä¡■ç¡őåŘĆæŢřåŠÑål

```
def anyargs(*args, **kwargs):
    print(args) # A tuple
    print(kwargs) # A dict
```

ä¡£çŤÍè£ŹäÿłåĞ¡æŢřæŮűïijÑæĽĂæIJL'ä¡■ç¡őåŘĆæŢřäijŽèćńæŤ¿åĹřargsåĚČçżĎäÿ■ïijÑæĽĂæIJL'åĚş

èőÍèőž

äÿĂäÿł*åŘĆæŢřåŘłèČ¡åĞžçŐřåIJÍåĞ¡æŢřåőŽäźL'äÿ■æIJĂåŘŐäÿĂäÿłä¡■ç¡őåŘĆæŢřåŘŐéİćïijÑèĂÑ
**åŘĆæŢřåŘłèČ¡åĞžçŐřåIJÍæIJĂåŘŐäÿĂäÿłåŘĆæŢřãĂĆ æIJL'äÿĂçĆźèęĄæşÍæĎŔçŽĎæŸŕïijÑåIJÍ*åŘĆæ

```
def a(x, *args, y):
    pass

def b(x, *args, y, **kwargs):
    pass
```

è£Źçğ■åŘĆæŢřåřśæŸŕæĹŚäžňæĽĂèŕť'çŽĎåijžåĹűåĚşéŤőå■ŮåŘĆæŢřïijÑåIJÍåŘŐéİć7.2åřŘèŁĆè£Ÿäij

7.2 åŘłæŐěåŘŮåĚşéŤőå■ŮåŘĆæŢřçŽĎåĞ¡æŢř

éŮőécŸ

ä¡ăåÿÑæIJŻåĞ¡æŢřçŽĎæ§ŘäžŻåŘĆæŢřåijžåĹűä¡£çŤÍåĚşéŤőå■ŮåŘĆæŢřäijäéĂŠ

èğčåĘşæŰźæąĹ

åřĘåijžåĹűåĚşéŤőå■ŮåŘĆæŢřæŤ¿åĹřæ§Řäÿł*åŘĆæŢřæĹŰèĂĚå■Ţäÿł*åŘŐéİcåřśèČ¡è¿¿åĹřè£Źçğ■æŢ

```
def recv(maxsize, *, block):
    'Receives a message'
    pass

recv(1024, True) # TypeError
recv(1024, block=True) # Ok
```

åĹľçŤĺè£Źçğ■æŁĂæIJŕïijŇæĹŚäžňè£ŸèČ¡åIJÍæŐěåŔŮäżżæĎŔåď˙Žäÿłä¡■ç¡őåŔĆæŢřçŽĎåĞ¡æŢřäÿ■æ

```
def mininum(*values, clip=None):
    m = min(values)
    if clip is not None:
        m = clip if clip > m else m
    return m

minimum(1, 5, 2, -5, 10) # Returns -5
minimum(1, 5, 2, -5, 10, clip=0) # Returns 0
```

èőÍèőž

å¿Ĺåď˙ŽæČĚåĘţäÿŃïijÑä¡£çŤĺåijžåĹűåĚşéŤőå■ŮåŔĆæŢřäijŽæŕŤä¡£çŤĺä¡■ç¡őåŔĆæŢřèąÍæĎŔæŻť˙åŁă
ä¿ŃåęĆïijÑèĂČèŹŚäÿŃåęĆäÿŃäÿĂäÿłåĞ¡æŢřèřČçŤĺïijŽ

```
msg = recv(1024, False)
```

åęĆæđIJèřČçŤĺèĂĚårźrecvåĞ¡æŢřåźűäÿ■æŸŕå¿ĹçĘ§æĆL˙ïijŇéĆčäžŰèĆŕåőŽäÿ■æŸŐçŹ¡éĆčäÿłFalseå
ä¡ĘæŸŕïijŇåęĆæđIJäžčçăĄåŔŸæĹŘäÿŃéİćè£Źæăůå■ŘçŽĎèŕÍåřśæÿĚæěŽåď˙ŽäžĘïijŽ

```
msg = recv(1024, block=False)
```

åŔęåď˙ŰïijÑä¡£çŤĺåijžåĹűåĚşéŤőå■ŮåŔĆæŢřäź§äijŽæŕŤä¡£çŤĺ**kwargsåŔĆæŢřæŻť˙åě¡ïijÑåŻääÿžåIJ

```
>>> help(recv)
Help on function recv in module __main__:
recv(maxsize, *, block)
    Receives a message
```

åijžåĹűåĚşéŤőå■ŮåŔĆæŢřåIJÍäÿĂäžŽæŻť˙éńŸçžğåIJžåŘĹåŘŇæăůäź§å¿ĹæIJL˙çŤĺãĆĆ
ä¿ŃåęĆïijÑåőČäžňåŔŕäžěèćńçŤĺæİěåIJÍä¡£çŤĺ*argsåŠŇ**kwargsåŔĆæŢřä¡IJäÿžè¿ŞåĚěçŽĎåĞ¡æŢřäÿ■æŔŠ

7.3 çżŹåĞ¡æŢřåŔĆæŢřåćđåŁääĚČä£ąæĄŕ

éŮőécŸ

ä¡ääĘŹåě¡äžĘäÿĂäÿłåĞ¡æŢřïijÑçĎűåŘŐæČşäÿžè£ŹäÿłåĞ¡æŢřçŽĎåŔĆæŢřåćđåŁääÿĂäžŽécİåď˙ŰçŽĎ

èğčåĘşæŰźæąĹ

ä¡£çŤĺåĞ¡æŢřåŔĆæŢřæşĺèğčæŸŕäÿĂäÿłå¿Ĺåě¡çŽĎåŁđæşŢïijÑåőČèČ¡æŘŘçď˙žçÍÑåžŘåŚŸåžŤèŕěæĂŐ
ä¿ŃåęĆïijÑäÿŃéİćæIJL˙äÿĂäÿłècńæşĺèğčäžĘçŽĎåĞ¡æŢřïijŽ

```
def add(x:int, y:int) -> int:
    return x + y
```

pythonèğčéĞŁåŹÍäÿ■äijŽåŕźè£ŹäžŻæşÍèğčæůżåŁääżżä¡ŢçŽĎèŕ■äźL'ãĂĆåőČäžňäÿ■äijŽèćńçśżåđŃæčĂçĎűěĂŇïijŇåŕźäžŐéĆčäžŻéŸĚèŕżæžŘçăĄçŽĎäžžæİěèőšåřšå¿ŁæIJL'åÿőåŁl'åŢęãĂĆçňňäÿL'æŰźåůěåĚůåŠŇ

```
>>> help(add)
Help on function add in module __main__:
add(x: int, y: int) -> int
>>>
```

åř¡çőąä¡ăåŔŕäžěä¡£çŤÍäżżæĎŘçśżåđŃçŽĎåŕźèśąçżŹåĞ¡æŢřæůżåŁäæşÍèğč(ä¿ŃåęĆæŢřå■ŮïijŇå■Ůçňęä

èőÍèőž

åĞ¡æŢřæşÍèğčåŔłå■ŸåĆÍåIJÍåĞ¡æŢřçŽĎ __annotations__ åśđæĂğäÿ■ãĂĆä¿ŃåęĆïijŽ

```
>>> add.__annotations__
{'y': <class 'int'>, 'return': <class 'int'>, 'x': <class 'int'>}
```

åř¡çőąæşÍèğčçŽĎä¡£çŤÍæŰźæşŢåŔŕèČ¡æIJL'å¿Ĺåd'Žçğ■ïijŇä¡ĘæŸŕåőČäžňçŽĎäÿżèęĄçŤÍéĂŤè£ŸæŸŕ
åŽääÿžpythonåźűæšąæIJL'çśżåđŃåčřæŸŐïijŇéĂŽåÿÿæİěèőšäžĚäžĚéĂŽè£ĞéŸĚèŕżæžŘçăĄå¿ĹéŽ¿ç§ééĄŞå
è£ŹæŮűåĂŹä¡£çŤÍæşÍèğčåŕśèČ¡çżŹçÍŃåžŔåŚŸæŻt'åd'ŽçŽĎæŔŘçd'žïijŇèől'äžŰäžňåŔŕäžěæ■čçąőçŽĎä¡£ç

åŔĆèĂČ9.20åřŔèŁĆçŽĎäÿĂäÿłæŻt'åŁäéńŸçžğçŽĎä¿Ńå■ŘïijŇæijŤçd'žäžĘåęĆä¡ŢåĹl'çŤÍæşÍèğčæİěåć

7.4 è£ŤåŻđåd'Žäÿłå Äijç ŽĎåĞ¡æŢř

éŮőécŸ

ä¡ăåÿŇæIJŻæđĎéĂääÿĂäÿłåŔŕäžěè£ŤåŻđåd'Žäÿłå ÄijçŽĎåĞ¡æŢř

èğčåĘşæŰźæąĹ

äÿžäžĘèČ¡è£ŤåŻđåd'Žäÿłå ÄijïijŇåĞ¡æŢřçŻt'æŐěreturnäÿĂäÿłåĚČçżĎåŕśèąŇäžĘãĂĆä¿ŃåęĆïijŽ

```
>>> def myfun():
...     return 1, 2, 3
...
>>> a, b, c = myfun()
>>> a
1
>>> b
2
>>> c
3
```

èőÍèőž

åř¡çőąmyfun()çIJŃäÿŁåŐżè£ŤåŻđäžĘåď'ŽäÿłåĭijïijŇåőđéŹĚäÿŁæŸŕåĚĹåĹŻåżžäžĘäÿĂäÿłåĚČçżĎçĎ
è£ŹäÿłèíæşŢçIJŃäÿŁåŐżærŕŤè¿ČåěĞæĂłïijŇåőđéŹĚäÿŁæĹŚäžňä¡£çŤĺçŽĎæŸŕéĂŮåŔůæİěçŤ§æĹŘäÿĂäÿ

```
>>> a = (1, 2) # With parentheses
>>> a
(1, 2)
>>> b = 1, 2 # Without parentheses
>>> b
(1, 2)
>>>
```

å¡ŞæĹŚäžňèřČçŤĺè£ŤåŻđäÿĂäÿłåĚČçżĎçŽĎåĞ¡æŢřçŽĎæŮűåĂŹ
ïijŇéĂŽåÿÿæĹŚäžňäijŽářĘçżŞæđIJètŃåĂijçżŹåď'ŽäÿłåŕŷéĞŕïijŇářšåČŕäÿŁéİçŽĎéĆčæăůãĂĆ
åĚűåőđè£Źářśæŷŕ1.1ářŕèŁĆäÿ■æĹŚäžňæL'Ăèŕť'çŽĎåĚČçżĎèğčåŇĚãĂĆè£ŤåŻđçżŞæđIJäź§åŕŕäžěètŃåĂij
è£ŹæŮűåĂŹè£ŹäÿłåŕŷéĞŕåĂijářśæŸŕåĞ¡æŢřè£ŤåŻđçŽĎéĆčäÿłåĚČçżĎæIJňèžňäžĘïijŽ

```
>>> x = myfun()
>>> x
(1, 2, 3)
>>>
```

7.5 åőŽäźĽælJL'éżŸèőď'åŔĆæŢřçŽĎåĞ¡æŢř

éŮőécŸ

ä¡ăæČşåőŽäźL'äÿĂäÿłåĞ¡æŢřæĹŰèĂĚæŰźæşŢïijŇåőČçŽĎäÿĂäÿłæĹŰåď'Žäÿłåŕ ĆæŢřæŸŕåŕŕéĂL'çŽ

èğčåĘşæŰźæąĹ

åőŽäźL'äÿĂäÿłæIJL'åŕŕéĂL'åŔĆæŢřçŽĎåĞ¡æŢřæŸŕéİđåÿÿçőĂå■ŢçŽĎïijŇçŻt'æŐěåIJÍåĞ¡æŢřåőŽäźL

```
def spam(a, b=42):
    print(a, b)

spam(1) # Ok. a=1, b=42
spam(1, 2) # Ok. a=1, b=2
```

åęĆæđIJéżŸèőď'åŔĆæŢřæŸŕäÿĂäÿłåŕŕ£őæŤźçŽĎåőźåŹĺæŕŤåęĆäÿĂäÿłåĹŮèąĺãĂĄéŽĘåŘĹæĹŰèĂĚ

```
# Using a list as a default value
def spam(a, b=None):
    if b is None:
        b = []
    ...
```

åęĆædIJä¡ăåźűäÿ■æČşæŔŘä¿ŻäÿĂäÿłéżŸèőď åĂijïijŇèĂŇæŸŕæČşäżĔäžĔæţŃèŕŢäÿŃæ§ŘäÿłéżŸèőď å

```
_no_value = object()

def spam(a, b=_no_value):
    if b is _no_value:
        print('No b value supplied')
    ...
```

æĹŚäžňæţŃèŕŢäÿŃè£ŹäÿłåĞ¡æŢřïijŽ

```
>>> spam(1)
No b value supplied
>>> spam(1, 2) # b = 2
>>> spam(1, None) # b = None
>>>
```

äžŤçżĘèğĆåŕ§åŘŕäžěåŘŚçŐřåĹřäijăéĂŠäÿĂäÿłNoneåĂijåŠŇäÿ■äijăåĂijäÿd’çğ■æČĚåĘţæŸŕæIJL’åůőå

èőÍèőž

åőŽäźL’åÿęéżŸèőď åĂijåŘĆæŢřçŽĎåĞ¡æŢřæŸŕå¿ĹçőĂå■ŢçŽĎïijŇä¡EçżÍäÿ■äžĔäžĔåŘłæŸŕè£ŹäÿłïijŇ

éęŰåĚĹïijŇéżŸèőď åŘĆæŢřçŽĎåĂijäžĔäžĔåIJÍåĞ¡æŢřåőŽäźL’çŽĎæŮűåĂŹèţŃåĂijäÿĂæňąãĂĆèŕŢçÍ

```
>>> x = 42
>>> def spam(a, b=x):
...     print(a, b)
...
>>> spam(1)
1 42
>>> x = 23 # Has no effect
>>> spam(1)
1 42
>>>
```

æşÍæĎŘåĹřå¡ŞæĹŚäžňæŤźåŘŸxçŽĎåĂijçŽĎæŮűåĂŹåŕźéżŸèőď åŘĆæŢřåĂijåźűæšąæIJL’å¡śåŞ■ïijŇè

åĚűæňąïijŇéżŸèőď åŘĆæŢřçŽĎåĂijåžŤèŕěæŸŕäÿ■åŘŕåŘŸçŽĎåŕźèśąïijŇæŕŤåęĆNoneãĂĄTrueãĂĄFal
çL’źåĹńçŽĎïijŇå■ČäÿĞäÿ■èęĄåČŘäÿŃéİćè£ŹæăůåĘŹäžčçăĄïijŽ

```
def spam(a, b=[]): # NO!
    ...
```

åęĆædIJä¡ăè£ŹäźĹåĄŽäžĘïijŇå¡ŞéżŸèőď åĂijåIJÍåĚűäžŰåIJřæŰźèćńä£őæŤźåŘŐä¡ăářĘäijŽéĄĞåĹřåŘ

```
>>> def spam(a, b=[]):
...     print(b)
...     return b
...
>>> x = spam(1)
```

```
>>> x
[]
>>> x.append(99)
>>> x.append('Yow!')
>>> x
[99, 'Yow!']
>>> spam(1) # Modified list gets returned!
[99, 'Yow!']
>>>
```

è£Źçğ■çżŞæđIJåžŤèŕěäÿ■æŸŕä¡ăæČşèęĄçŽĎãĂĆäÿžäžĘéĄ£åĚ■è£Źçğ■æČĚåĘţçŽĎåŔŞçŤ§ïijŇæIJĂå
çĎűåŘŐåIJĺåĞ¡æŢŕéĞŇéİćæčĂæ§ěåőČïijŇåĽ■éİćçŽĎä¿Ńå■ŘåŕśæŸŕè£Źæăůåą ŽçŽĎãĂĆ

åIJĺæţŃèŕŢNoneåĂijæŮűä¡£çŤĺ is æŞ■ä¡IJçňęæŸŕå¿ĹéĞ■èęĄçŽĎïijŇäź§æŸŕè£Źçğ■æŰźæąĹçŽĎåĚş
æIJĽæŮűåĂŹåď ğåőűäijŽçŁŕäÿŃäÿŃéİćè£Źæăůç ŽĎéŤŹèŕŕïijŽ

```
def spam(a, b=None):
    if not b: # NO! Use 'b is None' instead
        b = []
    ...
```

è£ŹäźĹåĘŹçŽĎéŮőécŸåIJĺäžŐår¡çőąNoneåĂijçąőåőđæŸŕècńå¡ŞæĹŘFalseïijŇ
ä¡ĘæŸŕè£ŸæIJĽåĚűäžŰçŽĎårźèšą(æŕŤåęĆéŢ£åžęäÿž0çŽĎå■Ůçňęäÿšã ĂĄåĹŮèąĺãĂĄåĚĈçżĎãĂĄå■ŮåĚÿ
åŻăæ■ď ïijŇäÿŁéİćçŽĎäžčçăĄäijŽèŕŕářĘäÿĂäžŻåĚűäžŰè¿ŞåĚěäź§å¡ŞæĹŘæŸŕæšąæIJĽè¿ŞåĚěãĆćŕŤåęĆ

```
>>> spam(1) # OK
>>> x = []
>>> spam(1, x) # Silent error. x value overwritten by default
>>> spam(1, 0) # Silent error. 0 ignored
>>> spam(1, '') # Silent error. '' ignored
>>>
```

æIJĂåŘŐäÿĂäÿłéŮőécŸæŕŤè¿Čå¿őåęŹïijŇéĆčářśæŸŕäÿĂäÿłåĞ¡æŢŕéIJĂèęĄæţŃèŕŢæ§ŘäÿłåŔŕéĂĽ'åŘ
è£ŹæŮűåĂŹéIJĂèęĄářŘå£ĈçŽĎæŸŕä¡ääÿ■èČ¡çŤĺæ§ŘäÿłéžŸèőď åĂijæŕŤåęĆNoneãĂĄ
0æĹŰèĂĚFalseåĂijæİěæţŃèŕŢçŤĺæĹůæŔŘä¿ŻçŽĎåĂij(åŽăäÿžè£ŹäźŽåĂijéČ¡æŸŕåŘĹæşŢçŽĎåĂijïijŇæŸ
åŻăæ■ď ïijŇä¡ăéIJĂèęĄåĚűäžŰçŽĎèğčåĘşæŰźæąĹäźĘãĂĆ

äÿžäźĘèğčåĘşè£ŹäÿłéŮőécŸïijŇä¡ăåŘŕäžěåĹŽåžžäÿĂäÿłçŃňäÿĂæŮăäžŇçŽĎçğĄæIJĽåŕźèśąåőđä¿Ńïij
åIJĺåĞ¡æŢŕéĞŇéİćïijŇä¡ăåŘŕäžěéĂŽè£ĞæčĂæ§ěèćńäijăéĂŠåŔĆæŢřåĂijèů§è£Źäÿłåőđä¿ŃæŸŕåĘäÿĂæăůå
è£ŹéĞŇçŽĎæİèůŕæŸŕçŤĺæĹůäÿ■åŘŕèČ¡åÖżäijăéĂŠè£Źäÿł_no_valueåőđä¿Ńä¡IJäÿžè¿ŞåĚěãĆ
åŻăæ■ď ïijŇè£ŹéĞŇéĂŽè£ĞæčĂæ§ěè£ŹäÿłåĂijåŕśèČ¡çąőåőŽæ§ŘäÿłåŔĆæŢřæŸŕåĘèćńäijăéĂŠè£ŽæİěäžĘ

è£ŹéĞŇåŕź `object()` çŽĎä¡£çŤĺçIJŃäÿŁåÖżæIJĽçĆźäÿ■åď łåÿÿèğĄãĂĆobject
æŸŕpythonäÿ■æĽĂæIJĽçśżçŽĎå§žçśżãĂĆ ä¡ăåŘŕäžěåĹŽåžž object
çśżçŽĎåőđä¿ŃïijŇä¡ĘæŸŕè£ŹäźŽåőđä¿ŃæšąäźĂäźĹåőđéŹĚçŤĺåď ĎïijŇåŽăäÿžåőČåźűæšąæIJĽäżżä¡ŢæIJĽ
äź§æšąæIJĽäżżä¡Ţåőđä¿ŃæŢřæ■ő(åŽăäÿžåőČæšąæIJĽäżżä¡ŢçŽĎåőđä¿Ńå■ŮåĚÿïijŇä¡ăçŤŽèĞşéČ¡äÿ■èČ¡ę
ä¡ăåŤŕäÿĂèČ¡åĄŽçŽĎåŕśæŸŕæţŃèŕŢåŘŇäÿĂæĂğãĆĆè£ŹäÿłåĹŽåě¡çňęåŘĹæĹŚçŽĎèęĄæśĆïijŇåŽăäÿžæĹ

7.6 åőŽäźĽåŇ£åŘ■æĹŰåĘĚèĄŤåĞ¡æŢř

éŮőécŸ

ä¡ăæČşäÿž `sort()` æŞ■ä¡IJåĹŻåžžäÿĂäÿłå¿Łç§■çŽĎåŻđèřČåĞ¡æŢřïijŇä¡ĘåŔĹäÿ■æČşçŢĺ
`def` åŐżåĘŹäÿĂäÿłå■ŢèąŇåĞ¡æŢřïijŇ èĂŇæŸŕåÿŇæIJŻéĂŽè£Ğæ§Řäÿłå£ńæ■ůæŰźåijŔäžěåĘĚèĄŤæŰźåi

èğčåĘşæŰźæąĹ

å¡ŞäÿĂäžŽåĞ¡æŢřå¿ŁçőĂå■ŢïijŇäżĚäżĚåŔłæŸŕèőąçőŮäÿĂäÿłèąĺè¿¿åijŔçŽĎåĂijçŽĎæŮűåĂŹïijŇåřś

```
>>> add = lambda x, y: x + y
>>> add(2,3)
5
>>> add('hello', 'world')
'helloworld'
>>>
```

è£ŹéĞŇä¡£çŤĺçŽĎlambdaèąĺè¿¿åijŔèů§äÿŃéİćçŽĎæŢĹæđIJæŸŕäÿĂæăůçŽĎïijŽ

```
>>> def add(x, y):
...     return x + y
...
>>> add(2,3)
5
>>>
```

lambdaèąĺè¿¿åijŔåĚÿåđŃçŽĎä¡£çŤĺåIJžæŹŕæŸŕæŐŠåžŔæĹŰæŢřæ■őreduceç■L'ïijŽ

```
>>> names = ['David Beazley', 'Brian Jones',
...          'Raymond Hettinger', 'Ned Batchelder']
>>> sorted(names, key=lambda name: name.split()[-1].lower())
['Ned Batchelder', 'David Beazley', 'Raymond Hettinger', 'Brian␣
 ↪Jones']
>>>
```

èőĺèőž

åř¡çőąlambdaèąĺè¿¿åijŔåĚĄèőÿä¡ăåőŽäźL'çőĂå■ŢåĞ¡æŢřïijŇä¡ĘæŸŕåőČçŽĎä¡£çŤĺæŸŕæIJL'éŹŘåĹűç
ä¡ăåŔłèČ¡æŇĞåőŽå■Ţäÿłèąĺè¿¿åijŔïijŇåőČçŽĎåĂijåřśæŸŕæIJĂåŘŐçŽĎè£ŤåŻđåĂijãĂĆäź§åřśæŸŕèŕt'äÿ■è
åŇĚæŇňåd'Žäÿłèŕ■åŔěãĂĄæİąäžűèąĺè¿¿åijŔãĂĄè£■äžčäžěåŔŁåijĆåÿÿåd'ĎçŘĘç■L'ç■L'ãĂĆ

ä¡ăåŔŕäžěäÿ■ä¡£çŤĺlambdaèąĺè¿¿åijŔåřśèČ¡çijŰåĘŹåd'ğéČĺåĹĘpythonäžčçăĄãĂĆ
ä¡ĘæŸŕïijŇå¡ŞæIJL'äžžçijŰåĘŹåd'ğéĞŔèőąçőŮèąĺè¿¿åijŔåĂijçŽĎç§■åřŔåĞ¡æŢřæĹŰèĂĚéIJĂèęĄçŤĺæĹůa
ä¡ăåřśäijŽçIJŃåĹřlambdaèąĺè¿¿åijŔçŽĎèžńå¡śäžĘãĂĆ

7.7 åŇ£åŘ■åĞ¡æŢřæ■ŢèŐůåŔŸéĞŔåĂij

éŮőécŸ

ä¡ăçŤÍlambdaåőŽäźL’äžĘäÿĂäÿłåŇ£åŘ■åĞ¡æŢřïijŇåźűæČşåIJÍåőŽäźL’æŮűæ■ŢèŐůåĹřæ§ŘäžŽåŔŸéĞ

èğčåĘşæŰźæąĹ

åĚĹçIJŃäÿŃäÿŃéİćäżčçăĄçŽĎæŢĹæđIJïijŽ

```
>>> x = 10
>>> a = lambda y: x + y
>>> x = 20
>>> b = lambda y: x + y
>>>
```

çŐřåIJÍæĹŚéŮőä¡ăïijŇa(10)åŠŇb(10)è£ŤåŻđçŽĎçżŞæđIJæŸŕäżĂäźĹïij§åęĆæđIJä¡äèőd’äÿžçżŞæđIJæŸ

```
>>> a(10)
30
>>> b(10)
30
>>>
```

è£ŹåĚűäÿ■çŽĎåěěåęŹåIJÍäžŐlambdaèąÍè¿¿åijŔäÿ■çŽĎxæŸŕäÿĂäÿłèĞłçŤśåŔŸéĞŔïijŇ
åIJÍè£ŘèąŇæŮűçżŚåőŽåĂijïijŇèĂŇäÿ■æŸŕåőŽäźL’æŮűåřśçżŚåőŽïijŇè£Źèů§åĞ¡æŢřçŽĎéżŸèőd’åĂijåŔĆa
åŻăæ■d’ïijŇåIJÍèřĈçŤÍè£ŹäÿłlambdaèąÍè¿¿åijŔçŽĎæŮűåĂŹïijŇxçŽĎåĂijæŸŕæL’ğèąŇæŮűçŽĎåĂijãĂĆä¿Ň

```
>>> x = 15
>>> a(10)
25
>>> x = 3
>>> a(10)
13
>>>
```

åęĆæđIJä¡ăæČşèől’æ§ŘäÿłåŇ£åŘ■åĞ¡æŢřåIJÍåőŽäźL’æŮűåřśæ■ŢèŐůåĹřåĂijïijŇåŔŕäżěářĘéĆčäÿłåŔĆ

```
>>> x = 10
>>> a = lambda y, x=x: x + y
>>> x = 20
>>> b = lambda y, x=x: x + y
>>> a(10)
20
>>> b(10)
30
>>>
```

èőÍèőž

åIJÍè£ŹéĞŇåĹŮåĞžæÌěçŽĎéŮőécŸæŸŕæŰřæĽŃå¿Ĺåőźæ ŸŞçŁŕçŽĎéŤŹèŕŕïijŇæIJĽäžŻæŰřæĽŃåŔŕè æŕŤåęĆïijŇéĂŽè£ĞåIJÍäÿĂäÿłå¿łçŐŕæĹŰåĹŮèąÍæŐÍåŕijäÿ■åĹŻåżžäÿĂäÿłlambdaèąÍè¿¿åijŔåĹŮèąÍïijŇåźűa

```
>>> funcs = [lambda x: x+n for n in range(5)]
>>> for f in funcs:
... print(f(0))
...
4
4
4
4
4
>>>
```

ä¡ĘæŸŕåőđéŹĚæŢĹæđIJæŸŕè£ŘèąŇæŸŕnçŽĎåĂijäÿžè£■äżčçŽĎæIJĂåŘŐäÿĂäÿłåĂijãĂĆçŐřåIJÍæĹŚä

```
>>> funcs = [lambda x, n=n: x+n for n in range(5)]
>>> for f in funcs:
... print(f(0))
...
0
1
2
3
4
>>>
```

éĂŽè£Ğä¡£çŤÍåĞ¡æŢřéżŸèőd'åĂijåŔĆæŢřå¡ćåijŔïijŇlambdaåĞ¡æŢřåIJÍåőŽäźL'æŮűåřśèČ¡çżŚåőŽåĹřå

7.8 åĞŔåřŚåŔŕèřČçŤÍåŕźèśąçŽĎåŔĆæŢřäÿłæŢř

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłèćńåĚűäżŰpythonäżčçăĄä¡£çŤÍçŽĎcallableåŕźèśąïijŇåŔŕèČ¡æŸŕäÿĂäÿłåŻđèřČåĞ¡æŢřa ä¡ĘæŸŕåőČçŽĎåŔĆæŢřåd'łåd'ŽäžĘïijŇåŕijèĞt'èřČçŤÍæŮűåĞžéŤŹãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJéIJĂèęĄåĞŔåřŚæ§ŘäÿłåĞ¡æŢřçŽĎåŔĆæŢřäÿłæŢřïijŇä¡ăåŔŕäżěä¡£çŤÍ `functools.partial()` ãĂĆ `partial()` åĞ¡æŢřåĚĄèőÿä¡ăçżŹäÿĂäÿłæĹŰåd'ŽäÿłåŔĆæŢřèő¿ç¡őåŻža äÿžäžĘæijŤçd'žæÿĚæěŽïijŇåĄĞèő¿ä¡ăæIJL'äÿŃéÍcè£ŹæăůçŽĎåĞ¡æŢřijŽ

```
def spam(a, b, c, d):
    print(a, b, c, d)
```

çŐřåIJÍæĹŚäžňä¡£çŤÍ `partial()` åĞ¡æŢřæÌěåŻžåőŽæ§ŘäžŻåŔĆæŢřåĂijïijŽ

```
>>> from functools import partial
>>> s1 = partial(spam, 1) # a = 1
>>> s1(2, 3, 4)
1 2 3 4
>>> s1(4, 5, 6)
1 4 5 6
>>> s2 = partial(spam, d=42) # d = 42
>>> s2(1, 2, 3)
1 2 3 42
>>> s2(4, 5, 5)
4 5 5 42
>>> s3 = partial(spam, 1, 2, d=42) # a = 1, b = 2, d = 42
>>> s3(3)
1 2 3 42
>>> s3(4)
1 2 4 42
>>> s3(5)
1 2 5 42
>>>
```

åŔŕäžěçIJŃåĞž `partial()` åŽžåőŽæ§ŘäžŽåŔĆæŢřåźűè£ŤåŻđäÿĂäÿłæŰřçŽĎcallableåŕźèśąãĂĆè£Źã
çĎűåŘŐèů§äźŃåĽ■åůšçżŔèţŃåĂijè£ĞçŽĎåŔĆæŢřåŘĹåźűèţůæİěïijŇæIJĂåŘŐåřĘæĽĂæIJĽåŔĆæŢřäijäéĂ

èőÍèőž

æIJňèŁĆèęĄèğčåĘşçŽĎéŮőéćŸæŸŕèőľåŐ§æIJňäÿ■åĚijåőźçŽĎäžčçăĄåŔŕäžěäÿĂèţůåůěä¡IJãĂĆäÿŃéİ

çňňäÿĂäÿłä¿Ńå■ŘæŸŕïijŇåĄĞèő¿ä¡ăæIJĽäÿĂäÿłçĆźçŽĎåĹŮèąĺæİěèąĺçď'ž(x,y)åİŘæăĞåĚČçżĎãĂĆ
ä¡ăåŔŕäžěä¡£çŤĺäÿŃéİćçŽĎåĞ¡æŢřæİěèőąçőŮäÿd'çĆźäźŃéŮť'çŽĎèůİçężïijŽ

```
points = [ (1, 2), (3, 4), (5, 6), (7, 8) ]

import math
def distance(p1, p2):
    x1, y1 = p1
    x2, y2 = p2
    return math.hypot(x2 - x1, y2 - y1)
```

çŐřåIJĺåĄĞèő¿ä¡ăæČşäžěæ§ŘäÿłçĆźäÿžå§žçĆźïijŇæăźæ■őçĆźåŠŇå§žçĆźäźŃéŮť'çŽĎèůİçężæİěæŐŠå
åĹŮèąĺçŽĎ `sort()` æŰźæşŢæŐěåŘŮäÿĂäÿłåĚşéŤőå■ŮåŘĆæŢřæİěèĞłåőŽäźĽ'æŐŠåžŔéÀżè¿ŠïijŇ
ä¡ĘæŸŕåőĆåŔłèČ¡æŐěåŘŮäÿĂäÿłå■ŢäÿłåŔĆæŢřçŽĎåĞ¡æŢř(distance()å¿ĹæŸŐæŸ¿æŸŕäÿ■çňęåŘĹæİąäžűĺ
çŐřåIJĺæĹŚäżňåŔŕäžěéĂŽè£Ğä¡£çŤĺ `partial()` æİěèğčåĘşè£ŹäÿłéŮőéćŸïijŽ

```
>>> pt = (4, 3)
>>> points.sort(key=partial(distance,pt))
>>> points
[(3, 4), (1, 2), (5, 6), (7, 8)]
>>>
```

æŻť’è£ŻäÿĂæ■ëïijŇpartial() éĂŽåÿÿèćńçŤĺæİěå¿őèřČåĚűäžŰåžŞåĞ¡æŢřæĽ’Ää¡£çŤĺçŽĎåŻđèřČå
ä¿ŇåęĆïijŇäÿŇéİćæŸŕäÿĂæőţäžčçăĄïijŇä¡£çŤĺ multiprocessing
æİěåijĆæ■ěèőąçőŰäÿĂäÿłçżŞædIJåĂijïijŇ çĎűåŘŐè£Źäÿłå Äijèćńäijąé ĂŠçżŹäÿĂäÿłæŐěåŘŰäÿĂäÿłresultå

```
def output_result(result, log=None):
    if log is not None:
        log.debug('Got: %r', result)

# A sample function
def add(x, y):
    return x + y

if __name__ == '__main__':
    import logging
    from multiprocessing import Pool
    from functools import partial

    logging.basicConfig(level=logging.DEBUG)
    log = logging.getLogger('test')

    p = Pool()
    p.apply_async(add, (3, 4), callback=partial(output_result,␣
↪log=log))
    p.close()
    p.join()
```

å¡ŞçżŹ apply_async() æŔŘä¿ŻåŻđèřČåĞ¡æŢřæŮűïijŇéĂŽè£Ğä¡£çŤĺ
partial() äijąéĂŠéćİåď’ŰçŽĎ logging åŔĆæŢřãĂĆ èĂŇ multiprocessing
åŕźè£ŹäžŻäÿĂæŮăæĽĂç§ěâĂŤâĂŤåőČäžĚäżĚåŘłæŸŕä¡£çŤĺå■Ţäÿłå Äijæİèèřç Ťĺå ŻđèřČåĞ¡æŢřãĂĆ

ä¡IJäÿžäÿĂäÿłçśżäijijçŽĎä¿Ńå■ŘïijŇèĂČèŹŚäÿŃçijŰåĘŹç¡ŚçżIJæIJ■åŁąåŹĺçŽĎéŮőécŸïijŇsockets
æĺąåİŮèőľåőČåŘŸå¿Ůå¿Ĺåőźæ ŸŞãĂĆ äÿŃéİćæŸŕäÿłçőĂå■ŢçŽĎechoæIJ■åŁąåŹĺïijŽ

```
from socketserver import StreamRequestHandler, TCPServer

class EchoHandler(StreamRequestHandler):
    def handle(self):
        for line in self.rfile:
            self.wfile.write(b'GOT:' + line)

serv = TCPServer(('', 15000), EchoHandler)
serv.serve_forever()
```

äÿ■è£ĞïijŇåĄĞèő¿ä¡ăæČşçżŹEchoHandleråćđåŁääÿĂäÿłåŔŕäžěæŐěåŘŰåĚűäžŰéÉ■ç¡őéĂĽ’éąźçŽĎ
__init__ æŰźæşŢãĂĆæŕŤåęĆïijŽ

```
class EchoHandler(StreamRequestHandler):
    # ack is added keyword-only argument. *args, **kwargs are
    # any normal parameters supplied (which are passed on)
    def __init__(self, *args, ack, **kwargs):
        self.ack = ack
```

```
        super().__init__(*args, **kwargs)

    def handle(self):
        for line in self.rfile:
            self.wfile.write(self.ack + line)
```

è£ŹäźĹä£őæŤźåŘŐïijŇæĹŚäžňåřšäÿ■éIJĂèęĄæŸ¿åijŔåIJřåIJÍTCPServerçśžäÿ■æůžåŁăåL'■çijĂäžĘãĂ
ä¡ĘæŸŕä¡ăåĘ■æňąè£ŘèąŇçÍŃåžŔåŘŐäijŽæŁěçśžäijijäÿŃéİćçŽĎéŤŹèŕŕïijŽ

```
Exception happened during processing of request from ('127.0.0.1',␣
↪59834)
Traceback (most recent call last):
...
TypeError: __init__() missing 1 required keyword-only argument: 'ack
↪'
```

åĹİçIJŃètůæİěåě¡åČŔå¿ĹéŽ¿ä£őæ■čè£ŹäÿłéŤŹèŕŕïijŇéŹď äžĘä£őæŤź
`socketserver` æĺąåİŮæžŘäžčçăĄæĹŰèĂĚä¡£çŤĺæ§ŘäžŽåěĞæĂłçŽĎæŰźæşŢäźŃåď ŰãĂĆ
ä¡ĘæŸŕïijŇåęĆæđIJä¡£çŤĺ `partial()` åŕśèČ¡å¿Ĺè¡żæİ¿çŽĎèğčåĘşâĂŤâĂŤçżŹåőČäijăéĂŠ
`ack` åŔĆæŢřçŽĎåĂijæİěåĹİåğŃåŇŰå■şåŔŕïijŇåęĆäÿŃïijŽ

```
from functools import partial
serv = TCPServer(('', 15000), partial(EchoHandler, ack=b'RECEIVED:
↪'))
serv.serve_forever()
```

åIJĺè£Źäÿłä¿Ńå■Řäÿ■ïijŇ`__init__()` æŰźæşŢäÿ■çŽĎack-
åŔĆæŢřåčřæŸŐæŰźåijŔçIJŃäÿŁåŐżå¿ĹæIJL'èűčïijŇåĚűåőđåřśæŸŕåčřæŸŐackäÿžäÿĂäÿłåijžåĹűåĚşéŤőå■Ů
åĚşäžŐåijžåĹűåĚşéŤőå■ŮåŔĆæŢřéŰőécŸæĹŚäžňåIJĺ7.2åřŔèŁĆæĹŚäžňåůšçżŔèőĺèőžè£ĞäžĘïijŇèŕźèĂĚåŕ

å¿Ĺåď ŽæŮűåĂŹ `partial()` èČ¡åőđçŐřçŽĎæŢĹæđIJïijŇlambdaèąĺè¿¿åijŔäź§èČ¡åőđçŐřãĆĆæŕŤåę

```
points.sort(key=lambda p: distance(pt, p))
p.apply_async(add, (3, 4), callback=lambda result: output_
↪result(result,log))
serv = TCPServer(('', 15000),
        lambda *args, **kwargs: EchoHandler(*args, ack=b'RECEIVED:',
↪ **kwargs))
```

è£ŹæăůåĘŹäź§èČ¡åőđçŐřåŘŇæăůçŽĎæŢĹæđIJïijŇäÿ■è£ĞçŻÿæŕŤèĂŇåůšäijŽæŸ¿å¿ŮæŕŤè¿ČèĞČèĆ
è£ŹæŮűåĂŹä¡£çŤĺ `partial()` åŔŕäžěæŻť'åŁăçŻť'èğĆçŽĎèąĺè¿¿ä¡ăçŽĎæĎŔåŻ¿(çżŹæ§ŘäžŽåŔĆæŢřécĎ

7.9 åřĘå■ŢæŰźæşŢçŽĎçśžè¡ňæ■ćäÿžåĞ¡æŢř

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłéŹď `__init__()` æŰźæşŢåď ŰåŔłåőŽäźL'äžĘäÿĂäÿłæŰźæşŢçŽĎçśžãĂĆäÿžäžĘçőĂ

èğčåĘşæŰźæąĹ

åď ğåď ŽæŢřæČĚåĘţäÿŃïijŇåŔŕäżěä¡£çŤĺéŮ■åŇĚæİěářĘå■Ţäÿ łæŰźæşŢçŽĎçśżè¡ñæ■ćæĹŘåĞ¡æŢřãĂ
äÿ¿äÿłä¿Ńå■ŘïijŇäÿŃéİćçď žä¿Ńäÿ■çŽĎçśżåĚĄèőÿä¡£çŤĺèĂĚæăźæ■őæ§Řäÿłæĺąæİ£æŰźæąĹæİěèŐůåŔŰå

```
from urllib.request import urlopen

class UrlTemplate:
    def __init__(self, template):
        self.template = template

    def open(self, **kwargs):
        return urlopen(self.template.format_map(kwargs))

# Example use. Download stock data from yahoo
yahoo = UrlTemplate('http://finance.yahoo.com/d/quotes.csv?s={names}
 ↪&f={fields}')
for line in yahoo.open(names='IBM,AAPL,FB', fields='sl1c1v'):
    print(line.decode('utf-8'))
```

è£ŹäÿłçśżåŕŕäżěèćńäÿĂäÿłæŻť çőĂå■ŢçŽĎåĞ¡æŢřæİěäžčæŻ£ïijŽ

```
def urltemplate(template):
    def opener(**kwargs):
        return urlopen(template.format_map(kwargs))
    return opener

# Example use
yahoo = urltemplate('http://finance.yahoo.com/d/quotes.csv?s={names}
 ↪&f={fields}')
for line in yahoo(names='IBM,AAPL,FB', fields='sl1c1v'):
    print(line.decode('utf-8'))
```

èőÍèőž

åď ğéČĺåĹĘæČĚåĘţäÿŃïijÑä¡ăæŃěæIJL’äÿĂäÿłå■ŢæŰźæşŢçśżçŽĎåŐ§åŻăæŸŕéIJĂèęĄå■ŸåĆĺæ§ŘäžŽ
æŕŤåęĆïijŇåőŽäźL’UrlTemplateçśżçŽĎåŤŕäÿĂçŻőçŽĎåřśæŸŕåĚĹåIJĺæ§Řäÿłå IJřæŰźå■ŸåĆĺæĺąæİ£åĂijïijŃ

ä¡£çŤĺäÿĂäÿłåĘĚéČĺåĞ¡æŢřæĹŰèĂĚéŮ■åŇĚçŽĎæŰźæąĹéĂŽåÿÿäijŽæŻť äijŸéŻĚäÿĂäžŻãĂĆçőĂå■
åŘłäÿ■è£ĞåIJĺåĞ¡æŢřåĘĚéČĺåÿęäÿŁäžĘäÿĂäÿłéćİåď ŰçŽĎåŔŸéĞŘçŐŕåćČãĂĆéŮ■åŇĚåĚşéŤőçL’źçĆźåřśå
åŻăæ■d’ïijŇåIJĺæĹŚäžňçŽĎèğčåĘşæŰźæąĹäÿ■ïijŇopener() åĞ¡æŢřèőřä¡ŔäžĘ
template åŔĆæŢřçŽĎåĂijïijŇåźűåIJĺæŐěäÿŃæİěçŽĎèřĈçŤĺäÿ■ä¡£çŤĺåőĈãĂĆ

äżżä¡ŢæŮűåĂŹåŘłèęĄä¡ăçćřåĹřéIJĂèęĄçżŹæ§ŘäÿłåĞ¡æŢřåćđåŁäéćİåď ŰçŽĎçŁűæĂĄä£ąæĄŕçŽĎéŮ
çŻÿæŕŤåřĘä¡ăçŽĎåĞ¡æŢřè¡ñæ■ćæĹŘäÿĂäÿłçśżèĂŇèĺĂïijŇéŮ■åŇĚéĂŽåÿÿæŸŕäÿĂçğ■æŻť åŁăçőĂæť ĄåŠ

7.10 åÿęéćİåď ŰçŁűæĂĄä£ąæĄŕçŽĎåŻđèřĈåĞ¡æŢř

éŮőécŸ

ä¡ăçŽĎäžčçăĄäÿ■éIJĂèęĄä¿İèţŰåĹřåŻđèřČåĞ¡æŢřçŽĎä¡£çŤĺ(æŕŤåęĆäžŃäžűåď ĎçŘĘåŹĺãĂĄç■L'å¿Ě
åźűäÿŤä¡äè£ŸéIJĂèęĄèől'åŻđèřČåĞ¡æŢřæŃěæIJL'éćİåď ŰçŽĎçŁűæĂĄåĂijïijŃäžëä¿£åIJĺåőČçŽĎåĘĚéČĺä

èğčåEşæŰźæąĹ

è£ŹäÿĂářŔèŁĆäÿżèęĄèőĺèőžçŽĎæŸŕéĆčäžŻåĞžçŐřåIJĺå¿Ĺåď'ŽåĞ¡æŢřåžŞåŠŇæąĘæđűäÿ■çŽĎåŻđèř
äÿžäžĘæijŤçď'žäÿŐæţŃèŕŢïijŇæĹŚäżňåĚĹåőŽäźL'åęĆäÿŃäÿĂäÿłéIJĂèęĄèřČçŤĺåŻđèřČåĞ¡æŢřçŽĎåĞ¡æŢ

```
def apply_async(func, args, *, callback):
    # Compute the result
    result = func(*args)

    # Invoke the callback with the result
    callback(result)
```

åőđéŹĚäÿŁïijŃè£ŹæőţäžčçăĄåŔŕäžěåĄŽäżżä¡ŢæŻt'éńŸçžğçŽĎåď'ĎçŘĘïijŇåŇĚæŃňçž£çĺŃãĂĄè£Żç
æĹŚäżňäžĚäžĚåŔłéIJĂèęĄåĚşæşĺåŻđèřČåĞ¡æŢřçŽĎèřČçŤĺãĆĆäÿŃéİćæŸŕäÿĂäÿłæijŤçď'žæĂŐæăůä¡£çŤĺ

```
>>> def print_result(result):
...     print('Got:', result)
...
>>> def add(x, y):
...     return x + y
...
>>> apply_async(add, (2, 3), callback=print_result)
Got: 5
>>> apply_async(add, ('hello', 'world'), callback=print_result)
Got: helloworld
>>>
```

æşĺæĎŔåĹř `print_result()` åĞ¡æŢřäžĚäžĚåŔłæŐěåŔŮäÿĂäÿłåŔĆæŢř `result`
ãĂĆäÿ■èČ¡åĘ■äijăåĚěåĚűäžŰä£ąæĄŕãĂĆ èĂŇå¡Şä¡äæČşèől'åŻđèřČåĞ¡æŢřèő£éŮőåĚűäžŰåŔŸéĞŔæĹŰèĂ

äÿžäžĘèől'åŻđèřČåĞ¡æŢřèő£éŮőåď ŰéČĺä£ąæĄŕïijŃäÿĂçğ■æŰźæşŢæŸŕä¡£çŤĺäÿĂäÿłçżŚåőŽæŰźæşŢ
æŕŤåęĆïijŃäÿŃéİćè£ŹäÿłçśžäijŽä£Ía■ŸäÿĂäÿłåĘĚéČĺážŔåĹŮåŔůïijŃæŕŔæňąæŐěæŤűåĹřäÿĂäÿł
`result` çŽĎæŮűåĂŹážŔåĹŮåŔůåŁă1ïijŽ

```
class ResultHandler:

    def __init__(self):
        self.sequence = 0

    def handler(self, result):
        self.sequence += 1
        print('[{}] Got: {}'.format(self.sequence, result))
```

ä¡£çŤĺè£ŹäÿłçśžçŽĎæŮűåĂŹïijŃä¡ăåĚĹåĹŽåžžäÿĂäÿłçśžçŽĎåőđä¿ŃïijŃçĎűåŘŐçŤĺåőČçŽĎ
`handler()` çżŚåőŽæŰźæşŢæİěåĄŽäÿžåŽđèřČåĞ¡æŢřijŽ

```
>>> r = ResultHandler()
>>> apply_async(add, (2, 3), callback=r.handler)
[1] Got: 5
>>> apply_async(add, ('hello', 'world'), callback=r.handler)
[2] Got: helloworld
>>>
```

çñňäžŇçğ■æŰźåijŔïijŇä¡IJäÿžçśžçŽĎæŻ£äżčïijŇåŔŕäžěä¡£çŤĺäÿĂäÿłéŮ■åŇĚæ■ŢèŐůçŁűæĂĄåĂijïijŇ

```
def make_handler():
    sequence = 0
    def handler(result):
        nonlocal sequence
        sequence += 1
        print('[{}] Got: {}'.format(sequence, result))
    return handler
```

äÿŃéÍćæŸŕä¡£çŤĺéŮ■åŇĚæŰźåijŔçŽĎäÿĂäÿłä¿Ńå■ŘïijŽ

```
>>> handler = make_handler()
>>> apply_async(add, (2, 3), callback=handler)
[1] Got: 5
>>> apply_async(add, ('hello', 'world'), callback=handler)
[2] Got: helloworld
>>>
```

è£ŸæIJL’åŔęåď’ŰäÿĂäÿłæŻť’énŸçžğçŽĎæŰźæşŢïijŇåŔŕäžěä¡£çŤĺå■ŔçÍŃæÍěåőŇæĹŘåŘŇæăůçŽĎäžŇ

```
def make_handler():
    sequence = 0
    while True:
        result = yield
        sequence += 1
        print('[{}] Got: {}'.format(sequence, result))
```

åŕźäžŐå■ŔçÍŃïijŇä¡ăéIJĂèęĄä¡£çŤĺåőČçŽĎ `send()` æŰźæşŢä¡IJäÿžåŻđèřČåĞ¡æŢřïijŇåęĆäÿŃæL’Ăç

```
>>> handler = make_handler()
>>> next(handler) # Advance to the yield
>>> apply_async(add, (2, 3), callback=handler.send)
[1] Got: 5
>>> apply_async(add, ('hello', 'world'), callback=handler.send)
[2] Got: helloworld
>>>
```

èőÍèőž

å§žäžŐåŻđèřČåĞ¡æŢřçŽĎè¡ŕäżűéĂŽåÿÿéČ¡æIJL’åŔŕèČ¡åŔŸå¿ŮéÍđåÿÿåď’■æÍĆãĂĆäÿĂéČĺåĹĘåŐ§åŻ
åŻăæ■d’ïijŇèřůæśĆæL’ğèąŇåŠŇåď’ĎçŘĘçżŞæđIJäźŃéŮt’çŽĎæL’ğèąŇçŐŕåćČåőđéŹĚäÿŁåůšçżŔäÿćåď’śäžĘ

éĆčä¡ăářśå£Ěéąžå Őżèğčå Eşåę Ćä¡Ţä£Ìå■ŸåŠŇæĄćåď■çŻÿå Ěşç ŽĎçŁűæĂĄä£ąæĄŕäžĘãĂĆ

èĞşářŚæIJL'äÿď'çğ■äÿżèęĄæŰźåijŔæÌěæ■ŢèŐůåŠŇä£Ìå■ŸçŁűæĂĄä£ąæĄŕïijŇä¡ăåŔŕäžěåIJÍäÿĂäÿłářż
äÿď'çğ■æŰźåijŔçŻÿæŕŤïijŇéŮ■åŇĚæĹŰèőÿæŸŕæŻť'åŁăè¡żéĞŔçžğåŠŇèĞłçĎűäÿĂçĆźïijŇåŻăäÿžåőČäžňål
åőČäžňè£ŸèČ¡èĞłåŁĺæ■ŢèŐůæL'ĂæIJL'èćńä¡£çŤĺåĹřçŽĎåŔŸéĞŔãĂĆåŻăæ■ď'ïijŇä¡ăæŮăéIJĂåŐżæŃĚå£Ğ

åęĆæđIJä¡£çŤĺéŮ■åŇĚïijŇä¡ăéIJĂèęĄæşĺæĎŔářźéĆčäžŻåŔŕä£őæŤźåŔŸéĞŔçŽĎæŞ■ä¡IJãĂĆåIJÍäÿŁé
`nonlocal` åčřæŸŐèŕ■åŔęçŤĺæÌěæŇĞçď'žæŐěäÿŃæÌěçŽĎåŔŸéĞŔäijŻåIJÍåŻđèřČåĞ¡æŢřäÿ■èćńä£őæŤźãĄ

èĂŇä¡£çŤĺäÿĂäÿłå■ŔçÍŃæÌěä¡IJäÿžäÿĂäÿłåŻđèřČåĞ¡æŢřářśæŻť'æIJL'èűčäžĘïijŇåőČèů§éŮ■åŇĚæŰźa
æ§Ŕçğ■æĎŔäźL'äÿŁæÌěèőšïijŇåőČæŸ¿å¿ŮæŻť'åŁăçőĂæť'ĄïijŇåŻăäÿžæĂżåĚśåřśäÿĂäÿłåĞ¡æŢřèĂŇåůšãĂ
åźűäÿŤïijŇä¡ăåŔŕäžěå¿ĹèĞłçŤśçŽĎä£őæŤźåŔŸéĞŔèĂŇæŮăéIJĂåŐżä¡£çŤÍ `nonlocal`
åčřæŸŐãĂĆ è£Źçğ■æŰźåijŔåŤŕäÿĂçijžçĆźåřśæŸŕçŻÿåŕźäžŐåĚűäžŰPythonæŁĂæIJŕèĂŇèÍĂæĹŰèőÿæŕŤè¿
åŔęåď'Úè£ŸæIJL'äÿĂäžŻæŕŤè¿ČéŽ¿æĞĆçŽĎéČĺåĹĘïijŇæŕŤåęĆä¡£çŤĺäźŃåL'■éIJĂèęĄèřČçŤÍ
`next()` ïijŇåődéŹĚä¡£çŤĺæŮűè£Źäÿłæ■ěéłď'å¿Ĺåőźæ ŸŞèćńå£Ÿèőřã ĂĆ
ář¡çőąåęĆæ■ď'ïijŇå■ŔçÍŃè£ŸæIJL'åĚűäžŰçŤĺåď'ĎïijŇæŕŤåęĆä¡IJäÿžäÿĂäÿłåĘĚèĄŤåŻđèřČåĞ¡æŢřçŽĎåőŻ

åęĆæđIJä¡ăäžĚäžĚåŔłéIJĂèęĄçżŹåŻđèřČåĞ¡æŢřäijåéĂŠéćÌåď'ŰçŽĎåĂijçŽĎèŕÌïijŇè£ŸæIJL'äÿĂçğ■ä
`partial()` çŽĎæŰźåijŔäź§å¿ĹæIJL'çŤĺãĂĆ åIJÍæšąæIJL'ä¡£çŤÍ `partial()`
çŽĎæŮűåĂŹïijŇä¡ăåŔŕèČ¡çżŔåÿÿçIJŃåĹřäÿŃéÌćè£Źçğ■ä¡£çŤĺlambdaèąĺè¿¿åijŔçŽĎåď'■æÍĆäžčçăĄïijŽ

```
>>> apply_async(add, (2, 3), callback=lambda r: handler(r, seq))
[1] Got: 5
>>>
```

åŔŕäžěåŔĆèĂČ7.8åřŔèŁĆçŽĎåĞăäÿłçď'žä¿ŃïijŇæŢŹä¡ăåęĆä¡Ţä¡£çŤÍ `partial()`
æÌěæŻť'æŤźåŔĆæŢřç■¿åŘ■æÌěçőĂåŃŰäÿŁè£ŕäžčçăĄãĂĆ

7.11 åĘĚèĄŤåŻđèřČåĞ¡æŢř

éŮőécŸ

å¡Şä¡ăçijŰåĘŹä¡£çŤĺåŻđèřČåĞ¡æŢřçŽĎäžčçăĄçŽĎæŮűåĂŹïijŇæŃĚå£Čå¿Ĺåď'ŽåřŔåĞ¡æŢřçŽĎæL'l'å
ä¡ăäÿŇæIJŻæL'¿åĹřæ§ŔäÿłæŰźæşŢæÌěèől'äžčçăĄçIJŃäÿŁåŐżæŻť'åČŔæŸŕäÿĂäÿłæŹőéĂŽçŽĎæL'ğèąŇåžĚ

èğčåĘşæŰźæąĹ

éĂŽè£Ğä¡£çŤĺçŤ§æĹŘåŹĺåŠŇå■ŔçÍŃåŔŕäžěä¡£å¿ŮåŻđèřČåĞ¡æŢřåĘĚèĄŤåIJÍæ§ŔäÿłåĞ¡æŢřäÿ■ãĂĆ
äÿžäžĘæijŤçď'žèŕť'æŸŐïijŇåĄĞèő¿ä¡ăæIJL'åęĆäÿŃæL'Ăçď'žçŽĎäÿĂäÿłæL'ğèąŇæ§Ŕçğ■èőąçőŮäžžåŁąçĎű

```
def apply_async(func, args, *, callback):
    # Compute the result
    result = func(*args)

    # Invoke the callback with the result
    callback(result)
```

æŐěäÿŃæÌěèől'æĹŠäžńçIJŃäÿĂäÿŃäÿŃéÍćçŽĎäžčçăĄïijŇåőČåŃĚåŔńäžĘäÿĂäÿł
`Async` çśžåŠŇäÿĂäÿł `inlined_async` èčĚéěřåŹĺïijŽ

```
from queue import Queue
from functools import wraps

class Async:
    def __init__(self, func, args):
        self.func = func
        self.args = args

def inlined_async(func):
    @wraps(func)
    def wrapper(*args):
        f = func(*args)
        result_queue = Queue()
        result_queue.put(None)
        while True:
            result = result_queue.get()
            try:
                a = f.send(result)
                apply_async(a.func, a.args, callback=result_queue.
↪put)
            except StopIteration:
                break
    return wrapper
```

è£Źäÿd'äÿłäžčçăĄçL'ĞæőţåĚĄèőÿä¡ăä¡£çŤÍ `yield` èŕ■åŔěåĘĚèĄŤåŻđèřČæ■ěéłd'ãĂĆæŕŤåęĆïijŽ

```
def add(x, y):
    return x + y

@inlined_async
def test():
    r = yield Async(add, (2, 3))
    print(r)
    r = yield Async(add, ('hello', 'world'))
    print(r)
    for n in range(10):
        r = yield Async(add, (n, n))
        print(r)
    print('Goodbye')
```

åęĆædIJä¡ăèřČçŤÍ `test()` ïijŇä¡ăäijŽå¿ŮåĹřçśżäijijåęĆäÿŃçŽĎè¿ŞåĞžïijŽ

```
5
helloworld
0
2
4
6
8
10
12
```

```
14
16
18
Goodbye
```

ä¡ääijŽåŔŠçŐřïijŇéŹď äžĘéĆčäÿłçL’źåĹńçŽĎèčĚéěřåŹĺåŠŇ `yield` èŕ■åŘěåď ŰïijŇåĚűäžŰåIJřæŰźåźűæšąæIJL’åĞžçŐřäżżä¡ŢçŽĎåŻđèřČåĞ¡æŢř(åĚűåőđæŸŕåIJĺåŘŐåŔřåőŽäź

èőĺèőž

æIJňåřŔèŁĆäijŽåőđåőđåIJĺåIJĺçŽĎæţŃèŕŢä¡ääĚşäžŐåŻđèřČåĞ¡æŢřãĂĄçŤ§æĹŘåŹĺåŠŇæŐğåĹűæţĄçŽ

éęŰåĚĹïijŇåIJĺéIJĂèęĄä¡£çŤĺåĹřåŻđèřČçŽĎäżčçăĄäÿ■ïijŇåĚşéŤőçĆźåIJĺäžŐå¡ŞåL’■èőąçŐůåůëä¡IJäij
å¡Şèőąçő̊ŮéĞ■åŘŕæŮűïijŇåŻđèřČåĞ¡æŢřèćńèřČçŤĺæİěçżğçż■åď ĎçŘĘçżŞæđIJãĂĆapply_async()
åĞ¡æŢřæijŤçď žäžĘæL’ğèąŇåŻđèřČçŽĎåőđéŹĚéĂżè¿ŚïijŇ år¡çőąåőđéŹĚæČĚåĘţäÿ■åőČåŘŕèČ¡äijŽæŻť’åŁä

èőąçőŮçŽĎæŽĆåĄIJäÿŐéĞ■åŘŕæĂİèůŕèů§çŤ§æĹŘåŹĺåĞ¡æŢřçŽĎæL’ğèąŇæĺąåđŃäÿ■èřŃèĂŇåŘĹãĂ
åĚůä¡ŞæİěèőšïijŇ`yield` æŞ■ä¡IJäijŽä¡£äÿĂäÿłçŤ§æĹŘåŹĺåĞ¡æŢřäžğçŤ§äÿĂäÿłåĂijåźűæŽĆåĄIJãĂĆ
æŐěäÿŃæİěèřČçŤĺçŤ§æĹŘåŹĺçŽĎ `__next__()` æĹŰ `send()`
æŰźæşŢåŔĹäijŽèőľ’åőČäżŐæŽĆåĄIJåď ĎçżğçżæL’ğèąŇãĂĆ

æăźæ■őè£Źäÿłæ ĂİèůŕïijŇè£ŹäÿĂåřŔèŁĆçŽĎæäÿå£Čåřšåİĺ `inline_async()`
èčĚéěřåŹĺåĞ¡æŢřäÿ■äžĘãĂĆ åĚşéŤőçĆźåřśæŸŕïijŇèčĚéěřåŹĺäijŽéĂŘæ■ěéĄ■åŐĘçŤ§æĹŘåŹĺåĞ¡æŢřçŽĎæ
`yield` èŕ■åŘěïijŇæŕŔäÿĂæňąäÿĂäÿłãĂĆ äÿžäžĘè£ŹæăůåĄŽïijŇåĹŽåijĂåğŃçŽĎæŮűåĂŹåĹŽåžžäžĘäÿĂä
`result` éŸ§åĹŮåźűåŘŠéĞŃéİćæŤ¿åĚěäÿĂäÿł `None` åĂijãĂĆ
çĎűåŘŐåijĂåğŃäÿĂäÿłå¿łçŐŕæŞ■ä¡IJïijŇäžŐéŸ§åĹŮäÿ■åŘŰåĞžçżŞæđIJåĂijåźűåŘŠéĂĄçżŹçŤ§æĹŘåŹĺï
`yield` èŕ■åŘěïijŇ åIJĺè£ŹéĞŃäÿĂäÿł `Async` çŽĎåőđä¿ŃèćńæŐěåŘŮåĹřãĂĆçĎűåŘŐå¿łçŐŕåijĂåğŃæčĂæ
`apply_async()` ãĂĆ çĎűèĂŇïijŇè£Źäÿłèőąçő̊ŮæIJL’äÿłæIJĂèŕąåijĆéČĺåĹĘæŸŕåőČåźűæšąæIJL’ä¡£çŤĺä
`put()` æŰźæşŢæİěåŽđèřČãĂĆ

è£ŹæŮűåĂŹïijŇæŸŕæŮűåĂŹèŕęçżĘèğčéĞŁäÿŃåĹřåžŢåŔŠçŤ§äžĘäżĂäźĹäžĘãĂĆäÿżå¿łçŐŕçńŃå■şè£
`get()` æŞ■ä¡IJãĂĆ åęĆæđIJæŢřæ■őå■ŸåIJĺïijŇåőČäÿĂåőŽæŸŕ `put()`
åŽđèřČå■ŸæŤ¿çŽĎçżŞæđIJãĂĆåęĆæđIJæšąæIJL’æŢřæ■őïijŇéĆčäźĹåĚĹæŽĆåĄIJæŞ■ä¡IJåźűç■L’å¿ĚçżŞæ
è£ŹäÿłåĚůä¡ŞæĂŐæăůåőđçŐŕæŸŕçŤś `apply_async()` åĞ¡æŢřæİěåĘşåőŽçŽĎãĂĆ
åęĆæđIJä¡ääÿ■çŻÿä£qäijŽæIJL’è£ŹäźĹçěđåěĞçŽĎäžŇæČĚïijŇä¡ääŔŕäżěä¡£çŤĺ
`multiprocessing` åžŞæİěèŕŢäÿĂäÿŃïijŇ åIJĺå■ŢçŃńçŽĎè£ŻçĺŃäÿ■æL’ğèąŇåijĆæ■ěèőąçő̊ŮæŞ■ä¡IJïijŇ

```
if __name__ == '__main__':
    import multiprocessing
    pool = multiprocessing.Pool()
    apply_async = pool.apply_async

    # Run the test function
    test()
```

åőđéŹĚäÿŁä¡ääijŽåŔŠçŐřè£ŹäÿłçIJ§çŽĎåřśæŸŕè£ŹæăůçŽĎïijŇä¡ĘæŸŕèęĄèğčéĞŁæÿĚæěŽåĚůä¡ŞçŽĹ

åřĘåď ■æİĆçŽĎæŐğåĹűæţĄéŽŘèŮŘåĹřçŤ§æĹŘåŹĺåĞ¡æŢřèČŃåŘŐçŽĎä¿Ńå■ŘåIJĺæăĞåĞĘåžŞåŠŇç
æŕŤåęĆïijŇåIJĺ `contextlib` äÿ■çŽĎ `@contextmanager`
èčĚéěřåŹĺä¡£çŤĺäžĘäÿĂäÿłäžď äžžèť źèğčçŽĎæŁĂåůğïijŇ éĂŽè£ĞäÿĂäÿł `yield`

èŕ■åŔěåřĘè£ŻåĚěåŠŇçężåijĂäÿŁäÿŃæŰĞçőąçŘĘåŹÍçšŸåŘĹåIJÍäÿĂèţůãĂĆ
åŔęåd'ŰéÍđåÿÿæţĄèąŇçŽĎ Twisted åŇĚäÿ■äź§åŇĚåŘńäžĘéÍđåÿÿçśżäijijçŽĎåĘĚèĄŤåŻđèřČãĂĆ

7.12 èő£éŮőéŮ■åŇĚäÿ■åőŽäźĽçŽĎåŔŸéĞŔ

éŮőécŸ

ä¡ăæČşèęĄæĽl'åśŢåĞ¡æŢřäÿ■çŽĎæ§ŘäÿłéŮ■åŇĚïijŇåĚĄèőÿåőČèČ¡èő£éŮőåŠŇä£őæŤźåĞ¡æŢřçŽĎå

èğčåĘşæŰźæąĹ

éĂŽåÿÿæÍěèőšïijŇéŮ■åŇĚçŽĎåĘĚéČĺåŔŸéĞŔåŕźäžŐåd'ŰçŢŇæÍěèőšæŸŕåőŇåĚĺéŽŘèŮŔçŽĎãĂĆ
ä¡ĘæŸŕïijŇä¡ăåŔŕäžěéĂŽè£ĞçijŰåĘŹèő£éŮőåĞ¡æŢřåźűåřĘåĚűä¡IJäÿžåĞ¡æŢřåśđæĂğçżŠåőŽåĹřéŮ■åŇĚäÿ

```
def sample():
    n = 0
    # Closure function
    def func():
        print('n=', n)

    # Accessor methods for n
    def get_n():
        return n

    def set_n(value):
        nonlocal n
        n = value

    # Attach as function attributes
    func.get_n = get_n
    func.set_n = set_n
    return func
```

äÿŃéÍćæŸŕä¡£çŤÍçŽĎä¿Ńå■Ř:

```
>>> f = sample()
>>> f()
n= 0
>>> f.set_n(10)
>>> f()
n= 10
>>> f.get_n()
10
>>>
```

èőÍèőž

äÿžäžĘèŕťæŸŐæÿĚæěŽåőČåęĆä¡Ţåůëä¡IJçŽĎïijŇæIJL’äÿď’çĆźéIJĂèęĄèğčéĞŁäÿĂäÿŃãĂĆéęŰåĚĹïij
åčřæŸŐåŘŕäżěèőľ’æĹŚäžňçijŰåĘŹåĞ¡æŢřæİěä£őæŤźåĘĚéČĺåŔŸéĞŘçŽĎåĂijãĂĆ
åĚűæňąïijŇåĞ¡æŢřåśđæĂğåĚĄèőÿæĹŚäžňçŤĺäÿĂçğ■å¿ĹçőĂå■ŢçŽĎæŰźåijŔåřĘèő£éŮőæŰźæşŢçżŚåőŽåĹ

è£ŸåŘŕäżěè£ŻäÿĂæ■ěçŽĎæĽ’ľåśŢïijŇèőľéŮ■åŇĚæĺąæŃ§çśżçŽĎåőđä¿ŃãĂĆä¡äèęĄåĄŽçŽĎäżĚäżĚa

```
import sys
class ClosureInstance:
    def __init__(self, locals=None):
        if locals is None:
            locals = sys._getframe(1).f_locals

        # Update instance dictionary with callables
        self.__dict__.update((key,value) for key, value in locals.
↪items()
                             if callable(value) )
    # Redirect special methods
    def __len__(self):
        return self.__dict__['__len__']()

# Example use
def Stack():
    items = []
    def push(item):
        items.append(item)

    def pop():
        return items.pop()

    def __len__():
        return len(items)

    return ClosureInstance()
```

äÿŃéÍćæŸŕäÿĂäÿłäžď’äžŠåijŔäijŽèŕÍæİěæijŤçď’žåőČæŸŕåęĆä¡Ţåůëä¡IJçŽĎïijŽ

```
>>> s = Stack()
>>> s
<__main__.ClosureInstance object at 0x10069ed10>
>>> s.push(10)
>>> s.push(20)
>>> s.push('Hello')
>>> len(s)
3
>>> s.pop()
'Hello'
>>> s.pop()
20
>>> s.pop()
```

```
10
>>>
```

æIJL'èűčçŽĎæŸŕïijŇè£Źäÿłäžčçă Ąè£ŘèąŇètůæİěäijŽæŕŤäÿĂäÿłæŹőéĂŽçŽĎçśżåőŽäźL'èęĄå£ńå¿Ĺåď

```
class Stack2:
    def __init__(self):
        self.items = []

    def push(self, item):
        self.items.append(item)

    def pop(self):
        return self.items.pop()

    def __len__(self):
        return len(self.items)
```

åęĆædIJè£ŹæăůåĄŽïijŇä¡ăäijŽå¿ŮåĹřçśżäijijåęĆäÿŇçŽĎçżŞædIJïijŽ

```
>>> from timeit import timeit
>>> # Test involving closures
>>> s = Stack()
>>> timeit('s.push(1);s.pop()', 'from __main__ import s')
0.9874754269840196
>>> # Test involving a class
>>> s = Stack2()
>>> timeit('s.push(1);s.pop()', 'from __main__ import s')
1.0707052160287276
>>>
```

çżŞædIJæŸ¿çď'žïijŇéŮ■åŇĚçŽĎæŰźæĄè£ŘèąŇètůæİěèęĄå£ńåď'ğæęĆ8%ïijŇåď'ğéČĺåĹĘåŐ§åŻăæŸ
éŮ■åŇĚæŻť'å£ńæŸŕåŽääÿžäÿ■äijŽæűL'åŔŁåĹřéćİåď'ŰçŽĎselfåŔŸéĞŔãĂĆ

Raymond HettingerářźäžŐè£ŹäÿłéŮőécŸèő¿èőąåĞžäžĘæŻť'åŁäéŽ¿äżěçŘĘèğčçŽĎæŤźè£ŻæŰźæąĹãĂ
èĂŇäÿŤåőČåŔłæŸŕçIJ§åődçśżçŽĎäÿĂäÿłåěĞæĂłçŽĎæŻ£æ■ćèĂŇåůšïijŇä¿ŃåęĆïijŇçśżçŽĎäÿżèęĄçL'źæ
åźűäÿŤä¡äèęĄåĄŽäÿĂäžŻåĚűäżŰçŽĎåůěä¡IJæL'■èČ¡èőľäÿĂäžŻçL'źæőŁæŰźæşTçŤ§æŢĹ(æŕŤåęĆäÿŁéIć
`ClosureInstance` äÿ■éĞ■åĘŹè£ĞçŽĎ `__len__()` åődçŐřãĂĆ)

æIJĂåŘŐïijŇä¡ăåŔŕèČ¡è£ŸäijŽèőľåĚűäżŰéŸĖèŕża¡ääžčçăĄçŽĎäžžæĎ§åĹřçŰŚæČŚïijŇäÿžäżĂäźĹåőČ
(å¡ŞçĎűïijŇäżŰäżňäź§æČşç§ěéĄŞäÿžäżĂäźĹåőČè£ŘèąŇètůæİěäijŽæŻť'å£ń)ãĂĆår¡çőąåęĆæ■d'ïijŇè£Źårźä

æĂżä¡ŞäÿŁèőšïijŇåIJĺéĚ■ç¡őçŽĎæŮűåĂŹçżŹéŮ■åŇĚæůżåŁăæŰźæşŢäijŽæIJL'æŻť'åď'ŽçŽĎåőđçŤĺåŁ
æŕŤåęĆä¡ăéIJĂèęĄéĞ■ç¡őåĘĚéČĺçŁűæĂĄãĂĄåĹůæŰřçijŞåĘšåŇžãĂĄæÿĚéŹď'çijŞå■ŸæĹŮåĚűäżŰçŽĎåŕ

çňňåĚńçńäïijŽçśżäÿŐårźèšą

æIJňçńääÿżèęĄåĚşæşĺçĆźçŽĎæŸŕåŠŇçśżåőŽäźL'æIJL'åĚşçŽĎåÿÿèğĄçijŰçĺŃæĺąåđŃãĂĆåŇĚæŃňèől'
çśżårĄèčĚæŁĂæIJŕãĂĄçżğæL'£ãĂĄåĘĚå■ŸçőąçŘĘäžěåŔŁæIJL'çŤĺçŽĎèő¿èőąæĺąåijŔãĂĆ

Contents:

8.1 æŤźåŔŸåŕźèśąçŽĎå■ŮçňęäÿšæŸ¿çď'ž

éŮőécŸ

ä¡ăæČşæŤźåŔŸåŕźèśąåőđä¿ŃçŽĎæĽŞå■řæĹŰæŸ¿çď'žè¿ŞåĞžïijŇèől'åőČäžňæŻt'åĚůåŔŕèŕżæĂğãĂĆ

èğčåĘşæŰźæąĹ

èęĄæŤźåŔŸäÿĂäÿłåőđä¿ŃçŽĎå■Ůçňęäÿšèąĺçď'žïijŇåŔŕéĞ■æŰřåőŽäźL'åőČçŽĎ __str__() åŠŇ __repr__() æŰźæşŢãĂĆä¿ŃåęĆïijŽ

```
class Pair:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __repr__(self):
        return 'Pair({0.x!r}, {0.y!r})'.format(self)

    def __str__(self):
        return '({0.x!s}, {0.y!s})'.format(self)
```

__repr__() æŰźæşŢè£ŤåŻđäÿĂäÿłåőđä¿ŃçŽĎäžčçăĄèąĺçď'žå¡ćåijŔïijŇéĂŽåÿÿçŤĺæİěéĞ■æŰřædĎ åĘĚç¡őçŽĎ repr() åĞ¡æŢřè£ŤåŻđè£ŹäÿłåĞ■ŮçňęäÿšïijŇèů§æĹŠäžňä¡£çŤĺäžď'äžŠåijŔèğčéĞĽåŹĺæŸ¿çď'žc __str__() æŰźæşŢåřĘåőđä¿Ńè¡ňæ■ćäÿžäÿĂäÿłå■ŮçňęäÿšïijŇä¡£çŤĺ str() æĹŰ print() åĞ¡æŢřäijŽè¿ŞåĞžè£ŹäÿłåĞ■Ůçňęäÿšã ĂĆærŤåęĆïijŽ

```
>>> p = Pair(3, 4)
>>> p
Pair(3, 4) # __repr__() output
>>> print(p)
(3, 4) # __str__() output
>>>
```

æĹŠäžňåIJĺè£ŹéĞŇè£ŸæijŤçď'žäžĘåIJĺæăijåijŔåŇŰçŽĎæŮűåĂŹæĂŐæăůä¡£çŤĺäÿ■åŘŇçŽĎå■Ůçňęäÿ çĽźåĹńæİèèőšïijŇ!r æăijåijŔåŇŰäžčçăĄæŇĞæŸŐè¿ŞåĞžä¡£çŤĺ __repr__() æİěäžčæŽ£éżŸèőd'çŽĎ __str__() ãĂĆ ä¡ăåŔŕäžěçŤĺåĽ■éĺćçŽĎçśžæİèèŕŢçİĂæţŇèŕŢäÿŇïijŽ

```
>>> p = Pair(3, 4)
>>> print('p is {0!r}'.format(p))
p is Pair(3, 4)
>>> print('p is {0}'.format(p))
p is (3, 4)
>>>
```

èőÍèőž

èĞłåőŽäźĽ' __repr__() åŠŇ __str__() éĂŽåÿÿæŸŕå¿Ĺåě¡çŽĎäźžäæČŕïijŇåŻääÿžåőČèČ¡çőĂåŇŰä¿ŇåęĆïijŇåęĆæđIJäżĚäżĚåŔłæŸŕæĽŞå■řè¿ŞåĞžæĹŰæŮěå£Ůè¿ŞåĞžæ§Řäÿłåőđä¿ŇïijŇéĆčäźĹçÍŃåžŔåŚ

__repr__() çŤ§æĹŘçŽĎæŰĞæIJňå■ŮçňęäÿšæäĞåĞĘåĄŽæşŢæŸŕéIJĂèęĄèől' eval(repr(x)) == x äÿžçIJ§ãĂĆ åęĆæđIJåőđåIJÍäÿ■èČ¡è£ŹæäůåŔåĄŽïijŇåžŤèŕěåĹŻåžžäÿĂäÿłæIJL < åŠŇ > æŃňèţůæİěãĂĆæŕŤåęĆïijŽ

```
>>> f = open('file.dat')
>>> f
<_io.TextIOWrapper name='file.dat' mode='r' encoding='UTF-8'>
>>>
```

åęĆæđIJ __str__() æšąæIJL'èćńåőŽäźĽ'ïijŇéĆčäźĹåřšäijŽä¡£çŤÍ __repr__() æİěäžčæŻ£è¿ŞåĞžãĂĆ

äÿŁéİćçŽĎ format() æŰźæşŢçŽĎä¡£çŤÍçIJŃäÿŁåŐżå¿ĹæIJL'èűčïijŇæäijåijŔåŇŰäžčçăĄ {0.x} åŕźåžŤçŽĎæŸŕçňň1äÿłåŔĆæŢřçŽĎxåšđæĂğãĂĆ åŻăæ■d'ïijŇåIJÍäÿŃéİćçŽĎåĞ¡æŢřäÿ■ïijŇ0åőđéŹ self æIJňèžńïijŽ

```
def __repr__(self):
    return 'Pair({0.x!r}, {0.y!r})'.format(self)
```

ä¡IJäÿžè£Źçğ■åőđçŐřçŽĎäÿĂäÿłæŻ£äžčïijŇä¡äáź§åŔŕäžěä¡£çŤÍ % æŞ■ä¡IJçňęïijŇåřšåČŔäÿŃéİćè£ŹæăůïijŽ

```
def __repr__(self):
    return 'Pair(%r, %r)' % (self.x, self.y)
```

8.2 èĞłåőŽäźĽå■Ůçňęäÿšç ŽĎæäijåijŔåŇŰ

éŮőécŸ

ä¡ăæČşéĂŽè£Ğ format() åĞ¡æŢřåŠŇå■ŮçňęäÿšæŰźæşŢä¡£å¿ŮäÿĂäÿłåŕźèśąèČ¡æŤŕæŇĄèĞłåőŽäźI

èğčåĘşæŰźæąĹ

äÿžäžĘèĞłåőŽäźĽ'å■Ůçňęäÿšç ŽĎæäijåijŔåŇŰïijŇæĹŚäžňéIJĂèęĄåIJÍçśžäÿŁéİćåőŽäźĽ' __format__() æŰźæşŢãĂĆä¿ŃåęĆïijŽ

```
_formats = {
    'ymd' : '{d.year}-{d.month}-{d.day}',
    'mdy' : '{d.month}/{d.day}/{d.year}',
    'dmy' : '{d.day}/{d.month}/{d.year}'
    }

class Date:
```

```
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day

    def __format__(self, code):
        if code == '':
            code = 'ymd'
        fmt = _formats[code]
        return fmt.format(d=self)
```

çŐřåIJÍ `Date` çśžçŽĎåőđä¿ŃåŘŕäžěæŤŕæŇĄæäijåijŔåŇŰæŞ■ä¡IJäžĘïijŇåęĆåŘŇäÿŃéÍćè£Źæăůïijž

```
>>> d = Date(2012, 12, 21)
>>> format(d)
'2012-12-21'
>>> format(d, 'mdy')
'12/21/2012'
>>> 'The date is {:ymd}'.format(d)
'The date is 2012-12-21'
>>> 'The date is {:mdy}'.format(d)
'The date is 12/21/2012'
>>>
```

èőÍèőž

`__format__()` æŰźæşŢçżŹPythonçŽĎå■ŮçňęäÿšæäijåijŔåŇŰåŁ§èČ¡æŘŘä¿ŻäžĘäÿĂäÿłéŠľå■ŘãĂ
è£ŹéĞŇéIJĂèęĄçIĂéĞ■åijžèřČçŽĎæŸŕæäijåijŔåŇŰäžčçăĄçŽĎèğčæđŘåůěä¡IJåőŇåĚÍçŤśçśžèĞłåůśåĘşåőŽ
ä¿ŃåęĆïijŇåŔĆèĂČäÿŃéÍćæÍèèĞł `datetime` æĺąåİŮäÿ■çŽĎäžčçăĄïijž

```
>>> from datetime import date
>>> d = date(2012, 12, 21)
>>> format(d)
'2012-12-21'
>>> format(d,'%A, %B %d, %Y')
'Friday, December 21, 2012'
>>> 'The end is {:%d %b %Y}. Goodbye'.format(d)
'The end is 21 Dec 2012. Goodbye'
>>>
```

åŕźäžŐåĘĚç¡őçśžåđŃçŽĎæäijåijŔåŇŰæIJL'äÿĂäžŻæăĞåĞĘçŽĎçžęåőŽãĂĆ
åŘŕäžěåŔĆèĂČ stringæĺąåİŮæŰĞæąč èŕť'æŸŐãĂĆ

8.3 èőľåŕźèśąæŤŕæŇĄäÿŁäÿŃæŰĞçőąçŘĘå■Ŕèőő

éŮőécŸ

ä¡ăæČşèőľä¡ăçŽĎåŕźèśąæŤŕæŇĄäÿŁäÿŃæŰĞçőąçŘĘå■Ŕèőő(withèŕ■åŔě)ãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘèőľäÿĂäÿłåŕźèśąåĚijåőź `with` èŕ■åŔěïijŇä¡ăéIJĂèęĄåőđçŐř `__enter__()` åŠŇ `__exit__()` æŰźæşŢãĂĆ ä¿ŃåęĆïijŇèĂČèŹŚåęĆäÿŃçŽĎäÿĂäÿłçśżïijŇåőČèČ¡äÿžæĹŚäžňåĹŻåžžäÿ

```
from socket import socket, AF_INET, SOCK_STREAM

class LazyConnection:
    def __init__(self, address, family=AF_INET, type=SOCK_STREAM):
        self.address = address
        self.family = family
        self.type = type
        self.sock = None

    def __enter__(self):
        if self.sock is not None:
            raise RuntimeError('Already connected')
        self.sock = socket(self.family, self.type)
        self.sock.connect(self.address)
        return self.sock

    def __exit__(self, exc_ty, exc_val, tb):
        self.sock.close()
        self.sock = None
```

è£ŹäÿłçśżçŽĎåĚşéŤőçL'źçĆźåIJĺäžŐåőČèąĺçď'žäžĘäÿĂäÿłç¡ŚçżIJè£đæŐěïijŇä¡ĘæŸŕåĹİåğŃåŇŰçŽĎa
è£đæŐěçŽĎåžžçńŃåŠŇåĚşéŮ■æŸŕä¡£çŤĺ `with` èŕ■åŔèĞłåŁĺåőŇæĹŘçŽĎïijŇä¿ŃåęĆïijŽ

```
from functools import partial

conn = LazyConnection(('www.python.org', 80))
# Connection closed
with conn as s:
    # conn.__enter__() executes: connection open
    s.send(b'GET /index.html HTTP/1.0\r\n')
    s.send(b'Host: www.python.org\r\n')
    s.send(b'\r\n')
    resp = b''.join(iter(partial(s.recv, 8192), b''))
    # conn.__exit__() executes: connection closed
```

èőĺèőž

çijŰåĘŹäÿŁäÿŃæŰĞçőąçŘĘåŹĺçŽĎäÿżèęĄåŐ§çŘĘæŸŕä¡ăçŽĎäžčçăĄäijŽæŤ¿åĹř `with` èŕ■åŔěåİŮäÿ■æL'ğèąŇãĂĆ å¡ŞåĞžçŐř `with` èŕ■åŔęçŽĎæŮűåĂŹïijŇářźèśąçŽĎ

__enter__() æŰźæşŢèćńèğęåŔŚïijŇ åőČè£ŤåŻđçŽĎåĂij(åęĆæđIJæIJL'çŽĎèŕÍ)äijŽèćńèţŃåĂijçżŹ as åčřæŸŐçŽĎåŔŸéĞŔãĂĆçĎűåŘŐïijŇwith èŕ■åŘěåİŮéĞŇéİćçŽĎäžčçăĄåijĂågŇæL'ğèąŇãĂĆ æIJĂåŘŐïijŇ__exit__() æŰźæşŢèćńèğęåŔŚè£ŻèąŇæÿĚçŘĘåůěä¡IJãĂĆ

äÿ■çőą with äžčçăĄåİŮäÿ■åŔŚçŢ§äžĂäźĹïijŇäÿŁéİćçŽĎæŐğåĹűæţĄéČ¡äijŽæL'ğèąŇåőŇïijŇåřśçőŮä äžŇåőđäÿŁïijŇ__exit__() æŰźæşŢçŽĎçňňäÿL'äÿłåŔĆæŢřåŇĚåŘńäžĘåijĆåÿÿçśżåđŇãĂĄåijĆåÿÿåĂijåŠ __exit__() æŰźæşŢèČ¡èĞłåůśåĘşåőŽæĂŐæăůåĹl'çŤÍè£ŹäÿłåijĆåÿÿä£ąæĄřïijŇæĹŰèĂĚå£¡çŢěåőČåźűè åęĆæđIJ __exit__() è£ŤåŻđ True ïijŇéĆčäźĹåijĆåÿÿäijŽèćńæÿĚçl'žïijŇåřśåě¡åČŔäžĂäźĹéČ¡æšąåŔŚçŤ with èŕ■åŘěåŘŐéİćçŽĎçÍŇåžŘçżğçż■åIJÍæ■čåÿÿæL'ğèąŇãĂĆ

è£ŸæIJL'äÿĂäÿłçżĘèŁĆéŮőécŸåřśæŸŕ LazyConnection çśżæŸŕåŘęåĚĄèőÿåd'Žäÿł with èŕ■åŘěæİěåţŇåěŮä¡£çŤÍè£đæŐěãĂĆ å¿ĹæŸ¿çĎűïijŇäÿŁéİćçŽĎåőŽäźL'äÿ■äÿĂæňąåŔŁèČ¡åĚĄèőÿäÿĂäÿłsocketè£đæŐěïijŇåęĆæđIJæ■čåIJÍä¡£çŤ with èŕ■åŘěïijŇ åřśäijŽäžğçŢ§äÿĂäÿłåijĆåÿÿäžĘãĂĆäÿ■è£Ğä¡ăåŔŕäžěåČŔäÿŇéİćè£ŹæăůäŁőæŤźäÿŇäÿŁé

```
from socket import socket, AF_INET, SOCK_STREAM

class LazyConnection:
    def __init__(self, address, family=AF_INET, type=SOCK_STREAM):
        self.address = address
        self.family = family
        self.type = type
        self.connections = []

    def __enter__(self):
        sock = socket(self.family, self.type)
        sock.connect(self.address)
        self.connections.append(sock)
        return sock

    def __exit__(self, exc_ty, exc_val, tb):
        self.connections.pop().close()

# Example use
from functools import partial

conn = LazyConnection(('www.python.org', 80))
with conn as s1:
    pass
    with conn as s2:
        pass
        # s1 and s2 are independent sockets
```

åIJÍçňňäžŇäÿłçL'ĹæIJňäÿ■ïijŇLazyConnection çśżåŔŕäžěèćńçIJŃåĄŽæŸŕæ§Řäÿłè£đæŐěåůěåŐĆã æŕŔæňą __enter__() æŰźæşŢæL'ğèąŇçŽĎæŮűåĂŹïijŇåőČåd'■åĹűåĹŽåžžäÿĂäÿłæŰřçŽĎè£đæŐěåźűåi __exit__() æŰźæşŢçőĂå■ŢçŽĎäžŐæăĹäÿ■åijźåĞžæIJĂåŘŐäÿĂäÿłè£đæŐěåźűåĚşéŮ■åőČãĂĆ è£ŹéĞŇçĺ■å¿őæIJL'çĆźéŽ¿çŘĘèğčïijŇäÿ■è£ĞåőČèČ¡åĚĄèőÿåţŇåěŮä¡£çŤÍ with èŕ■åŘěåĹŽåžžåd'Žäÿłè£đæŐěïijŇåřśåęĆäÿŁéİćæijŢçd'žçŽĎéĆčæăůãĂĆ

åIJÍéIJĂèęĄçőąçŘĘäÿĂäžŻèţĎæžŘæŕŤåęĆæŰĞäžűãĂĄç¡ŚçżIJè£đæŐěåŠŇéŢĄçŽĎçijŰçĺŇçŐŕåćĆäÿ■ è£ŹäžŻèţĎæžŘçŽĎäÿĂäÿłäÿżèęĄçL'źå¿ĄæŸŕåőČäžňå£Ěéążèćńæl'ŇåŁĺçŽĎåĚşéŮ■æĹŰéĞŁæŢ¿æİěçąőä£

ä¿ÑåęĆïijÑåęĆædIJä¡ăèŕůæśĆäžĘäÿĂäÿłéŤĄïijŇéĆčäźĹä¡ăå£Ěéążçąőä£ÌäźÑåŘŐéĞŁæŤ¿äžĘåőČïijÑåŘęåĹ
éĂŽè£ĞåőđçŐř `__enter__()` åŠŇ `__exit__()` æŰźæşŢåźűä¡£çŤÍ `with`
èŕ■åŘěåŘŕäžěå¿Ĺåőźæ ŸŞçŽĎéĄ£åĚ■è£ŹäžŻéŮőécŸïijŇ åŻääÿž `__exit__()`
æŰźæşŢåŘŕäžěèőľ'ä¡ăæŮăéIJĂæŃĚå£Čè£ŹäžŻäžĘãĂĆ

åIJÍ `contextmanager` æĺąåİŮäÿ■æIJL'äÿĂäÿłæăĞåĞĘçŽĎäÿŁäÿŃæŰĞçőąçŘĘæŰźæąĹæĺąæİ£ïijÑå
åŘŇæŮűåIJÍ12.6åřŔèŁĆäÿ■è£ŸæIJL'äÿĂäÿłåŕźæIJňèŁĆçd'žä¿ŃçÍŃåžŔçŽĎçž£çÍŃåőL'åĚĺçŽĎä£őæŤźçL'Ĺ

8.4 åĹŻåžžåď'ğéĞŘåŕźèśąæŮűèŁĆçIJĄåĘĚå■ŸæŰźæşŢ

éŮőécŸ

ä¡ăçŽĎçÍŃåžŔèęĄåĹŻåžžåď'ğéĞŘ(åŔŕèČ¡äÿŁçŹ¿äÿĞ)çŽĎåŕźèśąïijÑåŕijèĞť'å■ăçŤÍå¿Ĺåď'ğçŽĎåĘĚå■Ÿ

èğčåĘşæŰźæąĹ

åŕźäžŐäÿżèęĄæŸŕçŤÍæÌěå¡ŞæĹŘçőĂå■ŢçŽĎæŢřæ■őçżŞædĎçŽĎçśżèĂŇèÍĂïijÑä¡ăåŔŕäžěéĂŽè£ĞçżŹ
`__slots__` åśđæĂğæÌěædĄåď'ğçŽĎåĞŘåřŠåőđä¿ŃæĽ'Ăå■ăçŽĎåĘĚå■ŸãĂĆæŕŤåęĆïijŽ

```
class Date:
    __slots__ = ['year', 'month', 'day']
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day
```

å¡Şä¡ăåőŽäźĽ' `__slots__` åŘŐïijÑPythonåřśäijŽäÿžåőđä¿Ńä¡£çŤÍäÿĂçğ■æŻť'åŁăçť'ğåĞŠçŽĎåĘĚéČ
åőđä¿ŃéĂŽè£ĞäÿĂäÿłå¿ĹåřŘçŽĎåŽžåőŽåď'ğåřŘçŽĎæŢřçżĎæÌěædĎåžžïijÑèĂŇäÿ■æŸŕäÿžæŕŔäÿłåőđä¿Ń
åIJÍ `__slots__` äÿ■åĹŮåĞžçŽĎåśđæĂğåŘ■åIJÍåĘĚéČÍèćńæŸăåřĎåĹřè£ŹäÿłæŢřçżĎçŽĎæŇĞåőŽåřŘæăČ
ä¡£çŤÍslotsäÿĂäÿłäÿ■åě¡çŽĎåIJřæŰźåŕśæŸŕæĹŠäžňäÿ■èČ¡åĘ■çżŹåőđä¿ŃæůžåŀăæŰřçŽĎåśđæĂğäžĘïijÑåĚ
`__slots__` äÿ■åőŽäźĽçŽĎéĆčäžŽåśđæĂğåŘ■ãĂĆ

èőÍèőž

ä¡£çŤÍslotsåŘŐèŁĆçIJĄçŽĎåĘĚå■ŸäijŽèů§å■ŸåĆÍåśđæĂğçŽĎæŢřéĞŔåŠŇçśżåđŃæIJL'åĚşãĂĆ
äÿ■è£ĞïijÑäÿĂèĹňæÌěèőšïijÑä¡£çŤÍåĹřçŽĎåĘĚå■ŸæĂżéĞŔåŠŇåřĘæŢřæ■őå■ŸåĆÍåIJÍäÿĂäÿłåĚČçżĎäÿ■å
äÿžäžĘçżŹä¡ăäÿĂäÿłçŻť'èğĆèőď'èŕĘïijÑåĄĞèő¿ä¡ăäÿ■ä¡£çŤÍslotsçŻť'æŐěå■ŸåĆÍäÿĂäÿłDateåőđä¿ŃïijŇ
åIJÍ64ä¡■çŽĎPythonäÿŁéİćèęĄå■ăçŤÍ428å■ŮèŁĆïijÑèĂŇåęĆædIJä¡£çŤÍäžĘslotsïijÑåĘĚå■Ÿå■ăçŤÍäÿŃéŹ
åęĆædIJçÍŃåžŔäÿ■éIJĂèęĄåŘŇæŮűåĹŻåžžåď'ğéĞŘçŽĎæŮěæIJ§åőđä¿ŃïijÑéČčäźĹè£ŹäÿłåřśèČ¡ædĄåď'ğ

åř¡çőąslotsçIJŃäÿŁåŐżæŸŕäÿĂäÿłå¿ĹæIJL'çŤÍçŽĎçL'źæĂğïijÑå¿Ĺåď'ŽæŮűåĂŹä¡ăè£ŸæŸŕå¿ŮåĞŔåř
PythonçŽĎå¿Ĺåď'ŽçL'źæĂğéČ¡ä¿ÌètŰäžŐæŹőéĂŽçŽĎå§žäžŐå■ŮåĚÿçŽĎåőđçŐřãĂĆ
åŘęåď'ŰïijÑåőŽäźĽäžĘslotsåŘŐçŽĎçśżäÿ■åĘ■æŤŕæŇĄäÿĂäžŻæŹőéĂŽçśżçL'źæĂğäžĘïijÑæŕŤåęĆåď'Žçżğ
åď'ğåď'ŽæŢřæČĚåĘţäÿŃïijÑä¡ăäžŤèŕěåŘłåIJÍéČčäžŽçżŔåÿÿèćńä¡£çŤÍåĹřçŽĎçŤÍä¡IJæŢřæ■őçżŞædĎçŽĎçś
(æŕŤåęĆåIJÍçÍŃåžŔäÿ■éIJĂèęĄåĹŻåžžæ§Răÿłçśżç ŽĎåĞăçŹ¿äÿĞäÿłåőđä¿Ńåŕźèśą)ãĂĆ

åĚşäžŐ `__slots__` çŽĎäÿĂäÿłåÿÿèğĄèŕŕåŇžæŸŕåőČåŔŕäžĕä¡IJäÿžäÿĂäÿłåřĄèčĚåůĕåĚůæİĕéŸšæ■ć
åř¡çőąä¡£çŤĺslotsåŔŕäžĕè¿¿åĹřè£ŹæăůçŽĎçŽőçŽĎïijŇä¡ĘæŸŕè£Źäÿłåźűäÿ■æŸŕåőČçŽĎåĹİèąůãĂĆ
`__slots__` æŻťåď˙ŽçŽĎæŸŕçŤĺæİĕä¡IJäÿžäÿĂäÿłåĘĚå■ŸäijŸåŇŰåůĕåĚůãĂĆ

8.5 åIJĺçśżäÿ■åřĄèčĚåśđæĂğåŘ■

éŮőéćŸ

ä¡ăæČşåřĄèčĚçśżçŽĎåőđä¿ŃäÿŁéİćçŽĎâĂIJçğĄæIJL’âĂİæŢřæ■őïijŇä¡ĘæŸŕPythonèŕ■èĺĂåźűæšąæIJ

èğčåĘşæŰźæąĹ

PythonçĺŃåžŔåŚŸäÿ■åŐżä¿İètŰèŕ■èĺĂçL’źæĂğåŐżåřĄèčĚæŢřæ■őïijŇèĂŇæŸŕéĂŽè£ĞéĄţå¿łäÿĂåőŽ
çňňäÿĂäÿłçžęåőŽæŸŕäżżä¡Ţäžĕå■ŢäÿŃåĹŠçž£_åijĂåď˙çŽĎåŘ■å■ŮéČ¡åžŤèŕěæŸŕåĘĚéČĺåőđçŐřãĂĆærŤå

```
class A:
    def __init__(self):
        self._internal = 0 # An internal attribute
        self.public = 1 # A public attribute

    def public_method(self):
        '''
        A public method
        '''
        pass

    def _internal_method(self):
        pass
```

Pythonåźűäÿ■äijŽçIJ§çŽĎéŸżæ■ćåĹńäžžèő£éŮőåĘĚéČĺåŘ■çğřãĂĆä¡ĘæŸŕåęĆæđIJä¡ăè£ŹäźĹåĄŽèĆŕ
åŘŇæŮűè£ŸèęĄæşĺæĎŔåĹřïijŇä¡£çŤĺäÿŃåĹŠçž£åijĂåď˙çŽĎçžęåőŽåŘŇæăůéĂĆçŤĺäžŐæĺąåİŮåŘ■åŠŇæ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăçIJŃåĹræ§ŘäÿłæĺąåİŮåŘ■äžĕå■ŢäÿŃåĹŠçž£åijĂåď˙ť(æŕŤåęĆ_socket)ïijŇéĆčåőČåřš
çśżäijijçŽĎïijŇæĺąåİŮçžğåĹńåĞ¡æŢřæŕŤåęĆ `sys._getframe()`
åIJĺä¡£çŤĺçŽĎæŮűåĂŹåřšå¿ŮåŁăăĂ■åřŔå£ČäžĘãĂĆ

ä¡ăè£ŸåŔŕèČ¡äijŽéĄĞåĹřåIJĺçśżåőŽäźL’äÿ■ä¡£çŤĺäÿd’äÿłäÿŃåĹŠçž£(__)åijĂåď˙çŽĎåŚ¡åŘ■ãĂĆærŤå

```
class B:
    def __init__(self):
        self.__private = 0

    def __private_method(self):
        pass

    def public_method(self):
        pass
        self.__private_method()
```

ä¡£çŤĺåŔŇäÿŇåĹŠçž£åijĂåğŃäijŽåŕijèĞť'èő£éŮőåŘ■çğřåŔŸæĹŘåĚűäžŰå¡ćåijŔãĂĆ
æŕŤåęĆïijŇåIJĺåL'■éÍćçŽĎçśżBäÿ■ïijŇçğĄæIJL'åśđæĂğäijŽècńåĹĘåĹńéĞ■åŚ¡åŘ■äÿž
_B__private åŠŇ _B__private_method ãĂĆ è£ŹæŮűåĂŹä¡ăåŔŕèČ¡äijŽéŮőè£ŹæăúéĞ■åŚ¡åŘ■çŽĎ

```
class C(B):
    def __init__(self):
        super().__init__()
        self.__private = 1 # Does not override B.__private

    # Does not override B.__private_method()
    def __private_method(self):
        pass
```

è£ŹéĞŇïijŇçğĄæIJL'åŘ■çğř __private åŠŇ __private_method
èćńéĞ■åŚ¡åŘ■äÿž _C__private åŠŇ _C__private_method
ïijŇè£ŹäÿłèůšçĹűçśżBäÿ■çŽĎåŘ■çğřæŸŕåőŇåĚĺäÿ■åŘŇçŽĎãĂĆ

èőÍèőž

äÿŁéÍćæŔŘåĹřæIJL'äÿd'çğ■äÿ■åŘŇçŽĎçijŰçăĄçžęåőŽ(å■ŢäÿŇåĹŠçž£åŠŇåŔŇäÿŇåĹŠçž£)æÌěåŚ¡åŘ
åd'ğåd'ŽæŢřèĂŇèÍĂïijŇä¡ăåžŤèŕěèől'ä¡ăçŽĎéÍđåĚňåĚśåŘ■çğřäžěå■ŢäÿŇåĹŠçž£åijĂåd'ť'ãĂĆä¡ĘæŸŕïijŇåę
åźűäÿŤæIJL'äžŻåĘĚéČĺåśđæĂğåžŤèŕěåIJĺå■Řçśżäÿ■éŽŘèŮŘètůæÌěïijŇéĆčäźĹæL'■èĂČèŹŚä¡£çŤĺåŔŇäÿŇ

è£ŸæIJL'äÿĂçĆźèęĄæşĺæĎŔçŽĎæŸŕïijŇæIJL'æŰűåĂŹä¡ăåőŽäźL'çŽĎäÿĂäÿłåŔŸéĞŔåŠŇæ§Řäÿłä£Ię

```
lambda_ = 2.0 # Trailing _ to avoid clash with lambda keyword
```

è£ŹéĞŇæĹŚäžňåźűäÿ■ä¡£çŤĺå■ŢäÿŇåĹŠçž£åL'■çijĂçŽĎåŐ§åŽăæŸŕåőČéĄ£åĚ■èŕŕèğčåőČçŽĎä¡£çŤĺ
(åęĆä¡£çŤĺå■ŢäÿŇåĹŠçž£åL'■çijĂçŽĎçŽőçŽĎæŸŕäÿžäžĘéŸšæ■ćåŚ¡åŘ■åĘšçłĄèĂŇäÿ■æŸŕæŇĞæŸŐè£Ź
éĂŽè£Ğä¡£çŤĺå■ŢäÿŇåĹŠçž£åŘŐçijĂåŔŕäžěèğčåĘşè£ŹäÿłéŮőécŸãĂĆ

8.6 åĹŻåžžåŔŕçőąçŘĘçŽĎåśđæĂğ

éŮőécŸ

ä¡ăæČşçżŹæ§Řäÿłåőđä¿ŃattributeåćđåŁăéŹd'èő£éŮőäÿŐä£őæŤźäźŇåd'ŰçŽĎåĚűäžŰåd'ĎçŘĘéĂžè¿Śi

èğčåĘşæŰźæąĹ

èĞłåőŽäźL'æ§ŘäÿłåśđæĂğçŽĎäÿĂçğ■çőĂå■ŢæŰźæşŢæŸŕåřĘåőČåőŽäźL'äÿžäÿĂäÿłpropertyãĂĆ
ä¿ŇåęĆïijŇäÿŇéÍćçŽĎäžčçăĄåőŽäźL'äžĘäÿĂäÿłpropertyïijŇåćđåŁăářźäÿĂäÿłåśđæĂğçőĂå■ŢçŽĎçśżåđŇæ

```
class Person:
    def __init__(self, first_name):
        self.first_name = first_name

    # Getter function
```

```
    @property
    def first_name(self):
        return self._first_name

    # Setter function
    @first_name.setter
    def first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._first_name = value

    # Deleter function (optional)
    @first_name.deleter
    def first_name(self):
        raise AttributeError("Can't delete attribute")
```

äÿŁè£řäžčçăĄäÿ■æIJL'äÿL'äÿłçŻÿåĚşèĄŤçŽĎæŰźæşŢïijŇè£ŹäÿL'äÿłæŰźæşŢçŽĎåŘ■å■ŮéČ¡å£Ěéąžäÿçňňäÿ Ăäÿłæ Ű źæşŢæŸřäÿ Ăäÿł getter åĞ¡æŢřïijŇåőČä¡£å¿Ů first_name æĹŘäÿžäÿ Ăäÿłåśđæ Ăğã ĂĆ åĚűäżŰäÿd'äÿłæŰźæşŢçżŹ first_name åśđæ Ăğæůžå Łäážę setter åŠŇ deleter åĞ¡æŢřã ĂĆ éIJĂèęĄåijžèřČçŽĎæŸřåŘłæIJL'åIJÍ first_name åśđæ Ăğèćńå ĹŻåžžåŘŐïijŇ åŘŐéİćçŽĎäÿd'äÿłèčĚéěřåŹÍ @first_name.setter åŠŇ @first_name.deleter æL'■èČ¡èćńåőŽäźL'ã ĂĆ

propertyçŽĎäÿ ĂäÿłåĕşéŤőçL'źå¿ĄæŸřåőČçIJŃäÿŁåŐżèů§æŹőéĂŽçŽĎattributeæšąäžĂäźĹäÿd'æăůïijŇä¡ĘæŸřèő£éŮőåőČçŽĎæŮűåĂŹäijŽèĞłåŁĺèğęåŘŚ getter ãĂĄsetter åŠŇ deleter æŰźæşŢã ĂĆä¿ŃåęĆïijŽ

```
>>> a = Person('Guido')
>>> a.first_name # Calls the getter
'Guido'
>>> a.first_name = 42 # Calls the setter
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "prop.py", line 14, in first_name
        raise TypeError('Expected a string')
TypeError: Expected a string
>>> del a.first_name
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: can`t delete attribute
>>>
```

åIJÍåőđçŐřäÿ ĂäÿłpropertyçŽĎæŮűåĂŹïijŇåžŢåśĆæŢřæ■ő(åęĆæđIJæIJL'çŽĎèřİ)äż■çĎűéIJĂèęĄå■ŸåŒåŻăæ■d'ïijŇåIJÍgetåŠŇsetæŰźæşŢäÿ■ïijŇä¡ääijŽçIJŃåĹřářź _first_name åśđæ Ăğç ŽĎæŞ■ä¡IJïijŇè£Źäź§æŸřåőđéŹĚæŢřæ■őä£Ía■ŸçŽĎåIJřæŰźã ĂĆ åŘęåd'ŰïijŇä¡ăåŘřèČ¡è£ŸäijŽéŮőäÿžäżĂäźĹ __init__() æŰźæşŢäÿ■èő¿ç¡őäžę self.first_name èĂŇäÿ■æŸŕ self._first_name ã ĂĆ åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇæĹŚäžňåĹŻåžžäÿ Ăäÿłpropertyç ŽĎçŽőçŽĎåřśæŸřåIJÍèő¿ç¡őattributeçŽĎæŮűåĂŹåŻăæ■d'ïijŇä¡ăåŘřèČ¡æČşåIJÍåĹÍåğŃåŇŰçŽĎæŮűåĂŹäź§è£ŻèąŇè£Źçğ■çśžåđŃæčÄæ§ěã ĂĆéĂŽè£ĞèŐ¿ç¡self.first_name ïijŇèĞłåŁĺèřČçŤÍ setter æŰźæşŢïijŇ

è£Źäÿ łæŰźæşŢéĞŇéİćäijŽè£Żèą ŇåŔĆæŢřçŽĎæčĂæ§ěïijŇåŘęåĹŹåřśæŸŕçŻť æŐěèő£éŮő self._first_name äžĘãĂĆ

è£ŸèČ¡åIJÍåůšå■ŸåIJÍçŽĎgetåŠŇsetæŰźæşŢå§žçąĂäÿŁáőŽäźL'propertyãĂĆä¿ŃåęĆïijŽ

```
class Person:
    def __init__(self, first_name):
        self.set_first_name(first_name)

    # Getter function
    def get_first_name(self):
        return self._first_name

    # Setter function
    def set_first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._first_name = value

    # Deleter function (optional)
    def del_first_name(self):
        raise AttributeError("Can't delete attribute")

    # Make a property from existing get/set methods
    name = property(get_first_name, set_first_name, del_first_name)
```

èőÍèőž

äÿĂäÿłpropertyåśđæĂğåĚűåőđåřśæŸŕäÿĂçşżåĹŮçŻÿåĚşçżŚáőŽæŰźæşŢçŽĎéŻĘåŘĹãĂĆåęĆæđIJä¡ăå
åřśäijŽåŘŚçŐřpropertyæIJňèžńçŽĎfgetãĂĄfsetåŠŇfdelåśđæĂğåřśæŸŕçśžéĞŇéİćçŽĎæŹőéĂŽæŰźæşŢãĂĆ

```
>>> Person.first_name.fget
<function Person.first_name at 0x1006a60e0>
>>> Person.first_name.fset
<function Person.first_name at 0x1006a6170>
>>> Person.first_name.fdel
<function Person.first_name at 0x1006a62e0>
>>>
```

éĂŽåÿÿæİèèőšïijŇä¡ăäÿ■äijŽçŻť æŐěåŘŰèřČçŤÍfgetæĹŰèĂĚfsetïijŇåőČäžňäijŽåIJÍèő£éŮőpropertyçŽ

åŘłæIJL'å¡Şä¡ăçąőåőđéIJĂèęĄåŕźattributeæL'ğèąŇåĚűäžŰéćİåd'ŰçŽĎæŞ■ä¡IJçŽĎæŮűåĂŹæL'■åžŤèřě
æIJL'æŮűåĂŹäÿĂäžŻäżŐåĚűäžŰçijŰçÍŃèí■èÍĂ(æŕŤåęĆJava)è£ĞæİęçŽĎçÍŃåžŘåŚŸæĂżèőd'äÿžæL'ĂæIJL
æL'Ăäżěäž Űäžňèőd'äÿžäżčçăĄåžŤèřěåČŘäÿŃéİćè£ŹæăůåĘŹïijŽ

```
class Person:
    def __init__(self, first_name):
        self.first_name = first_name

    @property
```

```
    def first_name(self):
        return self._first_name

    @first_name.setter
    def first_name(self, value):
        self._first_name = value
```

äÿ■èęĄåĘŹè£Źçğ■æšąæIJL’åĄŽäżżä¡ŢåĚűäżŰéćÍåď’ŰæŞ■ä¡IJçŽĎpropertyãĂĆ
éęŰåĚĹïijŇåőČäijŽèől’ä¡äçŽĎäżčçăĄåŔŸå¿Ůå¿ĹèĞČèĆ£ïijŇåźűäÿŤè£ŸäijŽè£ůæČŚéŸĚèŕżèĂĚãĂĆ
åĚűæňąïijŇåőČè£ŸäijŽèől’ä¡äçŽĎçÍŃåžŔè£ŔèąŇètůæİěåŔŸæĚćå¿Ĺåď’ŽãĂĆ
æIJĂåŘŐïijŇè£ŹæăůçŽĎèő¿èőąåźűæšąæIJL’åÿęæİěäżżä¡ŢçŽĎåě¡åď’ĎãĂĆ
çL’źåĹńæŸŕå¡Şä¡ăäżěåŘŐæČşçżŹæŹőéĂŽattributeèő£éŮőæůżåŁăéćÍåď’ŰçŽĎåď’ĎçŘĘéĂżè¿ŚçŽĎæŮűåĂŹ
ä¡ääŔŕäżěåřĘåőČåŔŸæĹŘäÿĂäÿłpropertyèĂŇæŮăéIJĂæŤźåŔŸåŐ§æİěçŽĎäżčçăĄãĂĆ
åŽääÿžèő£éŮőattributeçŽĎäżčçăĄè£ŸæŸŕä£ÍæŇĄåŐ§æăůãĂĆ

Propertiesè£ŸæŸŕäÿĂçğ■åőŽäźL’åŁĹæĂĄèőąçŐůattributeçŽĎæŰźæşŢãĂĆ
è£Źçğ■çśżådŇçŽĎattributesåźűäÿ■äijŽèćńåőđéŹĚçŽĎå■ŸåĆÍïijŇèĂŇæŸŕåIJĺéIJĂèęĄçŽĎæŮűåĂŹèőąçőŮ

```
import math
class Circle:
    def __init__(self, radius):
        self.radius = radius

    @property
    def area(self):
        return math.pi * self.radius ** 2

    @property
    def diameter(self):
        return self.radius ** 2

    @property
    def perimeter(self):
        return 2 * math.pi * self.radius
```

åIJÍè£ŹéĞŇïijŇæĹŚäžňéĂŽè£Ğä¡£çŤÍpropertiesïijŇářĘæL’ĂæIJL’çŽĎèő£éŮőæŐěåŔčå¡ćåijŔçż§äÿĂèţ
ářźå■Łå¿ĎãĂĄçŻt’å¿ĎãĂĄåŚĺéŢ£åŠŇéÍćçğŕçŽĎèő£éŮőéČ¡æŸŕéĂŽè£ĞåšđæĂğèő£éŮőïijŇářśèů§èő£éŮőç
åęĆæđIJäÿ■è£ŹæăůåĄŽçŽĎèŕÍïijŇéĆčäźĹářśèęĄåIJÍäžčçăĄäÿ■æůůåŘĹä¡£çŤÍçőĂå■ŢåšđæĂğèő£éŮőåŠŇæ
äÿŇéÍćæŸŕä¡£çŤÍçŽĎåőđä¿ŇïijŽ

```
>>> c = Circle(4.0)
>>> c.radius
4.0
>>> c.area  # Notice lack of ()
50.26548245743669
>>> c.perimeter  # Notice lack of ()
25.132741228718345
>>>
```

ář¡çőąpropertiesåŔŕäžěåőđçŐřäijŸéŻĚçŽĎçijŰçÍŃæŐěåŔčïijŇä¡ĘæIJL’äžŽæŮűåĂŹä¡ăè£ŸæŸŕäijŽæČ

```
>>> p = Person('Guido')
>>> p.get_first_name()
'Guido'
>>> p.set_first_name('Larry')
>>>
```

è£Źçğ■æČĚåĘţçŽĎåĞžçŐřéĂŽåÿÿæŸŕåŽääÿžPythonäžčçăĄèćńéŻĘæĹŘåĹřäÿĂäÿłåď ğåđŇå§žçąĂåźş
ä¿ŇåęĆïijŇæIJL’åŔŕèČ¡æŸŕäÿĂäÿłPythonçśžåĞĘåď ĞåŁăåĚěåĹřäÿĂäÿłå§žäžŐè£IJçÍŃè£ĞçÍŃèřČçŤÍçŽĎåd
è£Źçğ■æČĚåĘţäÿŃïijŇçŻt’æŐěä¡£çŤÍget/setæŰźæşŢ(æŹőéĂŽæŰźæşŢèřČçŤÍ)èĂŇäÿ■æŸŕpropertyæĹŰèőÿ

æIJĂåŘŐäÿĂçĆźïijŇäÿ■èęĄåČŔäÿŃéİćè£ŹæăůåĘŹæIJL’åď ğéĞŔéĞ■åď ■äžčçăĄçŽĎpropertyåőŽäźL’ï

```
class Person:
    def __init__(self, first_name, last_name):
        self.first_name = first_name
        self.last_name = last_name

    @property
    def first_name(self):
        return self._first_name

    @first_name.setter
    def first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._first_name = value

    # Repeated property code, but for a different name (bad!)
    @property
    def last_name(self):
        return self._last_name

    @last_name.setter
    def last_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._last_name = value
```

éĞ■åď ■äžčçăĄäijŽåŕijèĞt’èĞČèĆ£ãĂĄæŸŞåĞžéŤŹåŠŇäÿŚéŹŃçŽĎçÍŃåžŔãĂĆåě¡æűĹæĄŕæŸŕïijŇéĂ
åŔŕäžěåŔĆèĂČ8.9åŠŇ9.21åřŔèŁĆçŽĎåĘĚåőźãĂĆ

8.7 èřČçŤÍçĹűçśžæŰźæşŢ

éŮőécŸ

ä¡ăæČşåIJÍå■Řçśžäÿ■èřČçŤÍçĹűçśžçŽĎæ§ŘäÿłåůšçżŕèćńèęĘçŻŰçŽĎæŰźæşŢãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘèřČçŤÍçĹűçśż(èűĚçśż)çŽĎäÿĂäÿłæŰźæşŢïijŇåŔŕäžěä¡£çŤÍ super() åĞ¡æŢřïijŇærŤåęĆïijŽ

```
class A:
    def spam(self):
        print('A.spam')

class B(A):
    def spam(self):
        print('B.spam')
        super().spam()  # Call parent spam()
```

super() åĞ¡æŢřçŽĎäÿĂäÿłåÿÿèğĄçŤÍæşŢæŸŕåIJÍ __init__() æŰźæşŢäÿ■çąőä£ÌçĹűçśżèćńæ■čçąőçŽĎåĹİåğŃåŇŰäžĘïijŽ

```
class A:
    def __init__(self):
        self.x = 0

class B(A):
    def __init__(self):
        super().__init__()
        self.y = 1
```

super() çŽĎåŔęåd'ŰäÿĂäÿłåÿÿèğĄçŤÍæşŢåĞžçŐřåIJÍèęĘçŽŰPythonçL'źæőŁæŰźæşŢçŽĎäżčçăĄäÿ

```
class Proxy:
    def __init__(self, obj):
        self._obj = obj

    # Delegate attribute lookup to internal obj
    def __getattr__(self, name):
        return getattr(self._obj, name)

    # Delegate attribute assignment
    def __setattr__(self, name, value):
        if name.startswith('_'):
            super().__setattr__(name, value) # Call original __
↪setattr__
        else:
            setattr(self._obj, name, value)
```

åIJÍäÿŁéİćäżčçăĄäÿ■ïijŇ__setattr__() çŽĎåőđçŐřåŇĚåŔńäÿĂäÿłåŘ■å■ŮæčĂæ§ěãĂĆ åęĆæđIJæ§ŘäÿłåśđæĂğåŘ■äžěäÿŇåĹŠçž£(_)åijĂåd't'ïijŇářśéĂŽè£Ğ super() èřČçŤÍåŐ§åğŇçŽĎ __setattr__() ïijŇ åŘęåĹŹçŽĎèŕÍářśåğŤæt'¿çżŹåĘĚéČÍçŽĎäżčçŘĘářźèšą self._obj åŐżåd'ĎçŘĘãĂĆ è£ŹçIJŃäÿŁåŐżæIJL'çĆźæĎŔæĂİïijŇåŽääÿžářśçőŮæšąæIJL'æŸ¿åijŔçŽĎæ super() äż■çĎűåŔŕäžěæIJL'æŢĹçŽĎåůěä¡IJãĂĆ

èőÍèőž

åőđéŹĚäýŁïijŇåď ğåőűåŕźäžŐåIJÍPythonäÿ■åęĆä¡Ţæ■čçąőä¡£çŤÍ `super()` åĞ¡æŢřæŹőéĄ■ç§ěäźŃçŤŽåřŠãĂĆ ä¡ăæIJĽæŮűåĂŹäijŽçIJŃåĹřåČŔäÿŃéİćè£ŹæăůçŻť æŐěèřČçŤÍçĹűçśžç

```
class Base:
    def __init__(self):
        print('Base.__init__')

class A(Base):
    def __init__(self):
        Base.__init__(self)
        print('A.__init__')
```

åř¡çőąåŕźäžŐåď ğéČÍåĹEäżčçăĄèĂŇèÍĂè£ŹäźĹåĄŽæšąäżĂäźĹéŮőécŸïijŇä¡ĘæŸŕåIJÍæŻť åď ■æİĆçŽĎ æŕŤåęĆïijŇèĂČèŹŠåęĆäÿŃçŽĎæČĚåĘţïijŽ

```
class Base:
    def __init__(self):
        print('Base.__init__')

class A(Base):
    def __init__(self):
        Base.__init__(self)
        print('A.__init__')

class B(Base):
    def __init__(self):
        Base.__init__(self)
        print('B.__init__')

class C(A,B):
    def __init__(self):
        A.__init__(self)
        B.__init__(self)
        print('C.__init__')
```

åęĆæđIJä¡ăè£ŘèąŇè£ŹæőţäżčçăĄåřšäijŽåŔŠçŐř `Base.__init__()` èćńèřČçŤÍäÿď æňąïijŇåęĆäÿŃæĽĂçď žïijŽ

```
>>> c = C()
Base.__init__
A.__init__
Base.__init__
B.__init__
C.__init__
>>>
```

åŔŕèČ¡äÿď æňąèřČçŤÍ `Base.__init__()` æšąäżĂäźĹåİŔåď ĎïijŇä¡ĘæIJĽæŮűåĂŹå■t'äÿ■æŸŕãĂĆ åŔęäÿĂæŰźéİćïijŇåĄĞèő¿ä¡ăåIJÍäżčçăĄäÿ■æ■ćæĹŘä¡£çŤÍ `super()` ïijŇçżŞæđIJåřšå¿ĹåőŇç¿ŐäžĘïijŽ

```
class Base:
    def __init__(self):
        print('Base.__init__')

class A(Base):
    def __init__(self):
        super().__init__()
        print('A.__init__')

class B(Base):
    def __init__(self):
        super().__init__()
        print('B.__init__')

class C(A,B):
    def __init__(self):
        super().__init__()  # Only one call to super() here
        print('C.__init__')
```

è£Řèą Ňè£ŹäÿłæŭřçĽĹæIJňåŘŐïijÑä¡ăäijŽåŔŚçŐřæŕŔäÿł `__init__()` æŰźæşŢåŔłäijŽèćńèřČçŤĺäÿĂæňąäžĘïijŽ

```
>>> c = C()
Base.__init__
B.__init__
A.__init__
C.__init__
>>>
```

äÿžäžĘåijĎæÿĚåőČçŽĎåŐ§çŘĘïijÑæĹŚäžňéIJĂèęĄèŁśçĆźæŮűéŮť'ègčéĞŁäÿŃPythonæŸŕåęĆä¡Ţåőđ åŕźäžŐä¡ăåőŽäźĽçŽĎæŕŔäÿĂäÿłçśžïijÑPythonäijŽèőąçőŮåĞžäÿĂäÿłæĽ'ĂèřŞçŽĎæŰźæşŢègčæđŘéąžåžŔ(Ñ è£ŹäÿłMROåĹŮèąĺåřśæŸŕäÿĂäÿłçőĂå■ŢçŽĎæĽ'ĂæIJĽ'å§žçśžçŽĎçž£æĂğéąžåžŔèąĺãĂĆä¿ŃåęĆïijŽ

```
>>> C.__mro__
(<class '__main__.C'>, <class '__main__.A'>, <class '__main__.B'>,
<class '__main__.Base'>, <class 'object'>)
>>>
```

äÿžäžĘåőđçŐřçżğæĽ'£ïijÑPythonäijŽåIJÍMROåĹŮèąĺäÿŁäžŐåůęåĹřåŘşåijĂåğŃæ§ěæĽ'¿å§žçśžïijÑçŻt'

èĂŇè£ŹäÿłMROåĹŮèąÍçŽĎæđĎéĂäæŸŕéĂŽè£ĞäÿĂäÿłC3çž£æĂğåŇŰçőŮæşŢæİěåőđçŐřçŽĎãĂĆ æĹŚäžňäÿ■åŐżæůśçl'ûè£ŹäÿłçőŮæşŢçŽĎæŢřå■ęåŐ§çŘĘïijÑåőĆåőđéŹĚäÿŁářśæŸŕåŘĹåźűæĽ'ĂæIJĽ'çĹűçş

- å■ŘçśżäijŽåĚĹäžŐçĹűçśżèćńæčĂæ§ě
- åd'ŽäÿłçĺűçśżäijŽæăźæ■őåőČäžňåIJĺåĹŮèąĺäÿ■çŽĎéąžåžŔèćńæčĂæ§ě
- åęĆæđIJåŕźäÿŃäÿĂäÿłçśżå■ŸåIJĺäÿd'äÿłåŘĹæşŢçŽĎéĂĽ'æŃl'ïijÑéĂĽ'æŃl'çňňäÿĂäÿłçĹűçśż

èĂĄåőđèŕt'ïijÑä¡ăæĽ'ĂèęAç§ěéĄŞçŽĎåřśæŸŕMROåĹŮèąĺäÿ■çŽĎçśżéąžåžŔäijŽèől'ä¡ăåőŽäźĽçŽĎäż

å¡Şä¡ää¡£çŤĺ `super()` åĞ¡æŢřæŮűïijÑPythonäijŽåIJÍMROåĹŮèąĺäÿŁçżğçż■æŘIJçt'ćäÿŃäÿĂäÿłçśžãĂ åŘłèęĄæŕŔäÿłéĞ■åőŽäźĽçŽĎæŰźæşŢçż§äÿĂä¡£çŤĺ `super()`

åźűåŔłèřČçŤĺåőČäÿĂæňąïijŇ éĆčäźĹæŐğåĹűæţĄæIJĂçżĹäijŽéĄ■åŐĘåőŇæŢt’äÿłM-ROåĹŮèąĺïijŇæŕŔäÿłæŰźæşŢäź§åŔłäijŽèćńèřČçŤĺäÿĂæňąãĂĆ è£Źäź§æŸŕäÿžäżĂäźĹåIJĺçňňäžŇäÿłä¿Ńå■Řäÿ■ä¡ääÿ■äijŽèřČçŤĺäÿd’æňą `Base.__init__()` çŽĎåŐ§åŻããĂĆ

`super()` æIJL’äÿłäżd’äžžåŘČæČŁçŽĎåIJřæŰźæŸŕåőČåźűäÿ■äÿĂåőŽåŐżæ§ěæL’¿æ§Řäÿłçśżå IJÍMRO ä¡ăçŤŽèĞşåŔŕäžěåIJÍäÿĂäÿłæšąæIJL’çŻť’æŐěçĹűçśżçŽĎçśżäÿ■ä¡£çŤĺåőČãĂĆä¿ŃåęĆïijŇèĂČèŹŚåęĆäÿŇè

```
class A:
    def spam(self):
        print('A.spam')
        super().spam()
```

åęĆæđIJä¡ăèŕŢçİĂçŻť’æŐëä¡£çŤĺè£Źäÿłçśżåřšäij ŽåĞžéŤŹïijŽ

```
>>> a = A()
>>> a.spam()
A.spam
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "<stdin>", line 4, in spam
AttributeError: 'super' object has no attribute 'spam'
>>>
```

ä¡ĘæŸŕïijŇåęĆæđIJä¡ăä¡£çŤĺåd’ŽçżğæL’£çŽĎèŕİçIJŃçIJŃäijŽåŔŚçŢ§äżĂäźĹïijŽ

```
>>> class B:
...     def spam(self):
...         print('B.spam')
...
>>> class C(A,B):
...     pass
...
>>> c = C()
>>> c.spam()
A.spam
B.spam
>>>
```

ä¡ăåŔŕäžěçIJŃåĹřåIJÍçśżAäÿ■ä¡£çŤÍ `super().spam()` åőđéŹĚäÿŁèřČçŤÍçŽĎæŸŕèů§çśżAæŕńæŮăåĚşçşżçŽĎçśżBäÿ■çŽĎ `spam()` æŰźæşŢãĂĆ è£ŹäÿłçŤĺçśżcçŽĎMROåĹŮèąĺåřšåŔŕäžěåőŇåĚÍèğčéĞŁæÿĚæěŽäžĘïijŽ

```
>>> C.__mro__
(<class '__main__.C'>, <class '__main__.A'>, <class '__main__.B'>,
<class 'object'>)
>>>
```

åIJÍåőŽäźL’æůůåĚěçśżçŽĎæŮűåĂŹè£Źæăůä¡£çŤÍ `super()` æŸŕå¿ĹæŹőéĄ■çŽĎãĂĆåŔŕäžěåŔĆèĂČ8.13åŠŇ8.18åřŔèŁĆãĂĆ

çĎűèĂŇïijŇçŤšäžŐ `super()` åŔŕèČ¡äijŽěŕČçŤĺäÿ■æŸŕä¡ăæČşèęĄçŽĎæŰźæşŢïijŇä¡ăåžŤèŕěéĄţå¿łäÿ

éęŰåĚĹïijŇçąőä£ÌåIJÍçżğæL'£ä¡Şçşżäÿ■æL'ĂæIJL'çŻÿåŘŇåŘ■å■ŮçŽĎæŰźæşŢæŃěæIJL'åŔŕåĚijåőźçŽĎåŔ
è£Źæăůå Ŕŕäžěçąőä£Ì `super()` èřČçŤÍäÿĂäÿłéİđçŻt'æŐěçĹűçśżæŰźæşŢæŮűäÿ■äijŽåĞžéŤŹãĂĆ
åĚűæňąïijŇæIJĂåě¡çąőä£ÌæIJĂéąűåśÇçŽĎçśżæŔŘä¿ŻäžĘè£ŹäÿłæŰźæşŢçŽĎåőđçŐřïijŇè£Źæăůç ŽĎèŕÍåIJÍ

åIJÍPythonçď¿åŇžäÿ■åŕźäžŐ `super()` çŽĎä¡£çŤÍæIJL'æŮűåĂŹäijŽåijŢæÌěäÿĂäžŻäžL'èőőãĂĆ
år¡çőąåęĆæ■d'ïijŇåęĆæđIJäÿĂåĹĞéąžåĹl'çŽĎèŕÌïijŇä¡ăåžŤèŕěåIJÍä¡ăæIJĂæŰřäžčçăĄäÿ■ä¡£çŤÍåőČãĂĆ
Raymond Hettingeräÿžæ■d'åĘŹäžĘäÿĂçŕĞéİđåÿÿåě¡çŽĎæŰĞçńă âĂIJPythonâĂŹs super() Considered Super!âĂİ ïijŇ éĂŽè£Ğåd'ğéĞŘçŽĎä¿Ńå■ŘåŘŚæĹŚäžňèğčéĞŁäžĘäÿžäżĂäźĹ `super()` æŸŕæđĄåě¡çŽĎãĂĆ

8.8 å■Řçśżäÿ■æL'ľåśŢproperty

éŮőécŸ

åIJÍå■Řçśżäÿ■ïijŇä¡ăæČşèęĄæL'ľåśŢåőŽäźL'åIJÍçĹűçśżäÿ■çŽĎpropertyçŽĎåŁ§èČ¡ãĂĆ

èğčåĘşæŰźæąĹ

èĂČèŹŚåęĆäÿŃçŽĎäžčçăĄïijŇåőČåőŽäźL'äžĘäÿĂäÿłpropertyïijŽ

```
class Person:
    def __init__(self, name):
        self.name = name

    # Getter function
    @property
    def name(self):
        return self._name

    # Setter function
    @name.setter
    def name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._name = value

    # Deleter function
    @name.deleter
    def name(self):
        raise AttributeError("Can't delete attribute")
```

äÿŃéÌćæŸŕäÿĂäÿłçď žä¿Ńçśżïij ŇåőČçżğæL'£èĞłPersonåźűæL'ľåśŢäžĘ `name` åśđæĂğçŽĎåŁ§èČ¡ïijŽ

```
class SubPerson(Person):
    @property
    def name(self):
        print('Getting name')
```

```
        return super().name

    @name.setter
    def name(self, value):
        print('Setting name to', value)
        super(SubPerson, SubPerson).name.__set__(self, value)

    @name.deleter
    def name(self):
        print('Deleting name')
        super(SubPerson, SubPerson).name.__delete__(self)
```

æŐěäÿŃæİěä¡£çŤĺè£ŹäÿłæŰřçśżïijŽ

```
>>> s = SubPerson('Guido')
Setting name to Guido
>>> s.name
Getting name
'Guido'
>>> s.name = 'Larry'
Setting name to Larry
>>> s.name = 42
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 16, in name
        raise TypeError('Expected a string')
TypeError: Expected a string
>>>
```

åęĆædIJä¡ăäżĚäżĚåŘłæČşæL'l'åśŢpropertyçŽĎæ§ŘäÿĂäÿłæŰźæşŢïijŇéĆčäźĹåŕŕäżěåČŔäÿŃéİćè£Źæ

```
class SubPerson(Person):
    @Person.name.getter
    def name(self):
        print('Getting name')
        return super().name
```

æĹŰèĂĚïijŇä¡ăåŘłæČşä£őæŤźsetteræŰźæşŢïijŇåřśè£ŹäźĹåĘŹïijŽ

```
class SubPerson(Person):
    @Person.name.setter
    def name(self, value):
        print('Setting name to', value)
        super(SubPerson, SubPerson).name.__set__(self, value)
```

èőÍèőž

åIJÍå■Řçśżäÿ■æL'l'åśŢäÿĂäÿłpropertyåŘŕèČ¡äijŽåijŢèţůå¿Ĺåď'Žäÿ■æŸŞår§èğL'çŽĎéŮőécŸïijŇ åŻăäÿžäÿĂäÿłpropertyåĚűåőđæŸŕ `getter`ãĂĄ`setter` åŠŇ

deleter æŰźæşŢçŽĎéŻęåŘĹïijŇèĂŇäÿ■æŸŕå■ŢäÿłæŰźæşŢãĂĆ
åŻăæ■ď’ïijŇä¡Ęä¡ăæĽľ’åśŢäÿĂäÿłpropertyçŽĎæŮűåĂŹïijŇä¡ăéIJĂèęĄåĚĹçąőåőŽä¡ăæŸŕåŘęèęĄéĞ■æŰřåĉ

åIJÍçňňäÿĂäÿłä¿Ńå■Řäÿ■ïijŇæĽ’ĂæIJL’çŽĎpropertyæŰźæşŢéČ¡èćńéĞ■æŰřåőŽäźĽ’ãĂĆ
åIJÍæŕŘäÿĂäÿłæŰźæşŢäÿ■ïijŇä¡£çŢĺäžĘ `super()` æİèèřČçŢĺçĹűçśżçŽĎåőđçŐřãĂĆ
åIJÍ `setter` åĞ¡æŢřäÿ■ä¡£çŢĺ `super(SubPerson, SubPerson).name.__set__(self, value)` çŽĎèŕ■åŘěæŸŕæšąæIJL’éŤŹçŽĎãĂĆ
äÿžäžĘåğŤæĽ’ŸçżŹäźŃåĽ’■åőŽäźĽçŽĎsetteræŰźæşŢïijŇéIJĂèęĄåřĘæŐğåĹűæİČäijäéĂŠçżŹäźŃåĽ’■åőŽäź`__set__()` æŰźæşŢãĂĆ äÿ■è£ĞïijŇèŐůåŘŰè£ŹäÿłæŰźæşŢçŽĎåŤřäÿĂéĂŤå¿ĎæŸŕä¡£çŢĺçśżåŘŸéĞŘèĂè£Źäź§æŸŕäÿžäžĂäźĹæĹŚäžňèęĄä¡£çŢĺ `super(SubPerson, SubPerson)` çŽĎåŐ§åŻããĂĆ

åęĆæđIJä¡ăåŘłæČşéĞ■åőŽäźĽ’åĚűäÿ■äÿĂäÿłæŰźæşŢïijŇéĆčåŘłä¡£çŢĺ @property æIJňèžńæŸŕäÿ■åď§çŽĎãĂĆærŤåęĆïijŇäÿŃéİćçŽĎäžčçăĄåřśæŮăæşŢåůěä¡IJïijŽ

```
class SubPerson(Person):
    @property  # Doesn't work
    def name(self):
        print('Getting name')
        return super().name
```

åęĆæđIJä¡ăèŕŢçİÄè£ŘèąŇäijŽåŘŚçŐřsetteråĞ¡æŢřæŢt’äÿłæűĹåď śäžĘïijŽ

```
>>> s = SubPerson('Guido')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 5, in __init__
        self.name = name
AttributeError: can't set attribute
>>>
```

ä¡ăåžŤèŕěåČŘäźŃåĽ’■èŕt’è£ĞçŽĎéĆčæăůä£őæŤźäžčçăĄïijŽ

```
class SubPerson(Person):
    @Person.getter
    def name(self):
        print('Getting name')
        return super().name
```

è£ŹäźĹåĘŹåŘŐïijŇpropertyäźŃåĽ’■åůšçżŘåőŽäźĽ’è£ĞçŽĎæŰźæşŢäijŽèćńåď■åĹűè£ĞæİěïijŇèĂŇget

```
>>> s = SubPerson('Guido')
>>> s.name
Getting name
'Guido'
>>> s.name = 'Larry'
>>> s.name
Getting name
'Larry'
>>> s.name = 42
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
```

```
    File "example.py", line 16, in name
        raise TypeError('Expected a string')
TypeError: Expected a string
>>>
```

åIJÍè£ŹäýłçĽźåĹńçŽĎèğčåĘşæŰźæąĹäÿ■ïijŇæĹŚäżňæšąåŁđæşŢä¡£çŤÍæŻť'åŁäéĂŽçŤÍçŽĎæŰźåijŔåČ
`Person` çśżåŘ■ãĂĆ åęĆæđIJä¡ääÿ■ç§ěéĄŞåĹřåžŢæŸŕåŞłäýłå§žçśżåőŽäźĽ'äžĘpropertyïijŇ
éĆčä¡ăåŔłèČ¡éĂŽè£ĞéĞ■æŰřåőŽäźĽ'æĽ'ĂæIJL'propertyåźűä¡£çŤÍ `super()`
æÌěåřĘæŐğåĹűæÌČäijäéĂŠçżŹåL'■éÍćçŽĎåőđçŐřãĂĆ

åĂijçŽĎæşÍæĎŔçŽĎæŸŕäýŁéÍćæijŤçd'žçŽĎçňňäÿĂçğ■æŁĂæIJŕè£ŸåŔŕäžěèćńçŤÍæÌěæL'l'åśŢäÿĂäýła

```
# A descriptor
class String:
    def __init__(self, name):
        self.name = name

    def __get__(self, instance, cls):
        if instance is None:
            return self
        return instance.__dict__[self.name]

    def __set__(self, instance, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        instance.__dict__[self.name] = value

# A class with a descriptor
class Person:
    name = String('name')

    def __init__(self, name):
        self.name = name

# Extending a descriptor with a property
class SubPerson(Person):
    @property
    def name(self):
        print('Getting name')
        return super().name

    @name.setter
    def name(self, value):
        print('Setting name to', value)
        super(SubPerson, SubPerson).name.__set__(self, value)

    @name.deleter
    def name(self):
        print('Deleting name')
        super(SubPerson, SubPerson).name.__delete__(self)
```

æIJÄåŘŐåĂijçŽĎæşÍæĎŘçŽĎæŸŕïijŇèŕžåĹřè£ŹéĞŇæŮűïijŇä¡ăåžŤèŕěäijŽåŔŞçŐřå■Řçśżå
setter åŠŇ deleter æŰźæşŢåĚűåőđæŸŕå¿ĹçőÄå■ŢçŽĎãĂĆ
è£ŹéĞŇæijŤçd'žçŽĎèğčåĘşæŰźæąĹåŘŇæăúéĂĆçŤĺïijŇä¡ĘæŸŕåIJÍ PythonçŽĎissueéąţéÍć
æŁěåŚŁçŽĎäÿĂäÿłbugïijŇæĹŰèőÿäijŽä¡£å¿ŮåřĘæÌěçŽĎPythonçL'ĹæIJňäÿ■åĞžçŐřäÿĂäÿłæŻt'åŁăçőÄæt'

8.9 åĹŻåžžæŰřçŽĎçśżæĹŰåőđä¿ŃåśđæĂğ

éŮőécŸ

ä¡ăæČşåĹŻåžžäÿĂäÿłæŰřçŽĎæŇěæIJL'äÿĂäžŻécÍåd'ŰåŁ§èČ¡çŽĎåőđä¿ŃåśđæĂğçśżådŃïijŇæŕŤåęĆç

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăæČşåĹŻåžžäÿĂäÿłåĚĺæŰřçŽĎåőđä¿ŃåśđæĂğïijŇåŔŕäžěéĂŽè£ĞäÿĂäÿłæŔŔè£řåŹĺçśżçŽĎ

```
# Descriptor attribute for an integer type-checked attribute
class Integer:
    def __init__(self, name):
        self.name = name

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return instance.__dict__[self.name]

    def __set__(self, instance, value):
        if not isinstance(value, int):
            raise TypeError('Expected an int')
        instance.__dict__[self.name] = value

    def __delete__(self, instance):
        del instance.__dict__[self.name]
```

äÿĂäÿłæŔŔè£řåŹĺářśæŸŕäÿĂäÿłåőđçŐřäžĘäÿL'äÿłæăÿå£ČçŽĎåśđæĂğèő£éŮőæŞ■ä¡IJ(get,
set, delete)çŽĎçśżïijŇ åĹĘåĹńäÿž __get__() ãĂĄ__set__()
åŠŇ __delete__() è£ŹäÿL'äÿłçL'źæőŁçŽĎæŰźæşŢãĂĆ
è£ŹäžŻæŰźæşŢæŐěåŔŮäÿĂäÿłåőđä¿Ňä¡IJäÿžè¿ŞåĚëïijŇäźŃåŘŐçŻÿåžŤçŽĎæŞ■ä¡IJåőđä¿ŃåžŢåśĆçŽĎå■

äÿžäžĘä¡£çŤĺäÿĂäÿłæŔŔè£řåŹĺïijŇéIJĂářĘè£ŹäÿłæŔŔè£řåŹĺçŽĎåőđä¿Ňä¡IJäÿžçśżåśđæĂğæŤ¿åĹřäÿĂ

```
class Point:
    x = Integer('x')
    y = Integer('y')

    def __init__(self, x, y):
        self.x = x
        self.y = y
```

å¡Şä¡ãè£ŹæăůåĄŽåŘŐïijŇæĽĂæIJL’åŕźæŔŔè£řåŹĺåśđæĂğ(æŕŤåęĆxæĹŰy)çŽĎèő£éŮőäijŽèćń
__get__() ãĂĄ__set__() åŠŇ __delete__() æŰźæşŢæ■ŢèŐůåĽřãĂĆä¿ŃåęĆïijŽ

```
>>> p = Point(2, 3)
>>> p.x # Calls Point.x.__get__(p,Point)
2
>>> p.y = 5 # Calls Point.y.__set__(p, 5)
>>> p.x = 2.3 # Calls Point.x.__set__(p, 2.3)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "descrip.py", line 12, in __set__
        raise TypeError('Expected an int')
TypeError: Expected an int
>>>
```

ä¡IJäÿžè¿ŞåĚëïijŇæŔŔè£řåŹĺçŽĎæŕŔäÿĂäÿłæŰźæşŢäijŽæŐěåŔŮäÿĂäÿłæŞ■ä¡IJåőđä¿ŃãĂĆ
äÿžäžĘåőđçŐřèŕůæśĆæŞ■ä¡IJïijŇäijŽçŻÿåžŤçŽĎæŞ■ä¡IJåőđä¿ŃåžŢåśĆçŽĎå■ŮåĚÿ(__dict__åśđæĂğ)ãĂĆ
æŔŔè£řåŹĺçŽĎ `self.name` åśđæĂğå■ŸåĆĺäžĘåIJĺåőđä¿Ńå■ŮåĚÿäÿ■èćńåőđéŹĚä¡£çŤĺåĽřçŽĎkeyãĂĆ

èőĺèőž

æŔŔè£řåŹĺåŔŕåőđçŐřåď’ğéČĺåĹĘPythonçśžçĽźæĂğäÿ■çŽĎåžŢåśĆé■ŤæşŢïijŇ
åŇĚæŃň `@classmethod` ãĂĄ`@staticmethod` ãĂĄ`@property` ïijŇçŤŽèĞşæŸŕ
`__slots__` çĽźæĂğãĂĆ

éĂŽè£ĞåőŽäźĽäÿĂäÿłæŔŔè£řåŹĺïijŇä¡ăåŔŕäžěåIJĺåžŢåśĆæ■ŢèŐůæäÿå£ČçŽĎåőđä¿ŃæŞ■ä¡IJ(get, set, delete)ïijŇåźűäÿŤåŔŕåőŇåĚíèĞłåőŽäźĽåőČäžňçŽĎèąŇäÿžãĂĆ
è£ŹæŸŕäÿĂäÿłåijžåď’ğçŽĎåůěåĚůïijŇæIJL’äžĘåőČä¡ăåŔŕäžěåőđçŐřå¿Ĺåď’ŽéńŸçžğåŁ§èČ¡ïijŇåźűäÿŤåőČä

æŔŔè£řåŹĺçŽĎäÿĂäÿłæŕŤè¿ČåŽřæČŚçŽĎåIJřæŰźæŸŕåőČåŔłèČ¡åIJĺçśžçžğåĹńèćńåőŽäźĽïijŇèĂŇäÿ

```
# Does NOT work
class Point:
    def __init__(self, x, y):
        self.x = Integer('x') # No! Must be a class variable
        self.y = Integer('y')
        self.x = x
        self.y = y
```

åŘŇæŮűïijŇ__get__() æŰźæşŢåőđçŐřèţůæİěæŕŤçIJŃäÿŁåŐżèęĄåď’■æİĆå¿Ůåď’ŽïijŽ

```
# Descriptor attribute for an integer type-checked attribute
class Integer:

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return instance.__dict__[self.name]
```

`__get__()` çIJŃäÿŁåŐżæIJL'çĆźåď■æİĆçŽĎåŐ§åŻää¡ŠçżŞäžŐåőđä¿ŃåŔŸéĞŔåŠŇçśżåŔŸéĞŔçŽĎ åęĆæđIJäÿĂäÿłæŔŘè£řåŹÍèćńå¡ŞåĄŽäÿĂäÿłçśżåŔŸéĞŘæİèèő£éŮőïijŇéĆčäźĹ `instance` åŘĆæŢřèćńèő¿ç¡őæĹŘ `None` ãĂĆ è£Źçğ■æČĚåĘţäÿŃïijŇæăĞåĞĘåĄŽæşŢářśæŸŕçőĂå■ŢçŽĎè£ŤåŻđè£Źä

```
>>> p = Point(2,3)
>>> p.x # Calls Point.x.__get__(p, Point)
2
>>> Point.x # Calls Point.x.__get__(None, Point)
<__main__.Integer object at 0x100671890>
>>>
```

æŔŘè£řåŹÍéĂŽåÿÿæŸŕéĆčäžŻä¡£çŤÍåĹřečĚéěřåŹÍæĹŰåĚČçśżçŽĎåď ğåđŃæąĘæđűäÿ■çŽĎäÿĂäÿłçżĎ äÿ¿äÿłä¿Ńå■ŘïijŇäÿŃéÍćæŸŕäÿĂäžŻæŻť'énŸçžğçŽĎå§žäžŐæŔŘè£řåŹÍçŽĎäžčçăĄïijŇåźűæűĽ'åŔŁåĹřäÿĂ

```
# Descriptor for a type-checked attribute
class Typed:
    def __init__(self, name, expected_type):
        self.name = name
        self.expected_type = expected_type
    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return instance.__dict__[self.name]

    def __set__(self, instance, value):
        if not isinstance(value, self.expected_type):
            raise TypeError('Expected ' + str(self.expected_type))
        instance.__dict__[self.name] = value
    def __delete__(self, instance):
        del instance.__dict__[self.name]

# Class decorator that applies it to selected attributes
def typeassert(**kwargs):
    def decorate(cls):
        for name, expected_type in kwargs.items():
            # Attach a Typed descriptor to the class
            setattr(cls, name, Typed(name, expected_type))
        return cls
    return decorate

# Example use
@typeassert(name=str, shares=int, price=float)
class Stock:
    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

æIJĂåŘŐèęĄæŇĞåĞžçŽĎäÿĂçĆźæŸŕïijŇåęĆæđIJä¡ääŔłæŸŕæČşçőĂå■ŢçŽĎèĞłåőŽäźĽ'æ§Řäÿłçśżçž è£Źçğ■æČĚåĘţäÿŃä¡£çŤÍ8.6åřŘèŁĆäžŃçż■çŽĎpropertyæŁĂæIJŕäijŽæŻť'åŁääőźæŸŞãĂĆ

å¡ŞçĺŃåžŔäÿ■æIJĽ’å¿Ĺåď’ŽéĞ■åď’■äżčçăĄçŽĎæŮűåĂŹæŔŔè£řåŹĺåřśå¿ĹæIJĽ’çŤĺäžĘ
(æŕŤåęĆä¡ăæČşåIJĺä¡ăäżčçăĄçŽĎå¿Ĺåď’ŽåIJřæŰźä¡£çŤĺæŔŔè£řåŹĺæŔŘä¿ŻçŽĎåŁ§èČ¡æĹŰèĂĚåřĘåőČä¡

8.10 ä¡£çŤĺå żűè£§èőąçőŮåśđæĂğ

éŮőéćŸ

ä¡ăæČşåřĘäÿĂäÿłåŔłèŕżåśđæĂğåőŽäźĽ’æĹŘäÿĂäÿłpropertyïijŇåźűäÿŤåŔłåIJÍèő£éŮőçŽĎæŮűåĂŹæĽ
ä¡ĘæŸŕäÿĂæŮęèćńèő£éŮőåŘŐïijŇä¡ăåÿŇæIJŻçżŞæđIJåĂijèćńçijŞå■ŸèţůæİěïijŇäÿ■çŤĺæŕŔæňąéČ¡åŐżèőą

èğčåĘşæŰźæąĹ

åőŽäźĽ’äÿĂäÿłåżűè£§åśđæĂğçŽĎäÿĂçğ■éńŸæŢĹæŰźæşŢæŸŕéĂŽè£Ğä¡£çŤĺäÿĂäÿłæŔŔè£řåŹĺçśżïijŇ

```python
class lazyproperty:
    def __init__(self, func):
        self.func = func

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            value = self.func(instance)
            setattr(instance, self.func.__name__, value)
            return value
```

ä¡ăéIJĂèęĄåČŔäÿŃéİćè£ŹæăůåİláäÿĂäÿłçśżäÿ■ä¡£çŤĺåőČïijŽ

```python
import math

class Circle:
    def __init__(self, radius):
        self.radius = radius

    @lazyproperty
    def area(self):
        print('Computing area')
        return math.pi * self.radius ** 2

    @lazyproperty
    def perimeter(self):
        print('Computing perimeter')
        return 2 * math.pi * self.radius
```

äÿŃéİćåIJĺäÿĂäÿłäžď’äžŠçŐŕåćČäÿ■æijŤçď’žåőČçŽĎä¡£çŤĺïijŽ

```
>>> c = Circle(4.0)
>>> c.radius
```

```
4.0
>>> c.area
Computing area
50.26548245743669
>>> c.area
50.26548245743669
>>> c.perimeter
Computing perimeter
25.132741228718345
>>> c.perimeter
25.132741228718345
>>>
```

äžŤçżĘèğĆår§ä¡ăäijŽåŔŜçŐřæűĹæĄŕ `Computing area` åŠŇ `Computing perimeter` äžĚäžĚåĞžçŐřäÿĂæňąãĂĆ

èőÍèőž

å¿Ĺåď´ŽæŮűåĂŹïijŇæđĎéĂăäÿĂäÿłåžűè£§èőąçőŮåśđæĂğçŽĎäÿżèęĄçŻőçŽĎæŸŕäÿžäžĘæŔŘå■ĞæĂ
ä¿ŇåęĆïijŇä¡ăåŔŕäżěéĄ£åĚ■èőąçőŮè£ŹäžŻåśđæĂğåĂijïijŇéŹď´éİđä¡ăçIJ§çŽĎéIJĂèęĄåőČäžñãĂĆ
è£ŹéĞŇæijŤçď´žçŽĎæŰźæąĹåřśæŸŕçŤĺæİěåőđçŐřè£ŹæăůçŽĎæŢĹæđIJçŽĎïijŇ
åŘłäÿ■è£ĞåőČæŸŕéĂŽè£ĞäžěéİđåÿÿéńŸæŢĹçŽĎæŰźåijŔä¡£çŤĺæŔŔè£řåŹĺçŽĎäÿĂäÿłçš¿åęŹçL´źæĂğæİě

æ■čåęĆåIJĺåĚűäžŮåřŔèŁĆ(åęĆ8.9åřŔèŁĆ)æL´Ăèőšç ŽĎéĆčæăůïijŇå¡ŞäÿĂäÿłæŔŔè£řåŹĺèćńæŤ¿åĚěäÿ
æŕŘæňąèő£éŮőåśđæĂğæŮűåőČçŽĎ `__get__()` ãĂĄ `__set__()` åŠŇ `__delete__()`
æŰźæşŢåřśäijŽèćńèğęåŔŜãĂĆ äÿ■è£ĞïijŇåęĆæđIJäÿĂäÿłæŔŔè£řåŹĺäžĚäžĚåŘłåőŽäźL´äžĘäÿĂäÿł
`__get__()` æŰźæşŢçŽĎèŕİïijŇåőČæŕŤéĂŽåÿÿçŽĎåĚůæIJL´æŻť´åijśçŽĎçżŚåőŽãĂĆ
çL´źåĹńåIJřïijŇåŘłæIJL´å¡Şèćńèő£éŮőåśđæĂğäÿ■åIJĺåőđä¿ŇåžŢåśĆçŽĎå■ŮåĚÿäÿ■æŮű
`__get__()` æŰźæşŢæL´■äijŽèćńèğęåŔŜãĂĆ

`lazyproperty` çśżåĹl´çŤĺè£ŹäÿĂçĆźïijŇä¡£çŤĺ `__get__()`
æŰźæşŢåIJĺåőđä¿Ńäÿ■å■ŸåĆĺèőąçőŮåĞžæİěçŽĎåĂijïijŇ è£Źäÿłåőđä¿Ńä¡£çŤĺçŻÿåŘŇçŽĎåŘ■å■Ůä¡IJäÿž
è£ŹæăůäÿĂæİěïijŇçżŞæđIJåĂijècńå■ŸåĆĺåIJĺåőđä¿Ńå■ŮåĚÿäÿ■åźűäÿŤäžěåŘŐårśäÿ■éIJĂèęĄåĘ■åŐżèőąçő
ä¡ăåŔŕäžěåřİèŕŢæŻť´æůśåĚěçŽĎä¿Ńå■ŘæİèèğĆår§çżŞæđIJïijŽ

```
>>> c = Circle(4.0)
>>> # Get instance variables
>>> vars(c)
{'radius': 4.0}

>>> # Compute area and observe variables afterward
>>> c.area
Computing area
50.26548245743669
>>> vars(c)
{'area': 50.26548245743669, 'radius': 4.0}

>>> # Notice access doesn't invoke property anymore
>>> c.area
```

```
50.26548245743669

>>> # Delete the variable and see property trigger again
>>> del c.area
>>> vars(c)
{'radius': 4.0}
>>> c.area
Computing area
50.26548245743669
>>>
```

è£Źçğ■æŰźæąŁæIJL'äÿĂäÿłářŔçijžéŹůářśæŸŕèőąçőŮåĞžçŽĎåĂijèćńåŁŻåżžåŘŐæŸŕåŔŕäžěèćńä£őæŤ

```
>>> c.area
Computing area
50.26548245743669
>>> c.area = 25
>>> c.area
25
>>>
```

åęĆædIJä¡ăæŃĚå£Čè£ŹäÿłéŮőécŸïijŇéĆčäźŁåŔŕäžěä¡£çŤĺäÿĂçğ■çĺ■å¿őæšąéĆčäźŁéńŸæŢĹçŽĎåőđç

```
def lazyproperty(func):
    name = '_lazy_' + func.__name__
    @property
    def lazy(self):
        if hasattr(self, name):
            return getattr(self, name)
        else:
            value = func(self)
            setattr(self, name, value)
            return value
    return lazy
```

åęĆædIJä¡ăä¡£çŤĺè£ŹäÿłçL'ŁæIJñïijŇåřśäijŽåŔŚçŐřçŐřåIJĺä£őæŤźæŞ■ä¡IJåůšçżŔäÿ■èćńåĚĄèőÿäžĘïij

```
>>> c = Circle(4.0)
>>> c.area
Computing area
50.26548245743669
>>> c.area
50.26548245743669
>>> c.area = 25
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>>
```

çĎűèĂŇïijŇè£Źçğ■æŰźæąŁæIJL'äÿĂäÿłçijžçĆźåřśæŸŕæL'ĂæIJL'getæŞ■ä¡IJéČ¡å£ĚéążèćńåőŽåŘŚåĹřå
getter åĞ¡æŢřäÿŁåŐżãĂĆ è£Źäÿłèů§äźŇåL'■çőĂå■ŢçŽĎåIJĺåőđä¿Ńå■ŮåĚÿäÿ■æ§ěæL'¿åĂijçŽĎæŰźæą

åęĆæđIJæČşèŐůåŔŰæŻť'åď'ŽåĚşäžŐpropertyåŠŇåŔŕçőąçŘĘåśđæĂğçŽĎä£ąæĄŕïijŇåŔŕäžěåŔĆèĂČ8.6åřŔ

8.11 çőĂåŇŰæŢřæ■őçżŞæđĎçŽĎåĹlåğŃåŇŰ

éŮőécŸ

ä¡ăåĘŹäžĘå¿Ĺåď'ŽäżĚäżĚçŤĺä¡IJæŢřæ■őçżŞæđĎçŽĎçśżïijŇäÿ■æČşåĘŹåď'łåď'ŽçČęäžžçŽĎ `__init__()` åĞ¡æŢř

èğčåĘşæŰźæąĹ

åŔŕäžěåIJĺäÿĂäÿłå§žçśżäÿ■åĘŹäÿĂäÿłåĚňçŤĺçŽĎ `__init__()` åĞ¡æŢřïijŽ

```
import math

class Structure1:
    # Class variable that specifies expected fields
    _fields = []

    def __init__(self, *args):
        if len(args) != len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))
        # Set the arguments
        for name, value in zip(self._fields, args):
            setattr(self, name, value)
```

çĎűåŘŐä¡£ä¡ăçŽĎçśżçżğæL'£èĞłè£ŹäÿłåŞžçśż:

```
# Example class definitions
class Stock(Structure1):
    _fields = ['name', 'shares', 'price']

class Point(Structure1):
    _fields = ['x', 'y']

class Circle(Structure1):
    _fields = ['radius']

    def area(self):
        return math.pi * self.radius ** 2
```

ä¡£çŤĺè£ŹäžŽçśżçŽĎçď'žä¿ŇïijŽ

```
>>> s = Stock('ACME', 50, 91.1)
>>> p = Point(2, 3)
>>> c = Circle(4.5)
>>> s2 = Stock('ACME', 50)
```

```
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "structure.py", line 6, in __init__
        raise TypeError('Expected {} arguments'.format(len(self._
↪fields)))
TypeError: Expected 3 arguments
```

åęĆædIJè£ŸæČşæŤŕæŇĄåĚşéŤőå■ŮåŔĆæŢřijŇåŔŕäžěářĘåĚşéŤőå■ŮåŔĆæŢřèő¿ç¡őäÿžåőđä¿ŃåśđæÀ

```
class Structure2:
    _fields = []

    def __init__(self, *args, **kwargs):
        if len(args) > len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))

        # Set all of the positional arguments
        for name, value in zip(self._fields, args):
            setattr(self, name, value)

        # Set the remaining keyword arguments
        for name in self._fields[len(args):]:
            setattr(self, name, kwargs.pop(name))

        # Check for any remaining unknown arguments
        if kwargs:
            raise TypeError('Invalid argument(s): {}'.format(','.
↪join(kwargs)))
# Example use
if __name__ == '__main__':
    class Stock(Structure2):
        _fields = ['name', 'shares', 'price']

    s1 = Stock('ACME', 50, 91.1)
    s2 = Stock('ACME', 50, price=91.1)
    s3 = Stock('ACME', shares=50, price=91.1)
    # s3 = Stock('ACME', shares=50, price=91.1, aa=1)
```

ä¡ăè£ŸèČ¡åřĘäÿ■åIJÍ `_fields` äÿ■çŽĎåŘ■çğřåŁăåĚěåĹřåśđæĂğäÿ■åŐżïijŽ

```
class Structure3:
    # Class variable that specifies expected fields
    _fields = []

    def __init__(self, *args, **kwargs):
        if len(args) != len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))

        # Set the arguments
```

```
        for name, value in zip(self._fields, args):
            setattr(self, name, value)

        # Set the additional arguments (if any)
        extra_args = kwargs.keys() - self._fields
        for name in extra_args:
            setattr(self, name, kwargs.pop(name))

        if kwargs:
            raise TypeError('Duplicate values for {}'.format(','.
↪join(kwargs)))

# Example use
if __name__ == '__main__':
    class Stock(Structure3):
        _fields = ['name', 'shares', 'price']

    s1 = Stock('ACME', 50, 91.1)
    s2 = Stock('ACME', 50, 91.1, date='8/2/2012')
```

èőÍèőž

å¡Şä¡ăéIJĂèęĄä¡£çŤĺåď ğéĞŕå¿ĹåřŕçŽĎæŢřæ■őçżŞæđĎçśżçŽĎæŮűåĂŹïijŇ çŻÿæŕŤæĽŃåůëäÿĂäÿłäÿłåőŽäźĽ `__init__()` æŰźæşŢèĂŇåůšïijŇä¡£çŤĺè£Źçğ■æŰźåijŕåŕŕäžëåď ğåď ğ

åIJĺäÿŁéİćçŽĎåőđçŐřäÿ■æĹŚäžňä¡£çŤĺäžĘ `setattr()` åĞ¡æŢřçśżèő¿ç¡őåśđæĂğåĂijïijŇ ä¡ăåŕŕèČ¡äÿ■æČşçŤĺè£Źçğ■æŰźåijŕïijŇèĂŇæŸŕæČşçŻť æŐëæŻť æŰřåőć

```
class Structure:
    # Class variable that specifies expected fields
    _fields= []
    def __init__(self, *args):
        if len(args) != len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))

        # Set the arguments (alternate)
        self.__dict__.update(zip(self._fields,args))
```

åř¡çőąè£Źäźś§åŕŕäžëæ■čåÿÿåůëä¡IJïijŇä¡ĘæŸŕå¡ŞåőŽäźĽå■ŘçśżçŽĎæŮűåĂŹéŮőécŸåřśæİëäžĘãĂĆ å¡ŞäÿĂäÿłå■ŘçśżåőŽäźĽäžĘ `__slots__` æĹŰèĂĚéĂŽè£Ğproperty(æĹŰæŔŕè£řåŹĺ)æİëåŇĚècĚæ§Răÿłå éĆčäźĹçŻť æŐëèő£éŮőåőđä¿Ńå■ŮåĚÿåřśäÿ■ètůä¡IJçŤĺäžĘãĂĆæĹŚäžňäÿŁéİćä¡£çŤĺ `setattr()` äijŽæŸ¿å¿ŮæŻť éĂŽçŤĺäžŻïijŇåŽääÿžåőČäź§éĂĆçŤĺäžŐå■ŘçśżæČĚåĘţãĂĆ

è£Źçğ■æŰźæşŢåŤŕäÿĂäÿ■åě¡çŽĎåIJřæŰźåřśæŸŕåŕźæ§ŘäžZIDEèĂŇèĺĂïijŇåIJĺæŸ¿çd'žåÿőåŁľåĞ¡æŢ

```
>>> help(Stock)
Help on class Stock in module __main__:
class Stock(Structure)
...
```

```
| Methods inherited from Structure:
|
| __init__(self, *args, **kwargs)
|
...
>>>
```

åŔŕäżěåŔĆèĂČ9.16åřŔèŁĆæİěåijžåĹűåIJĺ `__init__()` æŰźæşŢäÿ■æŇĞåőŽåŔĆæŢřçŽĎçśżåđŃç■¿åŘ■ãĂĆ

8.12 åőŽäźĽ'æŐěåŔčæĹŰèĂĚæŁ¡èśąå§žçśż

éŮőécŸ

ä¡ăæČşåőŽäźĽäÿĂäÿłæŐěåŔčæĹŰæŁ¡èśąçśżïijŇåźűäÿŤéĂŽè£ĞæĽ'ğèąŇçśżåđŃæčĂæ§ěæİěçąőä£İå■Ř

èğčåĘşæŰźæąĹ

ä¡£çŤĺ `abc` æĺąåİŮåŔŕäżěå¿Ĺè¡żæİ¿çŽĎåőŽäźĽ'æŁ¡èśąå§žçśżïijŽ

```
from abc import ABCMeta, abstractmethod

class IStream(metaclass=ABCMeta):
    @abstractmethod
    def read(self, maxbytes=-1):
        pass

    @abstractmethod
    def write(self, data):
        pass
```

æŁ¡èśąçśżçŽĎäÿĂäÿłçL'źçĆźæŸŕåőČäÿ■èČ¡çŻť'æŐěèćńåőđä¿ŃåŇŰïijŇæŕŤåęĆä¡ăæČşåČŔäÿŃéİćè£Ź

```
a = IStream() # TypeError: Can't instantiate abstract class
              # IStream with abstract methods read, write
```

æŁ¡èśąçśżçŽĎçŻőçŽĎåřśæŸŕèől'åĹńçŽĎçśżçżğæL'£åőČåźűåőđçŐřçL'źåőŽçŽĎæŁ¡èśąæŰźæşŢïijŽ

```
class SocketStream(IStream):
    def read(self, maxbytes=-1):
        pass

    def write(self, data):
        pass
```

æŁ¡èśąå§žçśżçŽĎäÿĂäÿłäÿżèęĄçŤĺéĂŤæŸŕåIJĺäżčçăĄäÿ■æčĂæ§ěæ§ŘäžŽçśżæŸŕåŘęäÿžçL'źåőŽçśżåđ

```
def serialize(obj, stream):
    if not isinstance(stream, IStream):
        raise TypeError('Expected an IStream')
    pass
```

éŹď'äžĘçżğæL'£è£Źçğ■æŰźåijŔåď'ŰïijŇè£ŸåŔŕäżěéĂŽè£ĞæşĺåĘŇæŰźåijŔæİěëőĺ'æ§ŘäÿłçśžåőđçŐřæ

```
import io

# Register the built-in I/O classes as supporting our interface
IStream.register(io.IOBase)

# Open a normal file and type check
f = open('foo.txt')
isinstance(f, IStream) # Returns True
```

`@abstractmethod` è£ŸèČ¡æşĺèğčéİŹæĂĄæŰźæşŢãĂĄçżæŰźæşŢåŠŇ `properties` ãĂĆ ä¡ăåŔłéIJĂä£İèŕĄè£Źäÿłæşĺèğčçť'ğéİăåIJĺåĞ¡æŢřåőŽäźL'ål'■å■şåŔŕïijŽ

```
class A(metaclass=ABCMeta):
    @property
    @abstractmethod
    def name(self):
        pass

    @name.setter
    @abstractmethod
    def name(self, value):
        pass

    @classmethod
    @abstractmethod
    def method1(cls):
        pass

    @staticmethod
    @abstractmethod
    def method2():
        pass
```

èőĺèőž

æăĞåĞĘåžŞäÿ■æIJL'å¿Ĺåď'ŽçŤĺåĹřæŁ¡èśąå§žçśżçŽĎåIJřæŰźãĂĆ`collections` æĺąåİŮåőŽäźL'äžĘå¿Ĺåď'Žèů§åőźåŹĺåŠŇè£■äżčåŹĺ(åžŘåĹŮãĂĄæŸäářĎãĂĄéŽĘåŘĹç■L')æIJL'åĚşçŽĎæŁ `numbers` åžŞåőŽäźL'äžĘèů§æŢřå■Ůåŕźèśą(æŢť'æŢřãĂĄæţőçĆźæŢřãĂĄæIJL'çŘĘæŢřç■L')æIJL'åĚşçŽĎå§ åžŞåőŽäźL'äžĘå¿Ĺåď'Žèů§I/OæŞ■ä¡IJçŽÿåĚşçŽĎå§žçśžãĂĆ

ä¡ăåŔŕäżěä¡£çŤĺécĎåőŽäźL'çŽĎæŁ¡èśąçśżæİěæL'ğèąŇæŽť'éĂŽçŤĺçŽĎçśżåđŃæčĂæ§ěïijŇä¿ŃåęĆïijŽ

```
import collections

# Check if x is a sequence
if isinstance(x, collections.Sequence):
...

# Check if x is iterable
if isinstance(x, collections.Iterable):
...

# Check if x has a size
if isinstance(x, collections.Sized):
...

# Check if x is a mapping
if isinstance(x, collections.Mapping):
```

åř¡çőąABCsåŔŕäžěèőľæĹŚäžňå¿ĹæŰźä¿£çŽĎåĄŽçśžåđŃæčĂæ§ěïijŇä¡ĘæŸŕæĹŚäžňåIJĺäžčçăĄäÿ■æĹ
åŽääÿžPythonçŽĎæIJňèťĺæŸŕäÿĂéŮĺåŁĺæĂĄçijŰçĺŃèí■èĺĂïijŇåĚűçŽőçŽĎåřśæŸŕçżŹä¡äæŽťåď˙ŽçĄţæťżä
åijžåĹűçśžåđŃæčĂæ§ěæĹŰèőľä¡ääžčçăĄåŔŸå¿ŮæŽťåď˙■æİĆïijŇè£ŹæăůåĄŽæŮăåijĆäžŐèĹ■æIJňæśĆæIJ

8.13 åőđçŐřæŢřæ■őæĺąåđŃçŽĎçśžåđŃçžęæİ§

éŮőécŸ

ä¡äæČşåőŽäźL’æ§ŘäžŽåIJĺåśđæĂğètŃåĂijäÿŁéİćæIJL’éŹŘåĹűçŽĎæŢřæ■őçżŞæđĎãĂĆ

èğčåĘşæŰźæąĹ

åIJĺè£ŹäÿłéŮőécŸäÿ■ïijŇä¡äéIJĂèęĄåIJĺářźæ§ŘäžŽåőđä¿ŃåśđæĂğètŃåĂijæŮűè£ŽèąŇæčĂæ§ěãĂĆ
æĽĂäžěä¡äèęĄèĞłåőŽäźL’åśđæĂğètŃåĂijåĞ¡æŢřïijŇè£ŹçğæČĚåĘţäÿŃæIJĂåě¡ä¡£çŤĺæŔŔè£řåŹĺãĂĆ

äÿŃéİćçŽĎäžčçăĄä¡£çŤĺæŔŔè£řåŹĺåőđçŐřäžĘäÿĂäÿłçşżçż§çśžåđŃåŠŇètŃåĂijéłŇèŕĄæąĘæđűïijŽ

```
# Base class. Uses a descriptor to set a value
class Descriptor:
    def __init__(self, name=None, **opts):
        self.name = name
        for key, value in opts.items():
            setattr(self, key, value)

    def __set__(self, instance, value):
        instance.__dict__[self.name] = value


# Descriptor for enforcing types
class Typed(Descriptor):
    expected_type = type(None)
```

```
    def __set__(self, instance, value):
        if not isinstance(value, self.expected_type):
            raise TypeError('expected ' + str(self.expected_type))
        super().__set__(instance, value)


# Descriptor for enforcing values
class Unsigned(Descriptor):
    def __set__(self, instance, value):
        if value < 0:
            raise ValueError('Expected >= 0')
        super().__set__(instance, value)


class MaxSized(Descriptor):
    def __init__(self, name=None, **opts):
        if 'size' not in opts:
            raise TypeError('missing size option')
        super().__init__(name, **opts)

    def __set__(self, instance, value):
        if len(value) >= self.size:
            raise ValueError('size must be < ' + str(self.size))
        super().__set__(instance, value)
```

è£ŹäžŻçśžåřśæŸŕäąäèęĄåĹŻåžžçŽĎæŢřæ■őæĺąåđŃæĹŰçśžåđŃçşżçż§çŽĎå§žçąĂæđĎåžžæĺąåİŮãĂĆäÿŃéİćåřśæŸŕæĹŚäžňåőđéŹĚåőŽäźL’çŽĎåŘĎçğ■äÿ■åŘŇçŽĎæŢřæ■őçśžåđŃïijŽ

```
class Integer(Typed):
    expected_type = int

class UnsignedInteger(Integer, Unsigned):
    pass

class Float(Typed):
    expected_type = float

class UnsignedFloat(Float, Unsigned):
    pass

class String(Typed):
    expected_type = str

class SizedString(String, MaxSized):
    pass
```

çĎűåŘŐä¡£çŤĺè£ŹäžŻèĞłåőŽäźL’æŢřæ■őçśžåđŃïijŇæĹŚäžňåőŽäźL’äÿĂäÿłçśżïijŽ

```
class Stock:
    # Specify constraints
    name = SizedString('name', size=8)
    shares = UnsignedInteger('shares')
    price = UnsignedFloat('price')

    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

çĎűåŘŐæţŃèŕŢè£Źäÿłçśżç ŽĎåśđæĂğèţŃåĂijçžęæİ§ïijŇåŔŕåŔŚçŐřåŕźæ§ŘäžŽåśđæĂğçŽĎèţŃåĂijè£İ

```
>>> s.name
'ACME'
>>> s.shares = 75
>>> s.shares = -10
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 17, in __set__
        super().__set__(instance, value)
    File "example.py", line 23, in __set__
        raise ValueError('Expected >= 0')
ValueError: Expected >= 0
>>> s.price = 'a lot'
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 16, in __set__
        raise TypeError('expected ' + str(self.expected_type))
TypeError: expected <class 'float'>
>>> s.name = 'ABRACADABRA'
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 17, in __set__
        super().__set__(instance, value)
    File "example.py", line 35, in __set__
        raise ValueError('size must be < ' + str(self.size))
ValueError: size must be < 8
>>>
```

è£ŸæIJL’äÿĂäžŻæŁĂæIJŕåŔŕäžěçőĂåŇŰäÿŁéİççŽĎäžčçăĄïijŇåĚűäÿ■äÿĂçğ■æŸŕä¡£çŤĺçśżèčĚéěřåŹĺ

```
# Class decorator to apply constraints
def check_attributes(**kwargs):
    def decorate(cls):
        for key, value in kwargs.items():
            if isinstance(value, Descriptor):
                value.name = key
                setattr(cls, key, value)
            else:
                setattr(cls, key, value(key))
```

```
        return cls

    return decorate

# Example
@check_attributes(name=SizedString(size=8),
                  shares=UnsignedInteger,
                  price=UnsignedFloat)
class Stock:
    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

åŔęåď Űäÿ Äçğ■æŰźåijŔæŸŕä¡£çŤĺåĚČçśżïijŽ

```
# A metaclass that applies checking
class checkedmeta(type):
    def __new__(cls, clsname, bases, methods):
        # Attach attribute names to the descriptors
        for key, value in methods.items():
            if isinstance(value, Descriptor):
                value.name = key
        return type.__new__(cls, clsname, bases, methods)

# Example
class Stock2(metaclass=checkedmeta):
    name = SizedString(size=8)
    shares = UnsignedInteger()
    price = UnsignedFloat()

    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

èőÍèőž

æIJňèŁĆä¡£çŤĺäžĘå¿Ĺåď ŽéńŸçžğæŁĂæIJŕïijŇåŇĚæŃňæŔŔè£řåŹĺãĂĄæůůåĚěçśżãĂĄsuper() çŽĎä¡£çŤĺãĂĄçśżèčĚéěřåŹĺåŠŇåĚČçśżãĂĆ äÿ■åŔŕèČ¡åIJĺè£ŹéĞŇäÿĂäÿĂèŕęçżĘåśŢåijĂæİěèőšïijŇä¡ĘæŸä¡ĘæŸŕïijŇæĹŚåIJĺè£ŹéĞŇè£ŸæŸŕèęĄæŔŘäÿĂäÿŃåĞăäÿłéIJĂèęĄæşĺæĎŔçŽĎçĆźãĂĆ

éęŰåĚĹïijŇåIJÍ Descriptor å§žçśżäÿ■ä¡ăäijŽçIJŃåĹřæIJL'äÿł __set__() æŰźæşŢïijŇå■ť'æšąæIJL'çŻÿåžŤçŽĎ __get__() æŰźæşŢãĂĆ åęĆæđIJäÿĂäÿłæŔŔè£řäżĚäżĚæŸŕäżŐåžŢåśĆåőđä¿Ńå■ŮåĚÿäÿ■èŐůåŔŰæ§ŘäÿłåśđæĂğåĂijçŽĎèŕÍïijŇéĆ __get__() æŰźæşŢãĂĆ

æĽĂæIJL'æŔŔè£řåŹĺçśżéČ¡æŸŕå§žäžŐæůůåĚěçśżæİěåőđçŐřçŽĎãĂĆæŕŤåęĆ Unsigned åŠŇ MaxSized èęĄèů§åĚűäżŰçżğæL'£èĞł Typed çśżæůůåĚěãĂĆ

è£ŹéĞŇåĹľçŤĺåď Žçżğæ L'£æİěåőđçŐřçŻÿåžŤçŽĎåŁ§èČ¡ãĂĆ

æůůåĚěçšżçŽĎäÿĂäÿłæŕŤè¿ČéŽ¿çŘęèğčçŽĎåIJřæŰźæŸŕïijŇèřČçŤĺ
super() åĞ¡æŢřæŮűïijŇä¡ăåźűäÿ■ç§ěéĄŞçľűçń§èęĄèřČçŤĺåŞłäÿłåĚůä¡ŞçšżãĂĆ
ä¡ăéIJĂèęĄèů§åĚűäżŰçšżçżŞåŘĹåŘŐæL'■èČ¡æ■čçąőçŽĎä¡£çŤĺïijŇäź§ářśæŸŕå£Ěéążå ŘĹä¡IJæL'■èČ¡äžğçŤ

ä¡£çŤĺçšżèčĚéěřåŹĺåŠŇåĚČçšżéĂŽåÿÿåŔŕäžěçőĂåŇŰäżčçăĄãĂĆäÿŁéİćäÿd'äÿłä¿Ńå■Řäÿ■ä¡ääijŽåŔŚ

```
# Normal
class Point:
    x = Integer('x')
    y = Integer('y')

# Metaclass
class Point(metaclass=checkedmeta):
    x = Integer()
    y = Integer()
```

æL'ĂæIJL'æŰźæşŢäÿ■ïijŇçšżèčĚéěřåŹĺæŰźæąĹåžŤèřěæŸŕæIJĂçĄţæť'żåŠŇæIJĂéńŸæŸŐçŽĎãĂĆ
éęŰåĚĹïijŇåőČåźűäÿ■ä¿İèţŰäżżä¡ŢåĚűäżŰæŰřçŽĎæŁĂæIJŕïijŇæŕŤåęĆåĚČçšżãĂĆåĚűæňąïijŇèčĚéěřåŹĺå

æIJĂåŘŐïijŇèčĚéěřåŹĺè£ŸèČ¡ä¡IJäÿžæůůåĚěçšżçŽĎæŻ£äżčæŁĂæIJræİěåőđçŐřåŘŇæăůçŽĎæŢĹæđIJ

```
# Decorator for applying type checking
def Typed(expected_type, cls=None):
    if cls is None:
        return lambda cls: Typed(expected_type, cls)
    super_set = cls.__set__

    def __set__(self, instance, value):
        if not isinstance(value, expected_type):
            raise TypeError('expected ' + str(expected_type))
        super_set(self, instance, value)

    cls.__set__ = __set__
    return cls


# Decorator for unsigned values
def Unsigned(cls):
    super_set = cls.__set__

    def __set__(self, instance, value):
        if value < 0:
            raise ValueError('Expected >= 0')
        super_set(self, instance, value)

    cls.__set__ = __set__
    return cls


# Decorator for allowing sized values
```

```
def MaxSized(cls):
    super_init = cls.__init__

    def __init__(self, name=None, **opts):
        if 'size' not in opts:
            raise TypeError('missing size option')
        super_init(self, name, **opts)

    cls.__init__ = __init__

    super_set = cls.__set__

    def __set__(self, instance, value):
        if len(value) >= self.size:
            raise ValueError('size must be < ' + str(self.size))
        super_set(self, instance, value)

    cls.__set__ = __set__
    return cls


# Specialized descriptors
@Typed(int)
class Integer(Descriptor):
    pass


@Unsigned
class UnsignedInteger(Integer):
    pass


@Typed(float)
class Float(Descriptor):
    pass


@Unsigned
class UnsignedFloat(Float):
    pass


@Typed(str)
class String(Descriptor):
    pass


@MaxSized
class SizedString(String):
    pass
```

è£Źçğ■æŰźåijŔåőŽäźĽ'çŽĎçśžèů§äźŃåĽ'■çŽĎæŢĹæđIJäÿĂæăůïijŇèĂŇäÿŤæĽ'ğèąŇéĂ§åžęäijŽæŻť'å
èő¿ç¡őäÿĂäÿłçőĂå■ŢçŽĎçśžåđŃåśđæĂğçŽĎåĂijïijŇèčĚéěřåŹĺæŰźåijŔèęĄæŕŤäźŃåĽ'■çŽĎæůůåĚěçśžçŽĎ
çŐřåIJĺä¡ăåžŤèřěåžĘåźÿèĞłåůśèŕžåőŇäžĘæIJňèŁĆåĚĺéČĺåĘĚåőźäžĘåŘğïij§^_^

8.14 åőđçŐřèĞłåőŽäźĽ'åőźåŹĺ

éŮőéćŸ

ä¡ăæČşåőđçŐřäÿĂäÿłèĞłåőŽäźĽ'çŽĎçśžæÌěæĺąæŃ§åĘĚç¡őçŽĎåőźåŹĺçśžåŁ§èČ¡ïijŇærŤåęĆåĹŮèąĺåŠ

èğčåĘşæŰźæąĹ

`collections` åőŽäźĽ'äžĘå¿Ĺåd'ŽæŁ¡èśąå§žçśżïijŇå¡Şä¡ăæČşèĞłåőŽäźĽ'åőźåŹĺçśžçŽĎæŮűåĂŹåőČ
ærŤåęĆä¡ăæČşèől'ä¡ăçŽĎçśžæŤŕæŇĄè£■äžčïijŇéĆčåřśèől'ä¡ăçŽĎçśžçžğæL'£
`collections.Iterable` å■şåŔřïijŽ

```
import collections
class A(collections.Iterable):
    pass
```

äÿ■è£Ğä¡ăéIJĂèęĄåőđçŐř `collections.Iterable` æL'ĂæIJL'çŽĎæŁ¡èśąæŰźæşŢïijŇåŘęåĹŹäijŽæŁěéŤŹ:

```
>>> a = A()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: Can't instantiate abstract class A with abstract methods␣
 ↪__iter__
>>>
```

ä¡ăåŔłèęĄåőđçŐř `__iter__()` æŰźæşŢåřśäÿ■äijŽæŁěéŤŹäžĘ(åŔĆèĂČ4.2åŠŇ4.7åřŔèŁĆ)ãĂĆ

ä¡ăåŔŕäžěåĚĹèŕŢçİĂåŐžåőđä¿ŃåŇŰäÿĂäÿłåŕźèśąïijŇåIJĺéŤŹèŕŕæŔŘçd'žäÿ■åŔŕäžěæL'¿åĹřéIJĂèęĄåő

```
>>> import collections
>>> collections.Sequence()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: Can't instantiate abstract class Sequence with abstract␣
 ↪methods \
__getitem__, __len__
>>>
```

äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎçd'žä¿ŃïijŇçžğæL'£èĞłäÿŁéİćSequenceæŁ¡èśąçśžïijŇåźűäÿŤåőđçŐřåĚČ

```
class SortedItems(collections.Sequence):
    def __init__(self, initial=None):
        self._items = sorted(initial) if initial is not None else []
```

```
    # Required sequence methods
    def __getitem__(self, index):
        return self._items[index]

    def __len__(self):
        return len(self._items)

    # Method for adding an item in the right location
    def add(self, item):
        bisect.insort(self._items, item)


items = SortedItems([5, 1, 3])
print(list(items))
print(items[0], items[-1])
items.add(2)
print(list(items))
```

åŘŕäžěçIJŃåĹřïijŇSortedItemsèů§æŹőéĂŽçŽĎåžŘåĹŮæšąäżĂäźĹäÿď æăůïijŇæŤŕæŇĄæL’ĂæIJL’åÿÿç

è£ŹéĞŇéİćä¡£çŤĺåĹřäžĘ `bisect` æĺąåİŮïijŇåőČæŸŕäÿĂäÿłåIJĺæŐŠåžŘåĹŮèąĺäÿ■æŘŠåĚěåĚČçť ăçŽ

èőĺèőž

ä¡£çŤĺ `collections` äÿ■çŽĎæŁ¡èśąå§žçśżåŘŕäžěçąőä£İä¡ăèĞłåőŽäźL’çŽĎåőźåŹĺåőđçŐřäžĘæL’ĂæL
ä¡ăçŽĎèĞłåőŽäźL’åőźåŹĺäijŽæżąèűşåď’ğéČĺåĹĘçśżåđŃæčĂæ§ěéIJĂèęĄïijŇąęĆäÿŃæL’Ăçď’žïijŽ

```
>>> items = SortedItems()
>>> import collections
>>> isinstance(items, collections.Iterable)
True
>>> isinstance(items, collections.Sequence)
True
>>> isinstance(items, collections.Container)
True
>>> isinstance(items, collections.Sized)
True
>>> isinstance(items, collections.Mapping)
False
>>>
```

`collections` äÿ■å¿Łåď’ŽæŁ¡èśąçśżäijŽäÿžäÿĂäžŻåÿÿèğĄåőźåŹĺæŞ■ä¡IJæŘŘä¿ŻéżŸèőď’çŽĎåőđçČ
è£Źæăůäÿ ĂæİěäĄăåŘłéIJĂèęĄåőđçŐřéĆčäžŻä¡ăæIJĂæĎ§åĚť’èűčçŽĎæŰźæşŢå■şåŘŕãĂĆåĄĞèő¿ä¡ăçŽĎçśż
`collections.MutableSequence` ïijŇåęĆäÿŃïijŽ

```
class Items(collections.MutableSequence):
    def __init__(self, initial=None):
        self._items = list(initial) if initial is not None else []
```

```
    # Required sequence methods
    def __getitem__(self, index):
        print('Getting:', index)
        return self._items[index]

    def __setitem__(self, index, value):
        print('Setting:', index, value)
        self._items[index] = value

    def __delitem__(self, index):
        print('Deleting:', index)
        del self._items[index]

    def insert(self, index, value):
        print('Inserting:', index, value)
        self._items.insert(index, value)

    def __len__(self):
        print('Len')
        return len(self._items)
```

åęĆædIJä¡ăåĹŻåžž `Items` çŽĎåőđä¿ŃïijÑä¡ăäijŽåŔŚçŐřåőČæŤŕæŇĄåĞäázŐæĽ'ĂæIJL'çŽĎæäÿå£ČåĺäÿŃéIćæŸŕä¡£çŤĺæijŤçd'žïijŽ

```
>>> a = Items([1, 2, 3])
>>> len(a)
Len
3
>>> a.append(4)
Len
Inserting: 3 4
>>> a.append(2)
Len
Inserting: 4 2
>>> a.count(2)
Getting: 0
Getting: 1
Getting: 2
Getting: 3
Getting: 4
Getting: 5
2
>>> a.remove(3)
Getting: 0
Getting: 1
Getting: 2
Deleting: 2
>>>
```

æIJňářŔèŁĆåŔłæŸŕářźPythonæŁ¡èšąçśżåŁ§èČ¡çŽĎæŁŻçăŰåijŢçŐĽ'ãĂĆnumbers

æÍąåİŮæŔŘä¿ŻäžĘäÿĂäÿłçśżäijijçŽĎèů§æŢťæŢřçśżåđŃçŻÿåĚşçŽĎæŁ¡èśąçśżåđŃéŻĘåŘĹãĂĆ åŘŕäżěåŘĆèĂČ8.12åřŔèŁĆæİěæđĎéĂăæŻť åď ŽèĞłåőŽäźĽ'æŁ¡èśąå§žçśżãĂĆ

8.15 åśđæĂğçŽĎäżčçŘĘèő£éŮő

éŮőécŸ

ä¡ăæČşåřĘæ§Řäÿłåőđä¿ŃçŽĎåśđæĂğèő£éŮőäżčçŘĘåĹřåĘĚéČĺåŔęäÿĂäÿłåőđä¿Ńäÿ■åŐżïijŇçŻőçŽĎå

èğčåĘşæŰźæąĹ

çőĂå■Ţæİěèŕť ïijŇäżčçŘĘæŸŕäÿĂçğ■çijŰçĺŃæĺąåijŔïijŇåőČåřĘæ§Řäÿłæş■ä¡IJè¡ńçğżçżŹåŔęåď ŰäÿĂ æIJĂçőĂå■ŢçŽĎå¡ćåijŔåŔŕèČ¡æŸŕåČŔäÿŃéİćè£ŹæăůïijŽ

```
class A:
    def spam(self, x):
        pass

    def foo(self):
        pass


class B1:
    """çőĂå■ŢçŽĎäżčçŘĘ"""

    def __init__(self):
        self._a = A()

    def spam(self, x):
        # Delegate to the internal self._a instance
        return self._a.spam(x)

    def foo(self):
        # Delegate to the internal self._a instance
        return self._a.foo()

    def bar(self):
        pass
```

åęĆæđIJäżĚäżĚåřśäÿď äÿłæŰźæşŢéIJĂèęĄäżčçŘĘïijŇéĆčäźĹåČŔè£ŹæăůåĘŹåřśèűşåď §äžĘãĂĆä¡ĘæŸ éĆčäźĹä¡£çŤĺ `__getattr__()` æŰźæşŢæĹŰèőÿæĹŰæŻť åě¡äžŻïijŽ

```
class B2:
    """ä¡£çŤĺ__getattr__çŽĎäżčçŘĘïijŇäżčçŘĘæŰźæşŢæŕŤè¿Čåd'ŽæŮűåĂŹ"""

    def __init__(self):
        self._a = A()
```

```
    def bar(self):
        pass

    # Expose all of the methods defined on class A
    def __getattr__(self, name):
        ""
↪"è£ŹäÿłæŰźæşŢåIJĺèő£éŮőçŽĎattributeäÿ■å■ŸåIJĺçŽĎæŮűåĂŹèćńèřČçŢĺ
        the __getattr__() method is actually a fallback method
        that only gets called when an attribute is not found"""
        return getattr(self._a, name)
```

__getattr__ æŰźæşŢæŸŕåIJĺèő£éŮőattributeäÿ■å■ŸåIJĺçŽĎæŮűåĂŹèćńèřČçŢĺïijŇä¡£çŢĺæijŢçd'ži

```
b = B()
b.bar() # Calls B.bar() (exists on B)
b.spam(42) # Calls B.__getattr__('spam') and delegates to A.spam
```

åŔęåd'ŰäÿĂäÿłäżčçŘĘä¿Ńå■ŘæŸŕåőđçŐřäżčçŘĘæĺąåijŔïijŇä¿ŃåęĆïijŽ

```
# A proxy class that wraps around another object, but
# exposes its public attributes
class Proxy:
    def __init__(self, obj):
        self._obj = obj

    # Delegate attribute lookup to internal obj
    def __getattr__(self, name):
        print('getattr:', name)
        return getattr(self._obj, name)

    # Delegate attribute assignment
    def __setattr__(self, name, value):
        if name.startswith('_'):
            super().__setattr__(name, value)
        else:
            print('setattr:', name, value)
            setattr(self._obj, name, value)

    # Delegate attribute deletion
    def __delattr__(self, name):
        if name.startswith('_'):
            super().__delattr__(name)
        else:
            print('delattr:', name)
            delattr(self._obj, name)
```

ä¡£çŢĺè£ŹäÿłäżčçŘĘçśżæŮűïijŇä¡ăåŔłéIJĂèęĄçŢĺåőČæİěåŇĚèčĔäÿŃåĔűäżŰçśżå■şåŔřïijŽ

```
class Spam:
    def __init__(self, x):
        self.x = x
```

```
    def bar(self, y):
        print('Spam.bar:', self.x, y)

# Create an instance
s = Spam(2)
# Create a proxy around it
p = Proxy(s)
# Access the proxy
print(p.x)  # Outputs 2
p.bar(3)  # Outputs "Spam.bar: 2 3"
p.x = 37  # Changes s.x to 37
```

éĂŽè£ĞèĞłåŐžäźL’åśđæĂğèő£éŮőæŰźæşŢïijŇä¡ääŔŕäżěçŤĺäÿ■åŘŇæŰźåijŔèĞłåŐžäźL’äżčçŘĘçśżèąŇ

èőĺèőž

äżčçŘĘçśżæIJL’æŮűåĂŹåŔŕäżěä¡IJäÿžçżğæL’£çŽĎæŻ£äżčæŰźæąĹãĂĆä¿ŃåęĆïijŇäÿĂäÿłçőĂå■ŢçŽĎ

```
class A:
    def spam(self, x):
        print('A.spam', x)
    def foo(self):
        print('A.foo')

class B(A):
    def spam(self, x):
        print('B.spam')
        super().spam(x)
    def bar(self):
        print('B.bar')
```

ä¡£çŤĺäżčçŘĘçŽĎèŕİïijŇåřśæŸŕäÿŃéİćè£ŹæăůïijŽ

```
class A:
    def spam(self, x):
        print('A.spam', x)
    def foo(self):
        print('A.foo')

class B:
    def __init__(self):
        self._a = A()
    def spam(self, x):
        print('B.spam', x)
        self._a.spam(x)
    def bar(self):
        print('B.bar')
    def __getattr__(self, name):
        return getattr(self._a, name)
```

å¡Şåőđ̧ŐřäžčçŘĘæÍąåijŔæŮűïijŇè£ŸæIJL’äžŻçżĘèŁĆéIJĂèęĄæşÍæĎŔãĂĆ éęŰåĚĹïijŇ__getattr__() åőđéŹĚæŸŕäÿĂäÿłåŘŐåđ’ĞæŰźæşŢïijŇåŔłæIJL’åIJÍåśđæĂğäÿ■å■ŸåIJÍæŮűåŻăæ■đ’ïijŇåęĆæđIJäžčçŘĘçśżåőđä¿ŃæIJňèžńæIJL’è£ŹäÿłåśđæĂğçŽĎèŕÏïijŇéĆčäźĹäÿ■äijŽèğęåŔŚè£ŹäÿłåŔęåđ’ŰïijŇ__setattr__() åŠŇ__delattr__() éIJĂèęĄéĆÌåđ’ŰçŽĎé■ŤæşŢæÌěåŇžåĹĘäžčçŘĘåőđ_obj çŽĎåśđæĂğãĂĆ äÿĂäÿłéĂŽåÿÿçŽĎçžęåőŽæŸŕåŔłäžčçŘĘéĆčäžŻäÿ■äžěäÿŃåĹŠçž£_åijĂåđ’t’çŽĎåśđæĂğ(äžčçŘĘçśżåŔłæŽt’éIJšèćńäžčçŘĘçśżçŽĎåĚňåĚśåśđæĂğ)ãĂĆ

è£ŸæIJL’äÿĂçĆźéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijŇ__getattr__() åŕźäžŐåđ’ğéČÍåĹĘäžěåŔŇäÿŃåĹŠçž£(__)åijĂåğŃåŠŇçżŞår¿çŽĎåśđæĂğåźűäÿ■éĂĆçŤÍãĂĆ æŕŤåęĆïijŇèĂČèŹŚåęĆäÿŃçŽĎçśžïijŽ

```
class ListLike:
    """__getattr__
↪åŕźäžŐåŔŇäÿŃåĹŠçž£åijĂåğŃåŠŇçżŞår¿çŽĎæŰźæşŢæŸŕäÿ■èČ¡çŤÍçŽĎïijŇéIJĂèęĄäÿĂäÿłäÿł
↪"""

    def __init__(self):
        self._items = []

    def __getattr__(self, name):
        return getattr(self._items, name)
```

åęĆæđIJæŸŕåĹŻåžžäÿĂäÿłListLikeåŕźèśąïijŇäijŽåŔŞçŐřåőČæŤŕæŇĄæŹőéĂŽçŽĎåĹŮèąÍæŰźæşŢïijŇä¡ĘæŸŕå■t’äÿ■æŤŕæŇĄlen()ãĂĄåĚČçt’äæ§ěæL’¿ç■L’ãĂĆä¿ŃåęĆïijŽ

```
>>> a = ListLike()
>>> a.append(2)
>>> a.insert(0, 1)
>>> a.sort()
>>> len(a)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: object of type 'ListLike' has no len()
>>> a[0]
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: 'ListLike' object does not support indexing
>>>
```

äÿžäžĘèől’åőČæŤŕæŇĄè£ŹäžŻæŰźæşŢïijŇä¡åå£ĚéążæL’ŃåŁÍçŽĎåőđçŐřè£ŹäžŻæŰźæşŢäžčçŘĘïijŽ

```
class ListLike:
    """__getattr__
↪åŕźäžŐåŔŇäÿŃåĹŠçž£åijĂåğŃåŠŇçżŞår¿çŽĎæŰźæşŢæŸŕäÿ■èČ¡çŤÍçŽĎïijŇéIJĂèęĄäÿĂäÿłäÿł
↪"""

    def __init__(self):
        self._items = []

    def __getattr__(self, name):
```

```
        return getattr(self._items, name)

    # Added special methods to support certain list operations
    def __len__(self):
        return len(self._items)

    def __getitem__(self, index):
        return self._items[index]

    def __setitem__(self, index, value):
        self._items[index] = value

    def __delitem__(self, index):
        del self._items[index]
```

11.8åřŔèŁĆè£ŸæIJL'äÿĂäÿłåIJÍè£IJçÍŃæŰźæşŢèřČçŤÍçŐŕåćČäÿ■ä¡£çŤÍäžčçŘEçŽĎä¿Ńå■ŘãĂĆ

8.16 åIJÍçśżäÿ■åőŽäźĽåďŽäÿłæđĎéĂäåŹÍ

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłçśżïijŇéŹď'äžĘä¡£çŤÍ `__init__()` æŰźæşŢåď'ŰïijŇè£ŸæIJL'åĚűäžŰæŰźåijŔåŔŕäżěåĹİåğŃåŇŰåőČãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘåőđçŐřåď'ŽäÿłæđĎéĂäåŹÍïijŇä¡ăéIJĂèęĄä¡£çŤÍåĹřçśżæŰźæşŢãĂĆä¿ŃåęĆïijŽ

```
import time

class Date:
    """æŰźæşŢäÿĂïijžä¡£çŤÍçśżæŰźæşŢ"""
    # Primary constructor
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day

    # Alternate constructor
    @classmethod
    def today(cls):
        t = time.localtime()
        return cls(t.tm_year, t.tm_mon, t.tm_mday)
```

çŻť'æŐěèřČçŤÍçśżæŰźæşŢå■şåŔŕïijŇäÿŃéİćæŸŕä¡£çŤÍçđ'žä¿Ńïijž

```
a = Date(2012, 12, 21) # Primary
b = Date.today() # Alternate
```

èőÍèőž

çśżæŰźæşŢçŽĎäÿĂäÿłäÿżèęAçŤĺéĂŤåřśæŸŕåőŽäźL’åd’ŽäÿłæđĎéĂäåŹĺãĂĆåőČæŐěåŔŮäÿĂäÿł class ä¡IJäÿžçńňäÿĂäÿłåŔĆæŢř(cls)ãĂĆ ä¡ăåžŤèřěæşĺæĎŔåĹřäžĘè£Źäÿłçśżèćńç ŤĺæÌěåĹŽåżžåźűè£ŤåŻđa

```
class NewDate(Date):
    pass

c = Date.today() # Creates an instance of Date (cls=Date)
d = NewDate.today() # Creates an instance of NewDate (cls=NewDate)
```

8.17 åĹŻåżžäÿ■èřČçŤĺnitæŰźæşŢçŽĎåőđä¿Ń

éŮőécŸ

ä¡ăæČşåĹŽåżžäÿĂäÿłåőđä¿ŃïijŇä¡ĘæŸŕåÿŇæIJŻçżŢè£ĞæL’ğèąŇ __init__() æŰźæşŢãĂĆ

èğčåĘşæŰźæąĹ

åŔŕäżěéĂŽè£Ğ __new__() æŰźæşŢåĹŽåżžäÿĂäÿłæIJłåĹİåğŇåŇŰçŽĎåőđä¿ŃãĂĆä¿ŃåęĆèĂČèŹŚåę

```
class Date:
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day
```

äÿŃéİćæijŤçd’žåęĆä¡Ţäÿ■èřČçŤĺ __init__() æŰźæşŢæİěåĹŽåżžè£ŹäÿłDateåőđä¿ŃïijŽ

```
>>> d = Date.__new__(Date)
>>> d
<__main__.Date object at 0x1006716d0>
>>> d.year
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: 'Date' object has no attribute 'year'
>>>
```

çżŞæđIJåŔŕäżěçIJŃåĹřïijŇè£ŹäÿłDateåőđä¿ŃçŽĎåśđæĂğyearè£Ÿäÿ■å■ŸåIJĺïijŇæL’Ääžěä¡ăéIJĂèęĄæ

```
>>> data = {'year':2012, 'month':8, 'day':29}
>>> for key, value in data.items():
...     setattr(d, key, value)
...
```

```
>>> d.year
2012
>>> d.month
8
>>>
```

èőÍèőž

å¡ŞæĹŚäžňåIJĺåŔ■åžŔåĹŮåŕźèśąæĹŰèĂĚåőđçŐřæ§Řäÿłçśżæ Ű źæş Ţæđ Đé Ăăå Ğ¡æŢřæŮűéIJĂèęĄçżŢè
`__init__()` æŰźæş Ţæİěå ĹŻåžžåŕźèśąãĂĆ ä¿ŃåęĆïijŇåŕźäžŐäÿŁéİćçŽĎDateæİěèőšïijŇæIJL'æŮűåĂŹä¡ă
`today()` ïijŽ

```
from time import localtime

class Date:
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day

    @classmethod
    def today(cls):
        d = cls.__new__(cls)
        t = localtime()
        d.year = t.tm_year
        d.month = t.tm_mon
        d.day = t.tm_mday
        return d
```

åŘŇæăůïijŇåIJĺä¡ăåŔ■åžŔåĹŮåŇŰJSONæŢřæ■őæŮűäžğçŢ§äÿĂäÿłåęĆäÿŃçŽĎå■ŮåĚÿåŕźèśąïijŽ

```
data = { 'year': 2012, 'month': 8, 'day': 29 }
```

åęĆæđIJä¡ăæČşåřĘåőČè¡ňæ■ćæĹŘäÿĂäÿłDateçśżåđŃåőđä¿ŃïijŇåŔŕäžěä¡£çŢĺäÿŁéİćçŽĎæŁĂæIJŕãĂĆ

å¡Şä¡ăéĂŽè£Ğè£Źçğ■éİđåÿÿèğĎæŰźåijŔæİěåĹŻåžžåőđä¿ŃçŽĎæŮűåĂŹïijŇæIJĂåě¡äÿ■èęĄçŽť'æŐěåĆ
åŘęåĹŹçŽĎèŕİïijŇåęĆæđIJè£Źäÿłçśżä¡£çŢĺäžĘ `__slots__` ãĂĄproperties ãĂĄdescriptors æĹŰåĚűäžŰéńŸçžğæŁĂæIJŕçŽĎæŮűåĂŹäžčçăĄåřšäijŽåď śæŢĹãĂĆ
èĂŇè£ŹæŮűåĂŹä¡£çŢĺ `setattr()` æŰźæŞäijŽèőľ'ä¡äçŽĎäžčçăĄåŔŸå¿ŮæŽť'åŁäéĂŽçŢĺãĂĆ

8.18 åĹľçŢĺMixinsæL'ľåśŢçśżåŁ§èČ¡

éŮőécŸ

ä¡ăæIJL'å¿Ĺåď ŽæIJL'çŢĺçŽĎæŰźæş Ţïij ŇæČşä¡£çŢĺåőČäžňæİěæL'ľåśŢåĚűäžŰçśżçŽĎåŁ§èČ¡ãĂĆä¡Ęą
åŽăæ■ď'ä¡ääÿ■èČ¡çőĂå■ŢçŽĎåřĘè£ŹäžŽæŰźæş ŢæŤ¿åĚëäÿĂäÿłå§žçśżïijŇçĎůåŘŐècńåĚűäžŰçśżçžğæL'£ã

èğčåĘşæŰźæąĹ

éĂŽåÿÿå¡Şä¡ăæČşèĞłåőŽäźL’çśżçŽĎæŮűåĂŹäijŽçćřäÿŁè£ŹäžŽéŮőécŸãĂĆåŕřèČ¡æŸŕæ§Řäÿłåž
ä¡ååŕřäżěåĹl’çŤĺåőČäżňæİěæđĎéĂää¡äèĞłåůśçŽĎçśżãĂĆ

åĄĞèő¿ä¡ăæČşæĽl’åśŢæŸäářĎåŕźèšąïijŇçżŹåőČäżňæůżåŁăæŮěå£ŮãĂĄåŤŕäÿĂæĂğèő¿ç¡õãĂĄçśżåđ

```
class LoggedMappingMixin:
    """
    Add logging to get/set/delete operations for debugging.
    """
    __slots__ = ()  #␣
↪æůůåĚěçśżéČ¡æšąæIJL’åőđä¿ŃåŔŸéĞŔïijŇåžääÿžçŻt’æŐěåőđä¿ŃåŇŰæůůåĚěçśżæšąæIJL’äżż

    def __getitem__(self, key):
        print('Getting ' + str(key))
        return super().__getitem__(key)

    def __setitem__(self, key, value):
        print('Setting {} = {!r}'.format(key, value))
        return super().__setitem__(key, value)

    def __delitem__(self, key):
        print('Deleting ' + str(key))
        return super().__delitem__(key)


class SetOnceMappingMixin:
    '''
    Only allow a key to be set once.
    '''
    __slots__ = ()

    def __setitem__(self, key, value):
        if key in self:
            raise KeyError(str(key) + ' already set')
        return super().__setitem__(key, value)


class StringKeysMappingMixin:
    '''
    Restrict keys to strings only
    '''
    __slots__ = ()

    def __setitem__(self, key, value):
        if not isinstance(key, str):
            raise TypeError('keys must be strings')
        return super().__setitem__(key, value)
```

è£ŹäžŽçśżå■ŢçŃñä¡£çŤĺèţůæİěæšąæIJL’äżżä¡ŢæĎŔäźL’ïijŇäžŃåőđäÿŁåęĆæđIJä¡ääŐżåőđä¿ŃåŇŰäżżä

åőČäžňæŸŕçŤĺæİěéĂŽè£Ğåď'ŽçżğæL'£æİěåŠŇåĚűäžŰæŸăářĎåŕźèśąæůůåĚěä¡£çŤĺçŽĎãĂĆä¿ŃåęĆïijŽ

```
class LoggedDict(LoggedMappingMixin, dict):
    pass

d = LoggedDict()
d['x'] = 23
print(d['x'])
del d['x']

from collections import defaultdict

class SetOnceDefaultDict(SetOnceMappingMixin, defaultdict):
    pass


d = SetOnceDefaultDict(list)
d['x'].append(2)
d['x'].append(3)
# d['x'] = 23  # KeyError: 'x already set'
```

è£Źäýłä¿Ńå■Řäÿ■ïijŇåŔŕäžěçIJŃåĹřæůůåĚěçśżèů§åĚűäžŰåůšå■ŸåIJĺçŽĎçśż(æŕŤåęĆdictãĂĄdefaultd
çżŞåŘĹåŘŐåŕśèČ¡åŔŚæŇěæ■čåÿÿåŁ§æŢĹäžĘãĂĆ

èőÍèőž

æůůåĚěçśżåIJÍæăĞåĞĘåžŞäÿ■å¿Ĺåď'ŽåIJřæŰźéČ¡åĞžçŐřè£ĞïijŇéĂŽåÿÿéČ¡æŸŕçŤĺæİěåČŔäýŁéİćéĆ
åőČäžňäź§æŸŕåď'ŽçżğæL'£çŽĎäÿĂäýłäÿżèęĄçŤĺéĂŤãĂĆæŕŤåęĆïijŇå¡Şä¡äçijŰåĘŹç¡ŚçżIJäžčçăĄæŮűåĂŹ
ä¡ääijŽçżŔåÿÿä¡£çŤĺ `socketserver` æĺąåİŮäÿ■çŽĎ `ThreadingMixIn`
æİěçżŹåĚűäžŰç¡ŚçżIJçŽÿåĘşçśżåćđåŁäåď'Žçž£çĺŇæŤŕæŇĄãĂĆ
ä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿĂäýłåď'Žçž£çĺŃçŽĎXML-RPCæIJ■åŁąïijŽ

```
from xmlrpc.server import SimpleXMLRPCServer
from socketserver import ThreadingMixIn
class ThreadedXMLRPCServer(ThreadingMixIn, SimpleXMLRPCServer):
    pass
```

åŘŇæŮűåIJÍäÿĂäžŻåď'ğåđŃåžŞåŠŇæąĘæđűäÿ■äź§äijŽåŔŚçŐřæůůåĚěçśżçŽĎä¡£çŤĺïijŇçŤĺéĂŤåŘŇæ

åŕźäžŐæůůåĚěçśżïijŇæIJL'åĞăçĆźéIJĂèęĄèőřä¡ŔãĂĆéęŰåĚĹæŸŕïijŇæůůåĚěçśżäÿ■èČ¡çŻť'æŐèèćńåő
åĚűæňąïijŇæůůåĚěçśżæšąæIJL'èĞłåůśçŽĎçŁűæĂĄä£ąæĄŕïijŇäź§åřśæŸŕèŕť'åőČäžňåźűæšąæIJL'åőŽäźL'
`__init__()` æŰźæşŢïijŇåźűäÿŤæšąæIJL'åőđä¿ŃåšđæĂğãĂĆ
è£Źäź§æŸŕäÿžäžĂäźĹæĹŚäžňåIJÍäýŁéİćæŸŐçąőåőŽäźL'äžĘ `__slots__ = ()` ãĂĆ

è£ŸæIJL'äÿĂçğ■åőđçŐřæůůåĚěçśżçŽĎæŰźåijŔåřśæŸŕä¡£çŤĺçśżèčĚéěřåŹĺïijŇåęĆäÿŃæL'Ăçd'žïijŽ

```
def LoggedMapping(cls):
    """çňňäžŇçğ■æŰźåijŔïijžä¡£çŤĺçśżèčĚéěřåŹĺ"""
    cls_getitem = cls.__getitem__
    cls_setitem = cls.__setitem__
```

```
    cls_delitem = cls.__delitem__

    def __getitem__(self, key):
        print('Getting ' + str(key))
        return cls_getitem(self, key)

    def __setitem__(self, key, value):
        print('Setting {} = {!r}'.format(key, value))
        return cls_setitem(self, key, value)

    def __delitem__(self, key):
        print('Deleting ' + str(key))
        return cls_delitem(self, key)

    cls.__getitem__ = __getitem__
    cls.__setitem__ = __setitem__
    cls.__delitem__ = __delitem__
    return cls


@LoggedMapping
class LoggedDict(dict):
    pass
```

è£Źäÿłæ Ţĺæđ IJè ů§äź Ńå L’■çŽĎæŸŕäÿĂæăůçŽĎïijŇèĂŇäÿŤäÿ■åĘ■éIJĂèęĄä¡£çŤĺåď’ŽçżğæL’£äžĘãĂĆ
åŘĆèĂČ8.13åřŔèŁĆæ§ěçIJŃæŻť åď’ŽæůůåĕěçśžåŠŇçśżèčĔéěřåŹĺçŽĎä¿Ńå■ŘãĂĆ

8.19 åőđçŐřçŁűæĂĄåŕźèśąæĹŰèĂĔçŁűæĂĄæIJž

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłçŁűæĂĄæIJžæĹŰèĂĔæŸŕåIJĺäÿ■åŘŇçŁűæĂĄäÿŃæL’ğèąŇæŞ■ä¡IJçŽĎåŕźèśąïijI

èğčåĘşæŰźæąĹ

åIJĺå¿Ĺåď’ŽçĺŃåžŔäÿ■ïijŇæIJL’äžŻåŕźèśąäijŽæăźæ■őçŁűæĂĄçŽĎäÿ■åŘŇæİěæL’ğèąŇäÿ■åŘŇçŽĎæŞ

```
class Connection:
    """æŹőéĂŽæŰźæąĹïijŇåě¡åd’Žäÿłåĺd’æŰ■èŕ■åŔěïijŇæŢĹçŐĞä¡ŐäÿŃ~~"""

    def __init__(self):
        self.state = 'CLOSED'

    def read(self):
        if self.state != 'OPEN':
            raise RuntimeError('Not open')
        print('reading')
```

```
    def write(self, data):
        if self.state != 'OPEN':
            raise RuntimeError('Not open')
        print('writing')

    def open(self):
        if self.state == 'OPEN':
            raise RuntimeError('Already open')
        self.state = 'OPEN'

    def close(self):
        if self.state == 'CLOSED':
            raise RuntimeError('Already closed')
        self.state = 'CLOSED'
```

è£Źæăůå ĘŹæIJL’å¿Ĺåd’Žçijžç ĆźïijŇéęŰåĚĹæŸŕäžčçăĄåd’łåd’■æİĆäžĘïijŇåě¡åd’ŽçŽĎæİąäżűåĹd’æŰ■åŻääÿžäÿĂäžŻåÿÿèğĄçŽĎæŞ■ä¡IJæŕŤåęĆread()ãĂĄwrite()æŕŔæňąæL’ğèąŇåL’■éČ¡éIJĂèęĄæL’ğèąŇæčĂæş

äÿĂäÿłæŻť åě¡çŽĎåŁđæşŢæŸŕäÿžæŕŔäÿłçŁűæĂĄåőŽäźL’äÿĂäÿłåŕźèśąïijŽ

```
class Connection1:
    """æŰřæŰźæąĹâĂŤâĂŤåŕźæŕŔäÿłçŁűæĂĄåőŽäźL’äÿĂäÿłçśż"""

    def __init__(self):
        self.new_state(ClosedConnectionState)

    def new_state(self, newstate):
        self._state = newstate
        # Delegate to the state class

    def read(self):
        return self._state.read(self)

    def write(self, data):
        return self._state.write(self, data)

    def open(self):
        return self._state.open(self)

    def close(self):
        return self._state.close(self)


# Connection state base class
class ConnectionState:
    @staticmethod
    def read(conn):
        raise NotImplementedError()

    @staticmethod
    def write(conn, data):
```

```
        raise NotImplementedError()

    @staticmethod
    def open(conn):
        raise NotImplementedError()

    @staticmethod
    def close(conn):
        raise NotImplementedError()


# Implementation of different states
class ClosedConnectionState(ConnectionState):
    @staticmethod
    def read(conn):
        raise RuntimeError('Not open')

    @staticmethod
    def write(conn, data):
        raise RuntimeError('Not open')

    @staticmethod
    def open(conn):
        conn.new_state(OpenConnectionState)

    @staticmethod
    def close(conn):
        raise RuntimeError('Already closed')


class OpenConnectionState(ConnectionState):
    @staticmethod
    def read(conn):
        print('reading')

    @staticmethod
    def write(conn, data):
        print('writing')

    @staticmethod
    def open(conn):
        raise RuntimeError('Already open')

    @staticmethod
    def close(conn):
        conn.new_state(ClosedConnectionState)
```

äÿŃéÌćæŸŕä¡£çŤĺæijŤçď˙žïijŽ

```
>>> c = Connection()
>>> c._state
<class '__main__.ClosedConnectionState'>
>>> c.read()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example.py", line 10, in read
        return self._state.read(self)
    File "example.py", line 43, in read
        raise RuntimeError('Not open')
RuntimeError: Not open
>>> c.open()
>>> c._state
<class '__main__.OpenConnectionState'>
>>> c.read()
reading
>>> c.write('hello')
writing
>>> c.close()
>>> c._state
<class '__main__.ClosedConnectionState'>
>>>
```

èőÍèőž

åęĆæđIJäżčçăĄäÿ■åĞžçŐřåď'łåď'ŽçŽĎæİąäżűåĹď'æŰ■èŕ■åŔěçŽĎèŕİïijÑäżčçăĄåřśäijŽåŔŸå¿ŮéŽ¿äżè
è£ŹéĞŇçŽĎèğčåĘşæŰźæąĹæŸŕářĘæŕŔäÿłçŁűæĂĄæŁ¡åŔŰåĞžæİěåőŽäźL'æĹŘäÿĂäÿłçśžãĂĆ

è£ŹéĞŇçIJŃäÿŁåŐżæIJL'çĆźåěĞæĂłïijÑæŕŔäÿłçŁűæĂĄåŕźèśąéČ¡åŔłæIJL'éİŹæĂĄæŰźæşŢïijÑåźűæ
åőđéŹĚäÿŁïijÑæL'ĂæIJL'çŁűæĂĄä£ąæĄŕéČ¡åŔłå■ŸåĆĺåIJĺ Connection
åőđä¿Ńäÿ■ãĂĆ åIJĺå§žçśżäÿ■åőŽäźL'çŽĎ NotImplementedError
æŸŕäÿžäžĘçąőä£İå■ŘçśżåőđçŐřäžĘçŻÿåžŢçŽĎæŰźæşŢãĂĆ è£ŹéĞŇä¡ăæĹŰèőÿè£ŸæČşä¡£çŢĺ8.12åřŔèŁĆ

èő¿èőąæĺąåijŔäÿ■æIJL'äÿĂçğ■æĺąåijŔåŔńçŁűæĂĄæĺąåijŔïijÑè£ŹäÿĂåřŔèŁĆçőŮæŸŕäÿĂäÿłåĹİæ■ěå

8.20 éĂŽè£Ğå■ŮçňęäÿšèřČçŢĺåŕźèśąæŰźæşŢ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłå■Ůçňęäÿšå¡ćåijŔçŽĎæŰźæşŢåŘ■çğřïijÑæČşéĂŽè£ĞåőČèřČçŢĺæ§ŔäÿłåŕźèśąçŽĎåŕźå

èğčåĘşæŰźæąĹ

æIJĂçőĂå■ŢçŽĎæČĚåĘţïijÑåŔŕäżěä¡£çŢĺ `getattr()` ïijŽ

```
import math

class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __repr__(self):
        return 'Point({!r:},{!r:})'.format(self.x, self.y)

    def distance(self, x, y):
        return math.hypot(self.x - x, self.y - y)


p = Point(2, 3)
d = getattr(p, 'distance')(0, 0)  # Calls p.distance(0, 0)
```

åŘęåď ŰäÿĂçğ■æŰźæşŢæŸŕä¡£çŤÍ `operator.methodcaller()` ïijŇä¿ŃåęĆïijŽ

```
import operator
operator.methodcaller('distance', 0, 0)(p)
```

å¡Şä¡ăéIJĂèęĄéĂŽè£ĞçŻÿåŘŇçŽĎåŔĆæŢřåď Žæňąèřĉç ŤÍæ§ŘäÿłæŰźæşŢæŮűïijŇä¡£çŤÍ `operator.methodcaller` åřśå¿ĹæŰźä¿£ăžĘãĂĆ æŕŤåęĆä¡ăéIJĂèęĄæŐŠåžŔäÿĂçşżåĹŮçŽĎçĆźïijŇåř

```
points = [
    Point(1, 2),
    Point(3, 0),
    Point(10, -3),
    Point(-5, -7),
    Point(-1, 8),
    Point(3, 2)
]
# Sort by distance from origin (0, 0)
points.sort(key=operator.methodcaller('distance', 0, 0))
```

èőÍèőž

èřČçŤÍäÿĂäÿłæŰźæşŢåőđéŹĚäÿŁæŸŕäÿď éČÍçŃňçńŃæŞ■ä¡IJïijŇçňňäÿĂæ■ěæŸŕæ§ěæĽ¿åśđæĂğïijŇçåŽăæ■d'ïijŇäÿžäžĘèřČçŤÍæ§ŘäÿłæŰźæşŢïijŇä¡ăåŔŕäžěéęŰåĚĹéĂŽè£Ğ `getattr()` æİěæ§ěæĽ¿åĹřè£ŹäÿłåśđæĂğïijŇçĎűåŘŐåĘ■åŐżäžěåĞ¡æŢřæŰźåijŔèřČçŤÍåőČå■şåŔŕãĂĆ

`operator.methodcaller()` åĹŻåžžäÿĂäÿłåŔŕèřČçŤÍåŕźèśąïijŇåźűåŘŇæŮűæŔŘä¿ŻæĽĂæIJL'å£çĎűåŘŐèřČçŤÍçŽĎæŮűåĂŹåŔłéIJĂèęĄåřĘåőđä¿ŃåŕźèśąäijăéĂŠçżŹåőČå■şåŔŕïijŇæŕŤåęĆïijŽ

```
>>> p = Point(3, 4)
>>> d = operator.methodcaller('distance', 0, 0)
>>> d(p)
```

```
5.0
>>>
```

éĂŽè£ĞæŰźæşŢåŘ■çğřå■ŮçňęäÿšæİěèřČçŤĺæŰźæşŢéĂŽåÿÿåĞžçŐřåIJĺéIJĂèęĄæĺąæŃ§ case èŕ■åŘěæĹŰåőđçŐřèő£éŮőèĂĚæĺąåijŔçŽĎæŮűåĂŹãĂĆ åŔĆèĂČäÿŃäÿĂåřŔèŁĆèŐůåŔŰæŻt'åd'ŽéńŸçžğä¿Ńå■ŘãĂĆ

8.21 åőđçŐřèő£éŮőèĂĚæĺąåijŔ

éŮőécŸ

ä¡ăèęĄåd'ĎçŘĘçŤśåd'ğéĞŔäÿ■åŘŃçśżåđŃçŽĎåŕźèśąçżĎæĹŘçŽĎåd'■æİĆæŢřæ■őçżŞæđĎïijŇæŕŔäÿĂ æŕŤåęĆïijŇéĄ■åŐĘäÿĂäÿłæăŚå¡ćçżŞæđĎïijŇçĎűåŘŐæăźæ■őæŕŔäÿłèŁĆçĆźçŽĎçŻÿåžŤçŁűæĂĄæL'ğèąŇ

èğčåĘşæŰźæąĹ

è£ŹéĞŇéĄĞåĹřçŽĎéŮőécŸåIJĺçijŰçĺŃécĘå§§äÿ■æŸŕå¿ĹæŹőéĄ■çŽĎïijŇæIJL'æŮűåĂŹäijŽæđĎåżžä åĄĞèő¿ä¡ăèęĄåĘŹäÿĂäÿłèąĺçd'žæŢřå■ęèąĺè¿¿åijŔçŽĎçĺŃåžŔïijŇéĆčäźĹä¡ăåŔřèČ¡éIJĂèęĄåőŽäźL'åęĆäÿŃ

```
class Node:
    pass

class UnaryOperator(Node):
    def __init__(self, operand):
        self.operand = operand

class BinaryOperator(Node):
    def __init__(self, left, right):
        self.left = left
        self.right = right

class Add(BinaryOperator):
    pass

class Sub(BinaryOperator):
    pass

class Mul(BinaryOperator):
    pass

class Div(BinaryOperator):
    pass

class Negate(UnaryOperator):
    pass

class Number(Node):
```

```
    def __init__(self, value):
        self.value = value
```

çĎűåŘŐåĹľçŤĺè£ŹäžŽçśżæđĎåżžåţŇåěŮæŢřæ■őçżŞæđĎïijŇåęĆäÿŃæĽĂçđ'žïijŽ

```
# Representation of 1 + 2 * (3 - 4) / 5
t1 = Sub(Number(3), Number(4))
t2 = Mul(Number(2), t1)
t3 = Div(t2, Number(5))
t4 = Add(Number(1), t3)
```

è£ŹæăůåĄŽçŽĎéŮőécŸæŸřåŕžäžŐæŕŘäÿłèąĺè¿¿åijŔïijŇæŕŘæňąéČ¡èęĄéĞ■æŰřåőŽäźĽäÿĂéĄ■ïijŇæl
è£ŹéĞŇæĹŚäžňä¡£çŤĺèő£éŮőèĂĚæĺąåijŔåŘŕäżěè¿¿åĹřè£ŹæăůçŽĎçŽőçŽĎïijŽ

```
class NodeVisitor:
    def visit(self, node):
        methname = 'visit_' + type(node).__name__
        meth = getattr(self, methname, None)
        if meth is None:
            meth = self.generic_visit
        return meth(node)

    def generic_visit(self, node):
        raise RuntimeError('No {} method'.format('visit_' +␣
↪type(node).__name__))
```

äÿžäžĘä¡£çŤĺè£ŹäÿłçśżïijŇåŘŕäżěåőŽäźĽäÿĂäÿłçśżçżğæĽ£åőČåźűäÿŤåőđçŐřåŘĎçğ■
visit_Name() æŰźæşŢïijŇåĚűäÿ■NameæŸŕnodeçśżåđŃãĂĆ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăæČşæśĆèąĺè¿¿åijŔçŽĎåĂijïijŇåŘŕäżěè£ŹæăůåĘŹïijŽ

```
class Evaluator(NodeVisitor):
    def visit_Number(self, node):
        return node.value

    def visit_Add(self, node):
        return self.visit(node.left) + self.visit(node.right)

    def visit_Sub(self, node):
        return self.visit(node.left) - self.visit(node.right)

    def visit_Mul(self, node):
        return self.visit(node.left) * self.visit(node.right)

    def visit_Div(self, node):
        return self.visit(node.left) / self.visit(node.right)

    def visit_Negate(self, node):
        return -node.operand
```

ä¡£çŤĺçd'žä¿ŃïijŽ

```
>>> e = Evaluator()
>>> e.visit(t4)
0.6
>>>
```

ä¡IJäÿžäÿĂäÿłäÿ■åŘŇçŽĎä¿Ńå■ŘïijŇäÿŃéİćåőŽäźL'äÿĂäÿłçśžåIJÍäÿĂäÿłæăĹäÿŁéİćářĘäÿĂäÿłèąÍè¿¿å

```
class StackCode(NodeVisitor):
    def generate_code(self, node):
        self.instructions = []
        self.visit(node)
        return self.instructions

    def visit_Number(self, node):
        self.instructions.append(('PUSH', node.value))

    def binop(self, node, instruction):
        self.visit(node.left)
        self.visit(node.right)
        self.instructions.append((instruction,))

    def visit_Add(self, node):
        self.binop(node, 'ADD')

    def visit_Sub(self, node):
        self.binop(node, 'SUB')

    def visit_Mul(self, node):
        self.binop(node, 'MUL')

    def visit_Div(self, node):
        self.binop(node, 'DIV')

    def unaryop(self, node, instruction):
        self.visit(node.operand)
        self.instructions.append((instruction,))

    def visit_Negate(self, node):
        self.unaryop(node, 'NEG')
```

ä¡£çŤĺçd'žä¿ŃïijŽ

```
>>> s = StackCode()
>>> s.generate_code(t4)
[('PUSH', 1), ('PUSH', 2), ('PUSH', 3), ('PUSH', 4), ('SUB',),
('MUL',), ('PUSH', 5), ('DIV',), ('ADD',)]
>>>
```

èőÍèőž

åĹŽåijĂåğŃçŽĎæŮűåĂŹä¡ăåŔřèČ¡äijŽåĘŹåď ğéĞŘçŽĎif/elseèŕ■åŘěæÌěåőđçŐřïijŇ è£ŹéĞŇèő£éŮőèĂĚæÍąåijŘçŽĎåě¡åď ĎåřśæŸŕé ĂŽè£Ğ `getattr()` æÌěèŐůåŔŰçŻÿåžŤçŽĎæŰźæşŢïijŇåźűåĹľçŤĺéĂŠå¡ŠæÌěéĄ■åŐĘæĽĂæIJL'çŽĎèŁĆçĆźïijŽ

```
def binop(self, node, instruction):
    self.visit(node.left)
    self.visit(node.right)
    self.instructions.append((instruction,))
```

è£ŸæIJL'äÿĂçĆźéIJĂèęĄæŇĞåĞžçŽĎæŸŕïijŇè£Źçğ■æŁĂæIJŕäź§æŸŕåőđçŐřåĚűäżŰèŕ■èÍĂäÿ■switch æŕŤåęĆïijŇåęĆæđIJä¡ăæ■čåIJĺåĘŹäÿĂäÿłHTTPæąĘæđűïijŇä¡ăåŔřèČ¡äijŽåĘŹè£Źæăůäÿ ÄäÿłèŕůæśĆåĹĘåŔ

```
class HTTPHandler:
    def handle(self, request):
        methname = 'do_' + request.request_method
        getattr(self, methname)(request)
    def do_GET(self, request):
        pass
    def do_POST(self, request):
        pass
    def do_HEAD(self, request):
        pass
```

è ő£éŮőèĂĚæÍąåijŔäÿĂäÿłçijžçĆźåřśæŸŕåőČäÿěéĞ■ä¿İèţŰéĂŠå¡ŠïijŇåęĆæđIJæŢřæ■őçżŞæđĎåţŇåěŮ æIJL'æŮůåĂŹäijŽèűĚè£ĞPythonçŽĎéĂŠå¡Šæůśåžęé ŹŘåĹű(åŔĆèĂČ `sys.getrecursionlimit()`)ãĂĆ

åŔŕäžěåŔĆçĚğ8.22åřŔèŁĆïijŇåĹľçŤĺçŤ§æĹŘåŹĺæĹŰè£■äžčåŹĺæÌěåőđçŐřéİđéĂŠå¡ŠéĄ■åŐĘçőŮæşŢ

åIJĺèů§èğčæđŘåŠŇçijŰèŕŚçŻÿåĚşçŽĎçijŰçĺŃäÿ■ä¡£çŤĺèő£éŮőèĂĚæÍąåijŔæŸŕéİđåÿÿåÿÿèğĄçŽĎãĂĆ PythonæIJňěžńçŽĎ `ast` æÍąåİŮåĂijçŽĎåĚşæşÍäÿŃïijŇåŔŕäžěåŐžçIJŃçIJŃæžŘçăĄãĂĆ 9.24åřŔèŁĆæijŤçď žäžĘäÿĂäÿłåĹľçŤĺ `ast` æÍąåİŮæÌěåď ĎçŘĘPythonæžŘäžčçăĄçŽĎä¿Ńå■ŘãĂĆ

8.22 äÿ■çŤĺéĂŠå¡ŠåőđçŐřèő£éŮőèĂĚæÍąåijŔ

éŮőécŸ

ä¡ăä¡£çŤĺèő£éŮőèĂĚæÍąåijŔéĄ■åŐĘäÿĂäÿłå¿ĹæůśçŽĎåţŇåěŮæăŠå¡ćæŢřæ■őçżŞæđĎïijŇåźűäÿŤåŻäă ä¡ăæČşæűĹéŹď éĂŠå¡ŠïijŇåźűåŘŇæŮűä£ÌæŇĄè£éŮőèĂĚçijŰçĺŃæÍąåijŔãĂĆ

èğčåĘşæŰźæąĹ

éĂŽè£Ğåůğåę ŹçŽĎä¡£çŤĺçŤ§æĹŘåŹĺåŔŕäžěåIJÍæăŚéĄ■åŐĘæĽŰæŘIJçţ'ćçőŮæşŢäÿ■æűĹéŹď éĂŠå¡Š åIJÍ8.21åřŔèŁĆäÿ■ïijŇæĹŚäžňçżŹåĞžäžĘäÿĂäÿłèő£éŮőèĂĚçśżãĂĆ äÿŃéÌćæĹŚäžňåĹľçŤĺäÿĂäÿłæăĹåŠŇçŤ§æĹŘåŹĺéĞ■æŰřåőđçŐřè£Źäÿłçśżïijž

```
import types

class Node:
    pass

class NodeVisitor:
    def visit(self, node):
        stack = [node]
        last_result = None
        while stack:
            try:
                last = stack[-1]
                if isinstance(last, types.GeneratorType):
                    stack.append(last.send(last_result))
                    last_result = None
                elif isinstance(last, Node):
                    stack.append(self._visit(stack.pop()))
                else:
                    last_result = stack.pop()
            except StopIteration:
                stack.pop()

        return last_result

    def _visit(self, node):
        methname = 'visit_' + type(node).__name__
        meth = getattr(self, methname, None)
        if meth is None:
            meth = self.generic_visit
        return meth(node)

    def generic_visit(self, node):
        raise RuntimeError('No {} method'.format('visit_' +␣
↪type(node).__name__))
```

åęĆædIJä¡ăä¡£çŤÍè£ŹäÿłçśżïijŇäź§èČ¡è¿¿åĹřçŻÿåŘŇçŽĎæŢĹædIJãĂĆäžŇåőđäÿŁä¡ăåőŇåĚÍåŔŕäżěåřĿ
èĂČèŹŚåęĆäÿŇäžčçăĄïijŇéĄ■åŐĘäÿĂäÿłèąĺè¿¿åijŔçŽĎæăŚïijŽ

```
class UnaryOperator(Node):
    def __init__(self, operand):
        self.operand = operand

class BinaryOperator(Node):
    def __init__(self, left, right):
        self.left = left
        self.right = right

class Add(BinaryOperator):
    pass
```

```
class Sub(BinaryOperator):
    pass

class Mul(BinaryOperator):
    pass

class Div(BinaryOperator):
    pass

class Negate(UnaryOperator):
    pass

class Number(Node):
    def __init__(self, value):
        self.value = value

# A sample visitor class that evaluates expressions
class Evaluator(NodeVisitor):
    def visit_Number(self, node):
        return node.value

    def visit_Add(self, node):
        return self.visit(node.left) + self.visit(node.right)

    def visit_Sub(self, node):
        return self.visit(node.left) - self.visit(node.right)

    def visit_Mul(self, node):
        return self.visit(node.left) * self.visit(node.right)

    def visit_Div(self, node):
        return self.visit(node.left) / self.visit(node.right)

    def visit_Negate(self, node):
        return -self.visit(node.operand)

if __name__ == '__main__':
    # 1 + 2*(3-4) / 5
    t1 = Sub(Number(3), Number(4))
    t2 = Mul(Number(2), t1)
    t3 = Div(t2, Number(5))
    t4 = Add(Number(1), t3)
    # Evaluate it
    e = Evaluator()
    print(e.visit(t4))  # Outputs 0.6
```

åęĆædIJåţŇåěŮåśĆæňąåd'łæůśéĆčäźĽäÿŁè£řçŽĎEvaluatorårřšäijŽåd'śæŢĹïijŽ

```
>>> a = Number(0)
>>> for n in range(1, 100000):
```

```
... a = Add(a, Number(n))
...
>>> e = Evaluator()
>>> e.visit(a)
Traceback (most recent call last):
...
    File "visitor.py", line 29, in _visit
return meth(node)
    File "visitor.py", line 67, in visit_Add
return self.visit(node.left) + self.visit(node.right)
RuntimeError: maximum recursion depth exceeded
>>>
```

çŐřåIJÍæĹŚäżňçĺ■å¿őä£őæŤźäÿŃäÿŁéİćçŽĎEvaluatorïijŽ

```
class Evaluator(NodeVisitor):
    def visit_Number(self, node):
        return node.value

    def visit_Add(self, node):
        yield (yield node.left) + (yield node.right)

    def visit_Sub(self, node):
        yield (yield node.left) - (yield node.right)

    def visit_Mul(self, node):
        yield (yield node.left) * (yield node.right)

    def visit_Div(self, node):
        yield (yield node.left) / (yield node.right)

    def visit_Negate(self, node):
        yield - (yield node.operand)
```

åĘ■æňąè£ŘèąŇïijŇářśäÿ■äijŽæŁěéŤŹäžĘïijŽ

```
>>> a = Number(0)
>>> for n in range(1,100000):
...     a = Add(a, Number(n))
...
>>> e = Evaluator()
>>> e.visit(a)
4999950000
>>>
```

åęĆæđIJä¡ăè£ŸæČşæůżåŁăåĚűäżŰèĞłåőŽäźĽ'éĂżè¿Śäź§æšąéŮőécŸïijŽ

```
class Evaluator(NodeVisitor):
    ...
    def visit_Add(self, node):
        print('Add:', node)
```

```
        lhs = yield node.left
        print('left=', lhs)
        rhs = yield node.right
        print('right=', rhs)
        yield lhs + rhs
    ...
```

äÿŃéİćæŸŕçőÄå■ŢçŽĎæţŃèŕŢïijŽ

```
>>> e = Evaluator()
>>> e.visit(t4)
Add: <__main__.Add object at 0x1006a8d90>
left= 1
right= -0.4
0.6
>>>
```

èőĺèőž

è£ŹäÿĂåřŔèŁĆæĹŚäžňæijŤçd'žäžEçŤ§æĹŘåŹĺåŠŇå■ŔçĺŃåIJĺçĺŃåžŔæŐğåĹűæţĄæŰźéİćçŽĎåijžåd'ğ
éĄ£åĚ■éĂŠå¡ŠçŽĎäÿĂäÿłéĂŽåÿÿæŰźæşŢæŸŕä¡£çŤĺäÿĂäÿłæăĹæĹŰéŸ§åĹŮçŽĎæŢřæ■őçżŞæđĎãĂĆ
ä¿ŃåęĆïijŇæůšåžęäijŸåĚĹçŽĎéĄ■åŐEçőŮæşŢïijŇçňňäÿĂæňąçćřåĹřäÿĂäÿłèŁĆçĆźæŮűåřĘåĚűåŐŃåĚěæă
æŰźæşŢçŽĎæäÿå£ČæĂİèůŕåřśæŸŕè£Źæăůã ĂĆ

åŘęåd'ŰäÿĂäÿłéIJĂèęĄçŘĘèğčçŽĎåřśæŸŕçŤ§æĹŘåŹĺäÿ■yieldèŕ■åŔěãĂĆå¡Şçćřå Ĺřyieldèŕ■åŔěæŮűïij
äÿŁéİćçŽĎä¿Ńå■Řä¡£çŤĺè£ŹäÿłæŁĂæIJŕæİěäžčæŽ£äžĘéĂŠå¡ŠãĂĆä¿ŃåęĆïijŇäźŃåL'■æĹŚäžňæŸŕè£Źæăů

```
value = self.visit(node.left)
```

çŐřåIJĺæ■ćæĹŘyieldèŕ■åŔěijŽ

```
value = yield node.left
```

åőČäijŽåřĘ `node.left` è£ŤåŻđçżŹ `visit()` æŰźæşŢïijŇçĎűåŘŐ `visit()` æŰźæşŢěŕČçŤĺéĆčäÿłèŁĆçĆźçŻÿåžŤçŽĎ `visit_Name()` æŰźæşŢãĂĆ yield-æŽĆæŮűåřEçĺŃåžŔæŐğåĹűåŹĺèől'åĞžçżŹèřČçŤĺèĂĚïijŇå¡ŞæL'ğèąŇåőŇåŘŐïijŇçżŞæđIJäijŽèţŃåĂijçżŹv

çIJŃåőŇè£ŹäÿĂåřŔèŁĆïijŇä¡ăäź§èőÿæČşåŐżåŕżæL'¿åĚűåőČæšąæIJL'yieldèŕ■åŔěçŽĎæŰźæąĹãĂĆä¡E
ä¿ŃåęĆïijŇäÿžäžEąéűĹéŹd'éĂŠå¡ŠïijŇä¡ăå£ĚéążèęĄçżt'æŁd'äÿĂäÿłæăĹçżŞæđĎïijŇåęĆæđIJäÿ■ä¡£çŤĺçŤ§a
åőđéŹĚäÿŁïijŇä¡£çŤĺyieldèŕ■åŔěåŘŕäžěèől'ä¡ăåĘŹåĞžéİđåÿÿæijĆäžőçŽĎäžčçăĄïijŇåőČæűĹéŹd'äžĘéĂŠå¡

8.23 å¿łçŐŕåijŢçŤĺæŢřæ■őçżŞæđĎçŽĎåĘĚå■ŸçőąçŘĘ

éŮőécŸ

ä¡ăçŽĎçĺŃåžŔåĹŻåžžäžEå¿Ĺåd'Žå¿łçŐŕåijŢçŤĺæŢřæ■őçżŞæđĎ(æŕŤåęĆæăŚãĂĄåŻ¿ãĂĄèğĆår§èĂĚæ

èğčåĘşæŰźæąĹ

äÿĂäÿłçőÅå■ŢçŽĎå¿łçŐŕåijŢçŤĺæŢřæ■őçżŞæđĎä¿Ńå■Řåřśæ˙ŕäÿĂäÿłæăŚå¡ćçżŞæđĎïijŇåŔŇäžšèŁĆ
è£Źçğ■æČĚåĘţäÿŃïijŇåŔŕäżěèĂČèŹŚä¡£çŤĺ `weakref` åžŞäÿ■çŽĎåijśåijŢçŤĺãĂĆä¿ŃåęĆïijŽ

```
import weakref

class Node:
    def __init__(self, value):
        self.value = value
        self._parent = None
        self.children = []

    def __repr__(self):
        return 'Node({!r:})'.format(self.value)

    # property that manages the parent as a weak-reference
    @property
    def parent(self):
        return None if self._parent is None else self._parent()

    @parent.setter
    def parent(self, node):
        self._parent = weakref.ref(node)

    def add_child(self, child):
        self.children.append(child)
        child.parent = self
```

è£Źçğ■æŸŕæČşæŰźåijŔåĚĄèőÿparentéİŹéżŸçżĹæ■ćãĂĆä¿ŃåęĆïijŽ

```
>>> root = Node('parent')
>>> c1 = Node('child')
>>> root.add_child(c1)
>>> print(c1.parent)
Node('parent')
>>> del root
>>> print(c1.parent)
None
>>>
```

èőĺèőž

å¿łçŐŕåijŢçŤĺçŽĎæŢřæ■őçżŞæđĎåIJíPythonäÿ■æŸŕäÿĂäÿłå¿ĹæčŸæĽ'ŃçŽĎéŮőécŸïijŇåŻääÿžæ■čåÿ
ä¿ŃåęĆèĂČèŹŚåęĆäÿŃäžčçăĄïijŽ

```
# Class just to illustrate when deletion occurs
class Data:
    def __del__(self):
```

```
        print('Data.__del__')

# Node class involving a cycle
class Node:
    def __init__(self):
        self.data = Data()
        self.parent = None
        self.children = []

    def add_child(self, child):
        self.children.append(child)
        child.parent = self
```

äÿŃéÌćæĹŚäżñä¡£çŤÍè£ŹäÿłäžčçăĄæİěåĄŽäÿĂäžŽåđČåIJ¿åŽđæŤűèŕŢéłŃïijŽ

```
>>> a = Data()
>>> del a # Immediately deleted
Data.__del__
>>> a = Node()
>>> del a # Immediately deleted
Data.__del__
>>> a = Node()
>>> a.add_child(Node())
>>> del a # Not deleted (no message)
>>>
```

åŔŕäżěçIJŃåĹřïijŇæIJĂåŘŐäÿĂäÿłçŽĎåĹăéŹď'æŮűæL'Şå■řèŕ■åŘěæšąæIJL'åĞžçŐřãĂĆåŐ§åŽăæŸŕPy
å¡ŞäÿĂäÿłåŕźèšąçŽĎåijŢçŤÍæŢřåŔŸæĹŘ0çŽĎæŮűåĂŹæL'■äijŽçńŃå■şåĹăéŹď'æŐL'ãĂĆèĂŇåŕźäžŐå¿łçŐŕ
åŽăæ■d'ïijŇåIJÍäÿŁéÌćä¿Ńå■Řäÿ■æIJĂåŘŐéČÍåĹĘïijŇçĹűèŁĆçĆźåŠŇå■l'å■ŘèŁĆçĆźäžŠçŽÿæŃěæIJL'åŕźa

PythonæIJL'åŔęåd'ŰçŽĎåđČåIJ¿åŽđæŤűåŹÍæİěäÿŞéŮÍéŠĹåŕźå¿łçŐŕåijŢçŤÍçŽĎïijŇä¡ĘæŸŕä¡ăæřÿè£IJ
åŔęåd'Űä¡ăè£ŸåŔŕäžěæL'ŃåŁÍçŽĎèğęåŔŠåőČïijŇä¡ĘæŸŕäžčçăĄçIJŃäÿŁåŐżå¿ĹæŇńïijŽ

```
>>> import gc
>>> gc.collect() # Force collection
Data.__del__
Data.__del__
>>>
```

åęĆæđIJå¿łçŐŕåijŢçŤÍçŽĎåŕźèšąèĞłåůśè£ŸåőŽäźL'äžĘèĞłåůśçŽĎ
__del__() æŰźæşŢïijŇéĆčäźĹäijŽèől'æČĚåĘţåŔŸå¿ŮæŻt'çş§çşŢãĂĆ
åĄĞèő¿ä¡ăåČŔäÿŃéÌćè£ŹæăůçżŹNodeåőŽäźL'èĞłåůśçŽĎ __del__() æŰźæşŢïijŽ

```
# Node class involving a cycle
class Node:
    def __init__(self):
        self.data = Data()
        self.parent = None
        self.children = []

    def add_child(self, child):
```

```
        self.children.append(child)
        child.parent = self

    # NEVER DEFINE LIKE THIS.
    # Only here to illustrate pathological behavior
    def __del__(self):
        del self.data
        del.parent
        del.children
```

è£Źçğ■æČĚåĘţäÿŃïijŇåđČåIJ¿åŻđæŤűæřÿè£IJéČ¡äÿ■äijŽåŐżåŻđæŤűè£ŹäÿłářźèśąçŽĎïijŇè£ŸäijŽářij
åęĆæđIJä¡ăèŕŢçİĂåŐżè£ŘèąŇåőČäijŽåŔŚçŐřïijŇData.__del__
æűĹæĄŕæřÿè£IJäÿ■äijŽåĞžçŐřäžE,çŤŽèĞşåIJĺä¡ååijžåĹűåĘĚå■ŸåŻđæŤűæŮűïijŽ

```
>>> a = Node()
>>> a.add_child(Node())
>>> del a # No message (not collected)
>>> import gc
>>> gc.collect() # No message (not collected)
>>>
```

åijśåijŢçŤĺæűĹéŹđ’äžĘåijŢçŤĺå¿łçŐŕçŽĎè£ŹäÿłéŮőécŸïijŇæIJňèt’ĺæİèèőšïijŇåijśåijŢçŤĺářśæŸŕäÿĂäÿ
ä¡ååŔŕäžěéĂŽè£Ğ weakref æİěåĹŽåżžåijśåijŢçŤĺãĆCä¿ŃåęĆïijŽ

```
>>> import weakref
>>> a = Node()
>>> a_ref = weakref.ref(a)
>>> a_ref
<weakref at 0x100581f70; to 'Node' at 0x1005c5410>
>>>
```

äÿžäžĘèő£éŮőåijśåijŢçŤĺæL’ĂåijŢçŤĺçŽĎåŕźèśąïijŇä¡ååŔŕäžěåČŔåĞ¡æŢřäÿĂæăůåŐżèřČçŤĺåőČå■şåŔ
çŤśäžŐåŐ§ågŃåŕźèśąçŽĎåijŢçŤĺèőąæŢřæšąæIJL’åćđåŁäïijŇéĆčäźĹåŕśåŔŕäžěåŐżåĹäéŹđ’åőČäžĘãĂĆä¿Ńåę

```
>>> print(a_ref())
<__main__.Node object at 0x1005c5410>
>>> del a
Data.__del__
>>> print(a_ref())
None
>>>
```

éĂŽè£Ğè£ŹéĞŇæijŤçđ’žçŽĎåijśåijŢçŤĺæŁĂæIJŕïijŇä¡ăäijŽåŔŚçŐřäÿ■åĘ■æIJL’å¿łçŐŕåijŢçŤĺéŮőécŸ
ä¡ăè£ŸèČ¡åŔĆèĂČ8.25åřŔèŁĆåĚşäžŐåijśåijŢçŤĺçŽĎåŔęåđ’ŰäÿĂäÿłä¿Ńå■ŘãĂĆ

8.24 èőľçśżæŤŕæŇĄæŕŤè¿ČæŞ■ä¡IJ

éŮőécŸ

ä¡ăæČşèőľæ§Řäÿłçśžçž Ďåőđä¿ŃæŤŕæŇĄæăĞåĞĘçŽĎæŕŤè¿Čè£ŘçőŮ(æŕŤåęĆ>=,!=,<=,<ç■L')ïijŇä¡Ę

èğčåĘşæŰźæąĹ

PythonçśžåŕźæŕŕäÿłæŕŤè¿ČæŞ■ä¡IJéČ¡éIJĂèęĄåőđçŐřäÿĂäÿłçL'źæőŁæŰźæşŢæİěæŤŕæŇĄãĂĆ ä¿ŃåęĆäÿžäžĘæŤŕæŇĄ>=æŞ■ä¡IJçňęïijŇä¡ăéIJĂèęĄåőŽäźL'äÿĂäÿł `__ge__()` æŰźæşŢãĂĆ åř¡çőąåőŽäźL'äÿĂäÿłæŰźæşŢæšąäžĂäźĹéŮőécŸïijŇä¡ĘåęĆæđIJèęĄä¡ăåőđçŐřæL'ĂæIJL'åŕŕè

èčĚéěřåŹÍ `functools.total_ordering` åřśæŸŕçŤÍæİěçőĂåŇŰè£Źäÿłåd'ĎçŘĘçŽĎãĂĆ ä¡£çŤÍåőČæİěèčĚéěřäÿĂäÿłæİěïijŇä¡ăåŔłéIJĂåőŽäźL'äÿĂäÿł `__eq__()` æŰźæşŢïijŇ åd'ŰåŁăåĚűäžŰæŰźæşŢ(`__lt__`, `__le__`, `__gt__`, or `__ge__`)äÿ■çŽĎäÿĂäÿłå■şåŔřãĂĆ çĎűåŘŐèčĚéěřåŹĺäijŽèĞłåŁĺäÿžä¡ăåąńåĚĚåĚűåőČæŕŤè¿ČæŰźæşŢãĂĆ

ä¡IJäÿžä¿Ńå■ŘïijŇæĹŚäžňædĎåžžäÿĂäžŻæĹ£å■ŘïijŇçĎűåŘŐçżŹåőČäžňáćđåŁääÿĂäžŻæĹ£éŮt'ïijŇæ

```
from functools import total_ordering

class Room:
    def __init__(self, name, length, width):
        self.name = name
        self.length = length
        self.width = width
        self.square_feet = self.length * self.width

@total_ordering
class House:
    def __init__(self, name, style):
        self.name = name
        self.style = style
        self.rooms = list()

    @property
    def living_space_footage(self):
        return sum(r.square_feet for r in self.rooms)

    def add_room(self, room):
        self.rooms.append(room)

    def __str__(self):
        return '{}: {} square foot {}'.format(self.name,
                self.living_space_footage,
                self.style)

    def __eq__(self, other):
        return self.living_space_footage == other.living_space_
↪footage

    def __lt__(self, other):
```

```
        return self.living_space_footage < other.living_space_
↪footage
```

è£ŹéĞŇæĹŚäžňåŔłæŸŕçżŹHouseçśżåőŽäźL'äžĘäÿd'äÿłæŰźæşŢïijŽ__eq__() åŠŇ __lt__() ïijŇåőČåřśèČ¡æŤŕæŇĄæL'ĂæIJL'çŽĎæŕŤè¿ČæŞ■ä¡IJïijŽ

```
# Build a few houses, and add rooms to them
h1 = House('h1', 'Cape')
h1.add_room(Room('Master Bedroom', 14, 21))
h1.add_room(Room('Living Room', 18, 20))
h1.add_room(Room('Kitchen', 12, 16))
h1.add_room(Room('Office', 12, 12))
h2 = House('h2', 'Ranch')
h2.add_room(Room('Master Bedroom', 14, 21))
h2.add_room(Room('Living Room', 18, 20))
h2.add_room(Room('Kitchen', 12, 16))
h3 = House('h3', 'Split')
h3.add_room(Room('Master Bedroom', 14, 21))
h3.add_room(Room('Living Room', 18, 20))
h3.add_room(Room('Office', 12, 16))
h3.add_room(Room('Kitchen', 15, 17))
houses = [h1, h2, h3]
print('Is h1 bigger than h2?', h1 > h2) # prints True
print('Is h2 smaller than h3?', h2 < h3) # prints True
print('Is h2 greater than or equal to h1?', h2 >= h1) # Prints False
print('Which one is biggest?', max(houses)) # Prints 'h3: 1101-
↪square-foot Split'
print('Which is smallest?', min(houses)) # Prints 'h2: 846-square-
↪foot Ranch'
```

èőÍèőž

åĚűåőđ total_ordering èčĚéěřåŹÍäź§æšąéĆčäźĹçěđçğŸãĂĆ åőČåřśæŸŕåőŽäźL'äžĘäÿĂäÿłäžŐæŕŔäÿłæŕŤè¿ČæŤŕæŇĄæŰźæşŢåĹřæL'ĂæIJL'éIJĂèęĄåőŽäźL'çŽĎåĚűäżÚ æŕŤåęĆä¡ăåőŽäźL'äžĘ __le__() æŰźæşŢïijŇéĆčäźĹåőČåřśèćńçŤÍæİěædĎåžžæL'ĂæIJL'åĚűäżŰçŽĎéIJĂè åőđéŹĚäÿŁåřśæŸŕåIJÍçśżéĞŇéİćåČŔäÿŃéİćè£ŹæăůåőŽäźL'äžĘäÿĂäžŽçL'źæőŁæŰźæşŢïijŽ

```
class House:
    def __eq__(self, other):
        pass
    def __lt__(self, other):
        pass
    # Methods created by @total_ordering
    __le__ = lambda self, other: self < other or self == other
    __gt__ = lambda self, other: not (self < other or self == other)
    __ge__ = lambda self, other: not (self < other)
    __ne__ = lambda self, other: not self == other
```

å¡ŞçĎűïijŇä¡äèĞłåůšåŐžåĘŹäź§å¿Ĺåőźæ ŸŞïijŇä¡ĘæŸŕä¡£çŤÍ @total_ordering

åŔŕäžěçőĂåŇŰäžčçăĄïijŇä¡ŢäźŘèĂŇäÿ■äÿžåŚćãĂĆ

8.25 åĹŻåžžçijŞå■Ÿåőđä¿Ń

éŮőécŸ

åIJĺåĹŻåžžäÿĂäÿłçśżçŽĎåŕźèśąæŮűïijŇåęĆæđIJäźŃåL’■ä¡£çŤĺåŘŇæăůåŔĆæŢřåĹŻåžžè£Ğè£Źäÿłåŕźèś
ä¡ăæČçè£ŤåŻđåőČçŽĎçijŞå■ŸåijŢçŤĺãĂĆ

èğčåĘşæŰźæąĹ

è£Źçğ■éĂŽåÿÿæŸŕåŻăäÿžä¡ăåÿŇæIJŻçŻÿåŘŇåŔĆæŢřåĹŻåžžçŽĎåŕźèśąæŮűå■Ţä¿ŃçŽĎãĂĆ
åIJĺå¿Ĺåď’ŽåžŞäÿ■éČ¡æIJL’åőđéŹĚçŽĎä¿Ńå■ŘïijŇæŕŤåęĆ `logging`
æĺąåİŮïijŇä¡£çŤĺçŻÿåŘŇçŽĎåŘ■çğřåĹŻåžžçŽĎ `logger` åőđä¿Ńæřÿè£IJåŘłæIJL’äÿĂäÿłãĂĆä¿ŃåęĆïijŽ

```
>>> import logging
>>> a = logging.getLogger('foo')
>>> b = logging.getLogger('bar')
>>> a is b
False
>>> c = logging.getLogger('foo')
>>> a is c
True
>>>
```

äÿžäžĘè¿¿åĹřè£ŹæăůçŽĎæŢĹæđIJïijŇä¡ăéIJĂèęĄä¡£çŤĺäÿĂäÿłåŠŇçśżæIJňèžńåĹĘåijĂçŽĎåůěåŐĆåĞ

```
# The class in question
class Spam:
    def __init__(self, name):
        self.name = name

# Caching support
import weakref
_spam_cache = weakref.WeakValueDictionary()
def get_spam(name):
    if name not in _spam_cache:
        s = Spam(name)
        _spam_cache[name] = s
    else:
        s = _spam_cache[name]
    return s
```

çĎűåŘŐåĄŽäÿĂäÿłæţŃèŕŢïijŇä¡ăäijŽåŔŞçŐřèů§äźŃåL’■éĆčäÿłæŮěå£ŮåŕźèśąçŽĎåĹŻåžžèąŇäÿžæŸŕ

```
>>> a = get_spam('foo')
>>> b = get_spam('bar')
>>> a is b
```

```
False
>>> c = get_spam('foo')
>>> a is c
True
>>>
```

èőÍèőž

çijŰåĘŹäÿĂäÿłåůěåŐĆåĞ¡æŢřæİěä£őæŤźæŹőéĂŽçŽĎåőđä¿ŃåĹŻåžžèąŇäÿžéĂŽåÿÿæŸŕäÿĂäÿłæŕŤè¿ä¡ĘæŸŕæĹŚäžňè£ŸèČ¡åŘęæĽ’¿åĹřæŻt’äijŸéŻĚçŽĎè§čåĘşæŰźæąĹåŚćïij§

ä¿ŃåęĆïijŇä¡ăåŔŕèČ¡äijŽèĂČèŹŚéĞ■æŰřåőŽäźĽ’çśżçŽĎ `__new__()` æŰźæşŢïijŇářšåČŔäÿŃéİćè£ŹæăůïijŽ

```
# Note: This code doesn't quite work
import weakref

class Spam:
    _spam_cache = weakref.WeakValueDictionary()
    def __new__(cls, name):
        if name in cls._spam_cache:
            return cls._spam_cache[name]
        else:
            self = super().__new__(cls)
            cls._spam_cache[name] = self
            return self
    def __init__(self, name):
        print('Initializing Spam')
        self.name = name
```

åĹİçIJŃètůæİěåě¡åČŔåŔŕäžěè¿¿åĹřécĎæIJ§æŢĹæđIJïijŇä¡ĘæŸŕéŮőécŸæŸŕ `__init__()` æŕŔæňąéČ¡äijŽècńèřČçŤĺïijŇäÿ■çőąè£Źäÿłåőđä¿ŃæŸŕåŘęècńçijŞå■ŸäžĘãĂĆä¿ŃåęĆïijŽ

```
>>> s = Spam('Dave')
Initializing Spam
>>> t = Spam('Dave')
Initializing Spam
>>> s is t
True
>>>
```

è£ŹäÿłæĹŰèőÿäÿ■æŸŕä¡ăæČşèęĄçŽĎæŢĹæđIJïijŇåŻăæ■d’è£Źçğ■æŰźæşŢåźűäÿ■åŔŕåŔŰãĂĆ

äÿŁéİćæĹŚäžňä¡£çŤĺåĹřäžĘåijśåijŢçŤĺèőąæŢřïijŇårźäžŐåđČåIJ¿åŻđæŤűæİèèőšæŸŕå¿ĹæIJL’åÿőåŁľçŽå¡ŞæĹŚäžňä£ÍæŇĄåőđä¿ŃçijŞå■ŸæŮűïijŇä¡ăåŔŕèČ¡åŘłæČşåIJĺçĺŃåžŘäÿ■ä¡£çŤĺåĹřåőČäžňæŮűæL’■ä£Íå■äÿĂäÿł `WeakValueDictionary` åőđä¿ŃåŘłäijŽä£Íå■ŸéĆčäžŻåIJĺåĚűåőČåIJřæŰźè£ŸåIJĺèćńä¡£çŤĺçŽĎååŘęåĹŹçŽĎèŕİïijŇåŘłèęĄåőđä¿ŃäÿåĘ■èćńä¡£çŤĺäžĘïijŇåőČářšäžŐå■ŮåĚÿäÿ■èćńçğżéŹd’äžĘãĂĆèğĆår§ä

```
>>> a = get_spam('foo')
>>> b = get_spam('bar')
>>> c = get_spam('foo')
>>> list(_spam_cache)
['foo', 'bar']
>>> del a
>>> del c
>>> list(_spam_cache)
['bar']
>>> del b
>>> list(_spam_cache)
[]
>>>
```

åŕźäžŐåď ğéČĺåĹEçÍŃåžŔèĂŇåůšïijŇè£ŹéĞŇäżčçăĄåůšçżŔåď §çŤĺäžĘãĂĆäÿ■è£Ğè£ŸæŸŕæIJL'äÿĂäž

éęŰåĚĹæŸŕè£ŹéĞŇä¡£çŤĺåĹřäžĘäÿĂäÿłåĚĺåśĂåŔŸéĞŔïijŇåźűäÿŤåůěåŐĆåĞ¡æŢřèů§çśżæŤ¿åIJĺäÿĂ

```
import weakref

class CachedSpamManager:
    def __init__(self):
        self._cache = weakref.WeakValueDictionary()

    def get_spam(self, name):
        if name not in self._cache:
            s = Spam(name)
            self._cache[name] = s
        else:
            s = self._cache[name]
        return s

    def clear(self):
            self._cache.clear()

class Spam:
    manager = CachedSpamManager()
    def __init__(self, name):
        self.name = name

    def get_spam(name):
        return Spam.manager.get_spam(name)
```

è£ŹæăůçŽĎèŕÍäžčçăĄæŻť æÿĚæŹřïijŇåźűäÿŤäź§æŻť çĄţæť żïijŇæĹŚäžňåŔŕäžěåćđåŁăæŻť åď ŽçŽĎçij

è£ŸæIJL'äÿĂçĆźåřśæŸŕïijŇæĹŚäžňæŽť éIJšäžĘçśżçŽĎåőđä¿ŃåŇŰçżŹçŤĺæĹůåĳijŇçŤĺæĹůå¿Ĺåőźæ ŸŞ

```
>>> a = Spam('foo')
>>> b = Spam('foo')
>>> a is b
False
```

```
>>>
```

æIJL’åĞăçğ■æŰźåijŔåŔŕäžéÝšæ■ćçŤĺæĹůè£Źæăůå‍ĄŽïijŇçňñäÿĂäÿłæŸŕářEçśžçŽĎåŘ■å■Ůä£őæŤźäÿçňñäžŇçğ■ářśæŸŕèől’è£ŹäÿłçśžçŽĎ `__init__()` æŰźæşŢæŁŻåĞžäÿĂäÿłåijĆåÿÿïijŇèől’åőČäÿ■èČ¡èćńåĹ

```
class Spam:
    def __init__(self, *args, **kwargs):
        raise RuntimeError("Can't instantiate directly")

    # Alternate constructor
    @classmethod
    def _new(cls, name):
        self = cls.__new__(cls)
        self.name = name
```

çĎűåŘŐä£őæŤźçijŞå■Ÿçőąç‍ŘĘåŹĺäžčçăĄïijŇä¡£çŤĺ `Spam._new()` æİěåĹŻåžžåőđä¿ŃïijŇèĂŇäÿ■æŸŕçŻt’æŐěěřČçŤĺ `Spam()` æđĎéĂääĞ¡æŢřïijŽ

```
# ----------------------æIJĂåŘŐçŽĎä£őæ■čæŰźæąĹ--------------------
↪---
class CachedSpamManager2:
    def __init__(self):
        self._cache = weakref.WeakValueDictionary()

    def get_spam(self, name):
        if name not in self._cache:
            temp = Spam3._new(name)  # Modified creation
            self._cache[name] = temp
        else:
            temp = self._cache[name]
        return temp

    def clear(self):
            self._cache.clear()

class Spam3:
    def __init__(self, *args, **kwargs):
        raise RuntimeError("Can't instantiate directly")

    # Alternate constructor
    @classmethod
    def _new(cls, name):
        self = cls.__new__(cls)
        self.name = name
        return self
```

æIJĂåŘŐè£Źæăůç‍ŽĎæŰźæąĹářšåůšçżŔèűşåď’§åě¡äžĘãĂĆçijŞå■ŸåŠŇåĔűäżŰæđĎéĂäæĺąåijŔè£ŸåŔŕäžěä¡£çŤĺ9.13ářŔèŁĆäÿ■çŽĎåĔČçśżåőđçŐřçŽĎæŻt’äijŸéŻĔäÿ

çňňäźÌçńäïijŽåĚČçijŰçÍŃ

è¡ŕäžűåijĂåŔŠéćĘå§§äÿ■æIJĂçżŔåĚÿçŽĎåŔčåď't'çęĚåřśæŸŕâĂIJdonâĂŹt repeat yourselfâĂİãĂĆ äźšåřśæŸŕèŕt'ïijŇäżżä¡ŢæŮűåĂŹå¡Şä¡ăçŽĎçÍŃåžŔäÿ■å■ŸåIJÍéńŸåžęéĞ■åď■(æĹŰèĂĚæŸŕéĂ
åIJÍPythonå¡Şäÿ■ïijŇéĂŽåÿÿéČ¡åŔŕäżěéĂŽè£ĞåĚČçijŰçÍŃæİěèğčåĘşè£ŹçśžéŮőécŸãĂĆ
çőĂèĂŇèÍĂäźŃïijŇåĚČçijŰçÍŃåřśæŸŕåĘşäžŐåĹŻåżżæŞ■ä¡IJæžŘäżčçăĄ(æŕŤåęĆä£őæŤźãĂĄçŢ§æĹŘæĹŰ
äÿżèęĄæŁĂæIJŕæŸŕä¡£çŤÍèčĚéěřåŹÍãĂĄçśżèčĚéěřåŹÍåŠŇåĚČçśžãĂĆäÿ■è£Ğè£ŸæIJL'äÿĂäžŻåĚűäžŰæŁĂ
åŇĚæŃňç■¿åŘ■åŕźèśąãĂĄä¡£çŤÍ `exec()` æL'ğèąŇäżčçăĄäżěåŔŁåŕźåĘĚéČÍåĞ¡æŢřåŠŇçśżçŽĎåŔ■åřĎæŁ
æIJňçńäçŽĎäÿżèęĄçŽőçŽĎæŸŕåŘŠåď'ğåőűäżŃçż■è£ŹäžŻåĚČçijŰçÍŃæŁĂæIJŕïijŇåźűäÿŤçżŹåĞžåőđä¿Ńæ

Contents:

9.1 åIJÍåĞ¡æŢřäÿŁæůżåŁäåŇĚèčĚåŹÍ

éŮőécŸ

ä¡ăæČşåIJÍåĞ¡æŢřäÿŁæůżåŁääÿĂäÿłåŇĚèčĚåŹÍïijŇåćđåŁäécİåď'ŰçŽĎæŞ■ä¡IJåď'ĎçŘĘ(æŕŤåęĆæŮěå

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşä¡£çŤÍécİåď'ŰçŽĎäżčçăĄåŇĚèčĚäÿĂäÿłåĞ¡æŢřïijŇåŔŕäżěåőŽäźL'äÿĂäÿłèčĚéěřåŹÍåĞ

```
import time
from functools import wraps

def timethis(func):
    '''
    Decorator that reports the execution time.
    '''
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.time()
        result = func(*args, **kwargs)
        end = time.time()
        print(func.__name__, end-start)
        return result
    return wrapper
```

äÿŃéİćæŸŕä¡£çŤÍèčĚéěřåŹÍçŽĎä¿Ńå■ŘïijŽ

```
>>> @timethis
... def countdown(n):
...     '''
...     Counts down
...     '''
...     while n > 0:
...         n -= 1
```

```
...
>>> countdown(100000)
countdown 0.008917808532714844
>>> countdown(10000000)
countdown 0.87188299392912
>>>
```

èőÍèőž

äÿĂäÿłèčĚéěřåŹÍåřśæŸŕäÿĂäÿłåĞ¡æŢřïijŇåőČæŐěåŔŮäÿĂäÿłåĞ¡æŢřä¡IJäÿžåŔĆæŢřåźűè£ŤåŻđäÿĂäÿ
å¡Şä¡ăåČŔäÿŃéÍćè£ŹæăůåĘŹïijŽ

```
@timethis
def countdown(n):
    pass
```

èů§åČŔäÿŃéÍćè£ŹæăůåĘŹåĚűåőđæŢĹæđIJæŸŕäÿĂæăůçŽĎïijŽ

```
def countdown(n):
    pass
countdown = timethis(countdown)
```

éąžä¿£èŕť äÿĂäÿŃïijŇåĘĚç¡őçŽĎèčĚéěřåŹÍæŕŤåęĆ `@staticmethod`, `@classmethod`,`@property` åŐ§çŘĘäź§æŸŕäÿĂæăůçŽĎãĂĆ
ä¿ŃåęĆïijŇäÿŃéÍćè£ŹäÿďäÿłäžčçăĄçL’ĞæőţæŸŕç■L’äžůçŽĎïijŽ

```
class A:
    @classmethod
    def method(cls):
        pass

class B:
    # Equivalent definition of a class method
    def method(cls):
        pass
    method = classmethod(method)
```

åIJÍäÿŁéÍćçŽĎ `wrapper()` åĞ¡æŢřäÿ■ïijŇ èčĚéěřåŹÍåĘĚéČÍåőŽäźL’äžĘäÿĂäÿłä¡£çŤÍ `*args` åŠŇ `**kwargs` æÍěæŐěåŘŮäżżæĎŔåŔĆæŢřçŽĎåĞ¡æŢřãĂĆ
åIJÍè£ŹäÿłåĞ¡æŢřéĞŇéÍćèřČçŤÍäžĘåŐ§ågŇåĞ¡æŢřåźűårĘåĚűçżŞæđIJè£ŤåŻđïijŇäÿ■è£Ğä¡ăè£ŸåŔŕäžěæůż
çĎűåŘŐè£ŹäÿłæŰřçŽĎåĞ¡æŢřåŇĚèčĚåŹÍèćńä¡IJäÿžçżŞæđIJè£ŤåŻđæÍěäżčæŻ£åŐ§ågŇåĞ¡æŢřãĂĆ

éIJĂèęĄåijžèřČçŽĎæŸŕèčĚéěřåŹÍåźűäÿ■äijŽä£őæŤźåŐ§ågŇåĞ¡æŢřçŽĎåŔĆæŢřç■¿åŘ■äžěåŔŁè£ŤåŻ
ä¡£çŤÍ `*args` åŠŇ `**kwargs` çŻőçŽĎåřśæŸŕçąőä£Íäżżä¡ŢåŔĆæŢřéČ¡èČ¡éĂĆçŤÍãĂĆ
èĂŇè£ŤåŻđçżŞæđIJåĂijå§žæIJňéČ¡æŸŕèřČçŤÍåŐ§ågŇåĞ¡æŢř `func(*args, **kwargs)` çŽĎè£ŤåŻđçżŞæđIJïijŇåĚűäÿ■funcåřśæŸŕåŐ§ågŇåĞ¡æŢřãĂĆ

åĹŽåijĂågŇå■ęäźăèčĚéěřåŹÍçŽĎæŮűåĂŹïijŇäijŽä¡£çŤÍäÿĂäžŻçőÅå■ŢçŽĎä¿Ńå■ŘæÍěèŕť æŸŐïijŇæŕ
äÿ■è£ĞåđéŹĚåIJžæŹŕä¡£çŤÍæŮűïijŇè£ŸæŸŕæIJL’äÿĂäžŻçżĘèŁĆéŮőécŸèęĄæşÍæĎŔçŽĎãĂĆ

æґŤåęĆäÿŁéİćä¡£çŤÍ `@wraps(func)` æşÍèğčæŸŕå¿ĹéĞ■èęĄçŽĎïijŇ
åőČèČ¡ä£İçŢŹåŐ§åğŃåĞ¡æŢřçŽĎåĚČæŢřæ■ő(äÿŃäÿĂåřŔèŁĆäijŽèőšåĹř)ïijŇæŰřæĽŃçżŔåÿÿäijŽå£¡çŢèè
æŐěäÿŃæİěçŽĎåĞääÿłåřŔèŁĆæĹŚäžñäijŽæŻtʼåŁäæůśåĚěçŽĎèőšèğčèčĚéěřåŹĺåĞ¡æŢřçŽĎçżĘèŁĆéŮőécŸ

9.2 åĹŻåżžèčĚéěřåŹĺæŮűä£İçŢŹåĞ¡æŢřåĚČä£ąæĄŕ

éŮőécŸ

ä¡ääĘŹäžĘäÿĂäÿłèčĚéěřåŹĺä¡IJçŤĺåIJĺæ§ŘäÿłåĞ¡æŢřäÿŁïijŇä¡ĘæŸŕè£ŹäÿłåĞ¡æŢřçŽĎéĞ■èęĄçŽĎåĚ

èğčåĘşæŰźæąĹ

äżżä¡ŢæŮűåĂŹä¡ääőŽäźL'èčĚéěřåŹĺçŽĎæŮűåĂŹïijŇéČ¡åžŤèŕěä¡£çŤÍ `functools` åžŞäÿ■çŽĎ `@wraps` èčĚéěřåŹĺæİěæşÍèğčåžŢåśĆåŇĚèčĚåĞ¡æŢřãĂĆä¿ŇåęĆïijŽ

```
import time
from functools import wraps
def timethis(func):
    '''
    Decorator that reports the execution time.
    '''
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.time()
        result = func(*args, **kwargs)
        end = time.time()
        print(func.__name__, end-start)
        return result
    return wrapper
```

äÿŃéİćæĹŚäžñä¡£çŤÍè£ŹäÿłèćńåŇĚèčĚåŘŐçŽĎåĞ¡æŢřåźűæčĂæ§ěåőČçŽĎåĚČä£ąæĄŕïijŽ

```
>>> @timethis
... def countdown(n):
...     '''
...     Counts down
...     '''
...     while n > 0:
...         n -= 1
...
>>> countdown(100000)
countdown 0.008917808532714844
>>> countdown.__name__
'countdown'
>>> countdown.__doc__
'\n\tCounts down\n\t'
>>> countdown.__annotations__
```

```
{'n': <class 'int'>}
>>>
```

èőÍèőž

åIJÍçijŰåĘŹèčĚéěřåŹÍçŽĎæŮűåĂŹåď■åĹűåĚČä£ąæĄŕæŸŕäÿĂäÿłéİđåÿÿéĞ■èęĄçŽĎéČÍåĹĘãĂĆåęĆæ
@wraps ïijŇ éĆčäźĹä¡ääijŽåŔŞçŐřèćńèčĚéěřåĞ¡æŢřäÿćåď śäžĘæĽ ĂæIJĽæIJĽçŤÍçŽĎä£ąæĄŕãĂĆæŕŤåęĆ
@wraps åŘŐçŽĎæŢĹædIJæŸŕäÿŃéİćè£ŹæăůçŽĎïijŽ

```
>>> countdown.__name__
'wrapper'
>>> countdown.__doc__
>>> countdown.__annotations__
{}
>>>
```

@wraps æIJĽäÿĂäÿłéĞ■èęĄçĽźå¿ĄæŸŕåőČèČ¡èőľä¡äéĂŽè£ĞåśđæĂğ
__wrapped__ çŻť æŐèèő£éŮõèćńåŇĚèčĚåĞ¡æŢřãĂĆä¿ŃåęĆ:

```
>>> countdown.__wrapped__(100000)
>>>
```

__wrapped__ åśđæĂğè£ŸèČ¡èőľèćńèčĚéěřåĞ¡æŢřæ■čçąőæŽť éIJšåžŢåśĆçŽĎåŔĆæŢřç■¿åŘ■ä£ąæA

```
>>> from inspect import signature
>>> print(signature(countdown))
(n:int)
>>>
```

äÿĂäÿłå¿ĹæŹőéĄ■çŽĎéŮőécŸæŸŕæĂŐæăůèőľèčĚéěřåŹÍåŐżçŻť æŐèåď■åĹűåŐ§åğŃåĞ¡æŢřçŽĎåŔĆ
åęĆæđIJæČşèĞłåůśæĽŃåŁÍåőđçŐřçŽĎèřÍéIJĂèęĄåĄŽåď ğéĞŔçŽĎåůëä¡IJïijŇæIJĂåě¡åřśçőĂå■ŢçŽĎä¡£çŤ
@wraps èčĚéěřåŹÍãĂĆ éĂŽè£ĞåžŢåśĆçŽĎ __wrapped__
åśđæĂğèő£éŮőåĹřåĞ¡æŢřç■¿åŘ■ä£ąæĄŕãĂĆæŻť åď ŽåĚşäžŐç■¿åŘ■çŽĎåĘĚåőźåŔŕäżěåŔĆèĂČ9.16åřŔèŁ

9.3 èğčéŹď äÿĂäÿłèčĚéěřåŹÍ

éŮőécŸ

äÿĂäÿłèčĚéěřåŹÍåůšçżŔä¡IJçŤÍåIJÍäÿĂäÿłåĞ¡æŢřäŁïijŇä¡ăæČşæŠď éŤĂåőČïijŇçŻť æŐèèő£éŮőåŐ§åğ

èğčåĘşæŰźæąĹ

åĄĞèő¿èčĚéěřåŹÍæŸŕéĂŽè£Ğ @wraps (åŔĆèĂČ9.2åřŔèŁĆ)æİěåőđçŐřçŽĎïijŇéĆčäźĹä¡ääŔŕäżěéĂŽè
__wrapped__ åśđæĂğæİèèő£éŮőåŐ§åğŃåĞ¡æŢřïijŽ

```
>>> @somedecorator
>>> def add(x, y):
...     return x + y
...
>>> orig_add = add.__wrapped__
>>> orig_add(3, 4)
7
>>>
```

èőÍèőž

çŻť æŐèèő£éŮőæIJłåŇĚèčĚçŽĎåŐ§ågŃåĞ¡æŢřåIJÍèřČèřŢãĂĄåĘĚçIJĄåŠŇåĚűäżŮåĞ¡æŢřæŞ■ä¡IJæŮ
ä¡ĘæŸŕæĹŚäžňè£ŹéĞŇçŽĎæŰźæąĹäżĚäžĚéĂĆçŤÍäžŐåIJÍåŇĚèčĚåŹÍäÿ■æ■čçąőä¡£çŤÍäžĘ
`@wraps` æĹŰèĂĚçŻť æŐèèő¿ç¡őäžĘ `__wrapped__` åšđæĂğçŽĎæČĚåĘţãĂĆ

åęĆæđIJæIJL'åď'Žäÿ łåŇĚèčĚåŹÍïijŇéĆčäźĹèő£éŮő `__wrapped__`
åšđæĂğçŽĎèąŇäÿžæŸŕäÿ■åŔŕéćĎç§ěçŽĎïijŇåžŤèŕěéĄ£åĚ■è£ŹæăůåĄŽãĂĆ
åIJÍPython3.3äÿ■ïijŇåőČäijŽçŢěè£ĞæĽĂæIJL'çŽĎåŇĚèčĚåśĆïijŇærŤåęĆïijŇåĄĞåęĆä¡ăæIJL'åęĆäÿŃçŽĎ

```
from functools import wraps

def decorator1(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        print('Decorator 1')
        return func(*args, **kwargs)
    return wrapper

def decorator2(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        print('Decorator 2')
        return func(*args, **kwargs)
    return wrapper

@decorator1
@decorator2
def add(x, y):
    return x + y
```

äÿŃéİćæĹŚäžňåIJÍPython3.3äÿŃæţŃèŕŢïijŽ

```
>>> add(2, 3)
Decorator 1
Decorator 2
5
>>> add.__wrapped__(2, 3)
5
>>>
```

äÿŃéİćæĹŚäżňåIJÍPython3.4äÿŃæţŃèŕŢïijŽ

```
>>> add(2, 3)
Decorator 1
Decorator 2
5
>>> add.__wrapped__(2, 3)
Decorator 2
5
>>>
```

æIJĂåŘŐèęĄèŕť'çŽĎæŸŕïijŇåźűäÿ■æŸŕæL'ĂæIJL'çŽĎèčĚéěřåŹĺéČ¡ä¡£çŤĺäžĘ `@wraps` ïijŇåŻăæ■d'è£ŹéĞŇçŽĎæŰźæąĹåźűäÿ■åĚĺéČĺéĂĆçŤĺãĂĆçL'źåĹńçŽĎïijŇåĘĚç¡őçŽĎèčĚéěřåŹÍ `@staticmethod` åŠŇ `@classmethod` åřśæšąæIJL'éĄţå¿łè£ŹäÿłçžęåőŽ (åőČäżňæŁŁåŐ§åğŃåĞ¡æŢřå■ŸåĆĺåIJĺåśđæĂğ `__func__` äÿ■)ãĂĆ

9.4 åőŽäźL'äÿĂäÿłåÿęåŔĆæŢřçŽĎèčĚéěřåŹÍ

éŮőécŸ

ä¡ăæČşåőŽäźL'äÿĂäÿłåŔŕäžěæŐěåŔŮåŔĆæŢřçŽĎèčĚéěřåŹÍ

èğčåĘşæŰźæąĹ

æĹŚäżňçŤĺäÿĂäÿłä¿Ńå■ŘèŕęçżĕéŷŘè£ŕäÿŃæŐěåŔŮåŔĆæŢřçŽĎåď'ĎçŘĘè£ĞçÍŃãĂĆ åĄĞèő¿ä¡ăæČşåĘŹäÿĂäÿłècĚéěřåŹÍïijŇçżŹåĞ¡æŢřæůžåŁăæŮěå£ŮåŁ§èČ¡ïijŇå¡ŞæŮűåĚĄèőÿçŤĺæĹůæŇĞäÿŃéİćæŸŕè£ŹäÿłècĚéěřåŹÍçŽĎåőŽäźL'åŠŇä¡£çŤĺçď'žä¿ŃïijŽ

```
from functools import wraps
import logging

def logged(level, name=None, message=None):
    """
    Add logging to a function. level is the logging
    level, name is the logger name, and message is the
    log message. If name and message aren't specified,
    they default to the function's module and name.
    """
    def decorate(func):
        logname = name if name else func.__module__
        log = logging.getLogger(logname)
        logmsg = message if message else func.__name__

        @wraps(func)
        def wrapper(*args, **kwargs):
            log.log(level, logmsg)
            return func(*args, **kwargs)
```

```
        return wrapper
    return decorate

# Example use
@logged(logging.DEBUG)
def add(x, y):
    return x + y

@logged(logging.CRITICAL, 'example')
def spam():
    print('Spam!')
```

åĹİçIJŃèţůæİěïijŇè£Źçğ■åőđçŐřçIJŃäÿŁåŐżå¿Ĺåď■æİĆïijŇä¡ĘæŸŕæăÿå£ČæĂİæČşå¿ĹçőĂå■ŢãĂĆ æIJĂåď ŰåśĆçŽĎåĞ¡æŢř `logged()` æŐěåŔŮåŔĆæŢřáźűářĘåőČäżňä¡IJçŤĺåIJĺåĘĚéČĺçŽĎèčĚéěřåŹĺåĞ¡æ åĘĚåśĆçŽĎåĞ¡æŢř `decorate()` æŐěåŔŮäÿĂäÿłåĞ¡æŢřä¡IJäÿžåŔĆæŢřïijŇçĎűåŘŐåIJĺåĞ¡æŢřäŸŁéĺćæŤ è£ŹéĞŇçŽĎåĚşéŤőçĆźæŸŕåŇĚèčĚåŹĺæŸŕåŔŕäżěä¡£çŤĺäijåéĂŠçżŹ `logged()` çŽĎåŔĆæŢřçŽĎãĂĆ

èőÍèőž

åőŽäźĽäÿĂäÿłæŐěåŔŮåŔĆæŢřçŽĎåŇĚèčĚåŹĺçIJŃäÿŁåŐżæŕŤè¿Čåď■æİĆäÿżèęĄæŸŕåŽääÿžåžŢåśĆ

```
@decorator(x, y, z)
def func(a, b):
    pass
```

èčĚéěřåŹĺåď ĎçŘĘè£ĞçÍŃèů§äÿŃéİćçŽĎèřČçŤĺæŸŕç■Ľ æŢĹçŽĎ;

```
def func(a, b):
    pass
func = decorator(x, y, z)(func)
```

`decorator(x, y, z)` çŽĎè£ŤåŻđçżŞæđIJå£ĚéążæŸŕäÿĂäÿłåŔŕèřČçŤĺåŕźèśąïijŇåőČæŐěåŔŮäÿĂ åŔŕäžěåŔĆèĂČ9.7åřŔèŁĆäÿ■åŘęåď ŰäÿĂäÿłåŔŕæŐěåŔŮåŔĆæŢřçŽĎåŇĚèčĚåŹĺä¿Ńå■ŘãĂĆ

9.5 åŔŕèĞłåőŽäźĽåśđæĂğçŽĎèčĚéěřåŹĺ

éŮőécŸ

ä¡ăæČşåĘŹäÿĂäÿłèčĚéěřåŹĺæİěåŇĚèčĚäÿĂäÿłåĞ¡æŢřïijŇåźűäÿŤåĚĄèőÿçŤĺæĹůæŔŘä¿ŻåŔĆæŢřåIJĺè

èğčåĘşæŰźæąĹ

åijŢåĚěäÿĂäÿłèő£éŮőåĞ¡æŢřïijŇä¡£çŤĺ `nonlocal` æİěä£őæŤźåĘĚéČĺåŔŸéĞŘãĂĆ çĎűåŘŐè£Źäÿłèő£éŮőåĞ¡æŢřèćńä¡IJäÿžäÿĂäÿłåśđæĂğèţŃåĂijçżŹåŇĚèčĚåĞ¡æŢřãĂĆ

```python
from functools import wraps, partial
import logging
# Utility decorator to attach a function as an attribute of obj
def attach_wrapper(obj, func=None):
    if func is None:
        return partial(attach_wrapper, obj)
    setattr(obj, func.__name__, func)
    return func

def logged(level, name=None, message=None):
    '''
    Add logging to a function. level is the logging
    level, name is the logger name, and message is the
    log message. If name and message aren't specified,
    they default to the function's module and name.
    '''
    def decorate(func):
        logname = name if name else func.__module__
        log = logging.getLogger(logname)
        logmsg = message if message else func.__name__

        @wraps(func)
        def wrapper(*args, **kwargs):
            log.log(level, logmsg)
            return func(*args, **kwargs)

        # Attach setter functions
        @attach_wrapper(wrapper)
        def set_level(newlevel):
            nonlocal level
            level = newlevel

        @attach_wrapper(wrapper)
        def set_message(newmsg):
            nonlocal logmsg
            logmsg = newmsg

        return wrapper

    return decorate

# Example use
@logged(logging.DEBUG)
def add(x, y):
    return x + y

@logged(logging.CRITICAL, 'example')
def spam():
    print('Spam!')
```

äÿŃéÌćæŸŕäžd'äžŠçŐŕåćČäÿŃçŽĎä¡£çŤÍä¿Ńå■ŘïijŽ

```
>>> import logging
>>> logging.basicConfig(level=logging.DEBUG)
>>> add(2, 3)
DEBUG:__main__:add
5
>>> # Change the log message
>>> add.set_message('Add called')
>>> add(2, 3)
DEBUG:__main__:Add called
5
>>> # Change the log level
>>> add.set_level(logging.WARNING)
>>> add(2, 3)
WARNING:__main__:Add called
5
>>>
```

èőÍèőž

è£ŹäÿĂåřŔèŁĆçŽĎåĚşéŤőçĆźåIJÍäžŐèő£éŮőåĞ¡æŢř(åęĆ `set_message()` åŠŇ `set_level()`)ïijŇåőČäżňèćńä¡IJäÿžåśđæĂğèţŃçżŹåŇĚèčĚåŹÍãĂĆ æŕŔäÿłèő£éŮőåĞ¡æŢřåĚĄèőÿä¡£çŤÍ `nonlocal` æÌëä£őæŤźåĞ¡æŢřåĘĚéČÍçŽĎåŔŸéĞŔãĂĆ

è£ŸæIJL'äÿĂäÿłäžd'äžžåŘČæČŁçŽĎåIJřæŰźæŸŕèő£éŮőåĞ¡æŢřäijŽåIJÍåd'ŽåśĆèčĚéěřåŹÍéÚt'äijäæŠ■ `@functools.wraps` æşÍèğč)ãĂĆ ä¿ŃåęĆïijŇåĄĞèő¿ä¡ääijŢåĚěåŔęåd'ŰäÿĂäÿłèčĚéěřåŹÍïijŇærŤåęĆ9.2å `@timethis` ïijŇåČŔäÿŃéÌćè£Źæăůïijž

```
@timethis
@logged(logging.DEBUG)
def countdown(n):
    while n > 0:
        n -= 1
```

ä¡ääijŽåŔŚçŐřèő£éŮőåĞ¡æŢřä¿ÌæŮğæIJL'æŢĹïijž

```
>>> countdown(10000000)
DEBUG:__main__:countdown
countdown 0.8198461532592773
>>> countdown.set_level(logging.WARNING)
>>> countdown.set_message("Counting down to zero")
>>> countdown(10000000)
WARNING:__main__:Counting down to zero
countdown 0.8225970268249512
>>>
```

ä¡äè£ŸäijŽåŔŚçŐřå■şä¡£èčĚéěřåŹÍåČŔäÿŃéÌćè£ŹæăůäžèçŻÿåŔ■çŽĎæŰźåŘŚæŐŠæŤ¿ïijŇæŢĹæđIJäź

```
@logged(logging.DEBUG)
@timethis
def countdown(n):
    while n > 0:
        n -= 1
```

è£ŸèČ¡éĂŽè£Ğä¡£çŤĺlambdaèąĺè¿¿åijŔäžčçăĄæİèèőľèő£éŮőåĞ¡æŢřçŽĎè£ŤåŻđäÿ■åŘŇçŽĎèő¿åőŽåA

```
@attach_wrapper(wrapper)
def get_level():
    return level

# Alternative
wrapper.get_level = lambda: level
```

äÿĂäÿłærŤè¿ČéŽ¿çŘĘèğčçŽĎåIJřæŰźåřśæŸŕåŕźäžŐèő£éŮőåĞ¡æŢřçŽĎéęŰæňąä¡£çŤĺãĂĆä¿ŃåęĆïijŇ

```
@wraps(func)
def wrapper(*args, **kwargs):
    wrapper.log.log(wrapper.level, wrapper.logmsg)
    return func(*args, **kwargs)

# Attach adjustable attributes
wrapper.level = level
wrapper.logmsg = logmsg
wrapper.log = log
```

è£ŹäÿłæŰźæşŢäź§åŔŕèČ¡æ■čåÿÿåůëä¡IJïijÑä¡ĘåL’■æŔŘæŸŕåőČå£ĚéążæŸŕæIJĂåd’ŰåśÇçŽĎèčĚéěřåŹ
åęĆædIJåőČçŽĎäÿŁéİćè£ŸæIJL’åŔęåd’ŰçŽĎèčĚéěřåŹĺ(æŕŤåęĆäÿŁéİćæŔŘåĹřçŽĎ
`@timethis` ä¿Ńå■Ř)ïijŇéĆčäźĹåőČäijŽéŽŘèŮŔåžŢåśĆåśđæĂğïijÑä¡£å¿Ůä£őæŤźåőČäžñæšąæIJL’äżżä¡Ţ
èĂŇéĂŽè£Ğä¡£çŤĺèő£éŮőåĞ¡æŢřåŕśèČ¡éĄ£åĚ■è£ŹæăůçŽĎåśĂéŹŘæĂğãĂĆ

æIJĂåŘŐæŔŘäÿĂçĆźïijŇè£ŹäÿĂåřŔèŁĆçŽĎæŰźæąĹäź§åŔŕäžěä¡IJäÿž9.9åřŔèŁĆäÿ■èčĚéěřåŹĺçśżçŽĺ

9.6 åÿęåŔŕéĂĽåŔĆæŢřçŽĎèčĚéěřåŹĺ

éŮőécŸ

ä¡ăæČşåĘŹäÿĂäÿłèčĚéěřåŹĺïijŇæŮćåŔŕäžěäÿ■äijăåŔĆæŢřçżŹåőČïijŇæŕŤåęĆ
`@decorator` ïijŇ äź§åŔŕäžěäijăéĂŠåŔŕéĂĽåŔĆæŢřçżŹåőČïijŇæŕŤåęĆ
`@decorator(x,y,z)` ãĂĆ

èğčåĘşæŰźæąĹ

äÿŃéİćæŸŕ9.5åřŔèŁĆäÿ■æŮěå£ŮèčĚéěřåŹĺçŽĎäÿĂäÿłä£őæŤźçĽ’ĹæIJñïijŽ

```
from functools import wraps, partial
import logging
```

```
def logged(func=None, *, level=logging.DEBUG, name=None,␣
↪message=None):
    if func is None:
        return partial(logged, level=level, name=name,␣
↪message=message)

    logname = name if name else func.__module__
    log = logging.getLogger(logname)
    logmsg = message if message else func.__name__

    @wraps(func)
    def wrapper(*args, **kwargs):
        log.log(level, logmsg)
        return func(*args, **kwargs)

    return wrapper

# Example use
@logged
def add(x, y):
    return x + y

@logged(level=logging.CRITICAL, name='example')
def spam():
    print('Spam!')
```

åŔŕäžěçIJŃåĹřïijŇ`@logged`èčĚéěřåŹĺåŔŕäžěåŘŇæŮűäÿ■åÿęåŔĆæŢřæĹŰåÿęåŔĆæŢřãĂĆ

èőĺèőž

è£ŹéĞŇæŔŘåĹřçŽĎè£ŹäÿłéŮőécŸåřśæŸŕéĂŽåÿÿæĽĂèŕťçŽĎçijŰçĺŃäÿĂèĞťæĂğéŮőécŸãĂĆ
å¡ŞæĹŚäžňä¡£çŤĺèčĚéěřåŹĺçŽĎæŮűåĂŹïijŇåď ğéČĺåĹĘçĺŃåžŘåŚŸäźäæČŕäžĘèęĄäźĹäÿ■çżŹåőČäžňäijäéĂ
åĚűåðđäżŐæŁĂæIJŕäÿŁæİěèõšïijŇæĹŚäžňåŔŕäžěåőŽäźĽäÿĂäÿłæĽĂæIJĽåŔĆæŢřéČ¡æŸŕåŔŕéĂĽçŽĎèčĚ

```
@logged()
def add(x, y):
    return x+y
```

ä¡ĘæŸŕïijŇè£Źçğ■åĘŹæşŢåźűäÿ■çňęåŘĹæĹŚäžňçŽĎäźăæČŕïijŇæIJĽæŮűåĂŹçĺŃåžŘåŚŸå£ŸèőřåŁäă
è£ŹéĞŇæĹŚäžňåŘŚä¡ăåśŢçđ'žäžĘåęĆä¡Ţäžěäÿ ĂèĞť'çŽĎçijŰçĺŃéčŐæäijæİěåŘŇæŮűæżąęűşæšąæIJĽæŃňå

äÿžäžĘçŘĘèğčäžčçăĄæŸŕåęĆä¡Ţåůěä¡IJçŽĎïijŇä¡äéIJĂèęĄéİđåÿÿçE§æĆĽèčĚéěřåŹĺæŸŕåęĆä¡Ţä¡IJçŤ
ářźäžŐäÿĂäÿłåČŔäÿŃéİćè£ŹæăůçŽĎçőĂå■ŢèčĚéěřåŹĺïijŽ

```
# Example use
@logged
def add(x, y):
    return x + y
```

è£ŹäÿłèřčçŤĺåžŕåĹŮèů§äÿŃéİćç■L'äžůïijŽ

```
def add(x, y):
    return x + y

add = logged(add)
```

è£ŹæŮűåĂŹïijŇèćńèčĚéěřåĞ¡æŢřäijŽèćńå¡ŞåĄŽçňňäÿĂäÿłåŔĆæŢřçŻt'æŐëäijăéĂŠçżŹ
`logged` èčĚéěřåŹĺãĂĆ åŽăæ■d'ïijŇ`logged()` äÿ■çŽĎçňňäÿĂäÿłåŔĆæŢřåřśæŸŕèćńåŇĚèčĚåĞ¡æŢřæIJňè

èĂŇåŕźäžŐäÿĂäÿłäÿŃéİćè£Źæăůæ IJL'åŔĆæŢřçŽĎèčĚéěřåŹĺïijŽ

```
@logged(level=logging.CRITICAL, name='example')
def spam():
    print('Spam!')
```

èřČçŤĺåžŕåĹŮèů§äÿŃéİćç■L'äžůïijŽ

```
def spam():
    print('Spam!')
spam = logged(level=logging.CRITICAL, name='example')(spam)
```

åĹİåğŃèřČçŤĺ `logged()` åĞ¡æŢřæŮűïijŇèćńåŇĚèčĚåĞ¡æŢřåźűæšąæIJL'äijăéĂŠè£ŻæÍěãĂĆ
åŽăæ■d'åIJĺèčĚéěřåŹĺåĘĚïijŇåőČå£Ěéążæ ŸŕåŔŕéĂL'çŽĎãĂĆè£ŹäÿłåŔ■è£ĞæÍěäijŽè£ńä¡£åĚűäžŰåŔĆæŢřå
åźűäÿŤïijŇä¡Ęè£ŹäžŽåŔĆæŢřèćńäijăéĂŠè£ŻæÍěåŘŐïijŇèčĚéěřåŹĺèęĄè£ŤåŽđäÿĂäÿłæŐěåŔŮäÿĂäÿłåĞ¡æ
äÿžäžĘè£ŹæăůåĄŽïijŇæĹŚäžňä¡£çŤĺäžĘäÿĂäÿłæŁĂåůğïijŇåřśæŸŕåĹl'çŤĺ `functools.partial` ãĂĆ åőČäijŽè£ŤåŽđäÿĂäÿłæIJłåőŇåĚĺåĹİåğŃåŇŰçŽĎèĞłèžńïijŇéŹd'äžĘèćńåŇĚèčĚåĞ¡æŢřåd'
åŔŕäžěåŔĆèĂČ7.8åřŔèŁĆèŐůåŔŰæŻt'åd'Ž `partial()` æŰźæşŢçŽĎç§ěèŕĘãĂĆ

9.7 åĹl'çŤĺèčĚéěřåŹĺåijžåĹűåĞ¡æŢřäŸŁçŽĎçśžåđŃæčĂæ§ě

éŮőécŸ

ä¡IJäÿžæ§Řçğ■çijŰçÍŃèğĎçžęïijŇä¡ăæČşåIJĺåŕźåĞ¡æŢřåŔĆæŢřè£ŻèąŇåijžåĹűçśžåđŃæčĂæ§ěãĂĆ

èğčåĘşæŰźæąĹ

åIJÍæijŤçd'žåőđéŹĚäžčçăĄåL'■ïijŇåĚĹèŕt'æŸŐæĹŚäžňçŽĎçŻőæăĞïijŽèČ¡åŕźåĞ¡æŢřåŔĆæŢřçśžåđŃè

```
>>> @typeassert(int, int)
... def add(x, y):
...     return x + y
...
>>>
>>> add(2, 3)
5
>>> add(2, 'hello')
Traceback (most recent call last):
```

```
    File "<stdin>", line 1, in <module>
    File "contract.py", line 33, in wrapper
TypeError: Argument y must be <class 'int'>
>>>
```

äÿŃéÌćæŸŕä¡£çŤÍèčĚéěřåŹÍæŁĂæIJŕæÌěåőđçŐř `@typeassert` ïijŽ

```
from inspect import signature
from functools import wraps

def typeassert(*ty_args, **ty_kwargs):
    def decorate(func):
        # If in optimized mode, disable type checking
        if not __debug__:
            return func

        # Map function argument names to supplied types
        sig = signature(func)
        bound_types = sig.bind_partial(*ty_args, **ty_kwargs).
↪arguments

        @wraps(func)
        def wrapper(*args, **kwargs):
            bound_values = sig.bind(*args, **kwargs)
            # Enforce type assertions across supplied arguments
            for name, value in bound_values.arguments.items():
                if name in bound_types:
                    if not isinstance(value, bound_types[name]):
                        raise TypeError(
                            'Argument {} must be {}'.format(name,␣
↪bound_types[name])
                            )
            return func(*args, **kwargs)
        return wrapper
    return decorate
```

åŔŕäżěçIJŃåĞžè£ŹäÿłèčĚéěřåŹÍéÌđåÿÿçĄţæť'żïijŇæŮćåŔŕäżěæŇĞåőŽæĽĂæIJĽåŔĆæŢřçśžåđŃïijŇäź
åźűäÿŤåŔŕäżěéĂŽè£Ğä¡■ç¡őæĹŰåĚşéŤőå■ŮæÌěæŇĞåőŽåŔĆæŢřçśžåđŃãĂĆäÿŃéÌćæŸŕä¡£çŤÍçđ'žä¿ŃïijŽ

```
>>> @typeassert(int, z=int)
... def spam(x, y, z=42):
...     print(x, y, z)
...
>>> spam(1, 2, 3)
1 2 3
>>> spam(1, 'hello', 3)
1 hello 3
>>> spam(1, 'hello', 'world')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "contract.py", line 33, in wrapper
```

```
TypeError: Argument z must be <class 'int'>
>>>
```

èőĺèőž

è£ŹèŁĆæŸŕéńŸçžğèčĚéěřåŹĺçďžä¿ŃïijŇåijŢåĚëäžĘå¿Ĺåď'ŽéĞ■èęĄçŽĎæęĆå£ţãĂĆ

éęŰåĚĹïijŇèčĚéěřåŹĺåŔłäijŽåIJĺåĞ¡æŢřåőŽäźĽ'æŮűèćńèřČçŤĺäÿĂæňąãĂĆ æIJĽ'æŮűåĂŹä¡ääŐżæŐĽ'èčĚéěřåŹĺçŽĎåŁ§èČ¡ïijŇéĆčäźĹä¡ääŔłéIJĂèęĄçőĂå■ŢçŽĎè£ŤåŻđèćńèčĚéěřåĞ¡äÿŃéİćçŽĎäžčçăĄäÿ■ïijŇåęĆæđIJåĚĺåśĂåŔŸéĞŘãĂĂ__debug__ èćńèő¿ç¡őæĹŘäžĘFalse(å¡Şä¡ää¡£çŤĺ-OæĹŰ-OOåŔĆæŢřçŽĎäijŸåŇŰæĺąåijŕæĽ'ğèąŇçĺŃåžŕæŮű)ïijŇéĆčäźĹåřśçŻť'æŐèè£ŤåŻđæIJłä£őæŤźè£ĞçŽĎåĞ¡æŢřæIJňèžńïijŽ

```
def decorate(func):
    # If in optimized mode, disable type checking
    if not __debug__:
        return func
```

åĚűæňąïijŇè£ŹéĞŇè£ŸåŕźèćńåŇĚèčĚåĞ¡æŢřçŽĎåŔĆæŢřç■¿åŘ■è£ŻèąŇäžĘæčĂæ§ěïijŇæĹŚäžňä¡£çŤ inspect.signature() åĞ¡æŢřãĂĆ çőĂå■ŢæİěëőšïijŇåőČè£ŘèąŇä¡ăæŔŘåŔŰäÿĂäýłåŔŕèřČçŤĺåŕźèśąĚ

```
>>> from inspect import signature
>>> def spam(x, y, z=42):
...     pass
...
>>> sig = signature(spam)
>>> print(sig)
(x, y, z=42)
>>> sig.parameters
mappingproxy(OrderedDict([('x', <Parameter at 0x10077a050 'x'>),
('y', <Parameter at 0x10077a158 'y'>), ('z', <Parameter at␣
↪0x10077a1b0 'z'>)]))
>>> sig.parameters['z'].name
'z'
>>> sig.parameters['z'].default
42
>>> sig.parameters['z'].kind
<_ParameterKind: 'POSITIONAL_OR_KEYWORD'>
>>>
```

èčĚéěřåŹĺçŽĎåijĂåğŃéČĺåĹĘïijŇæĹŚäžňä¡£çŤĺäžĘ bind_partial() æŰźæşŢæİěæĽ'ğèąŇäžŐæŇĞåőŽçśžåđŃåĹřåŘ■çğřçŽĎéČĺåĹĘçżŚåőŽãĂĆ äÿŃéİćæŸŕä¿Ńå■ŘæijŤçď'žïijŽ

```
>>> bound_types = sig.bind_partial(int,z=int)
>>> bound_types
<inspect.BoundArguments object at 0x10069bb50>
>>> bound_types.arguments
```

```
OrderedDict([('x', <class 'int'>), ('z', <class 'int'>)])
>>>
```

åIJÍè£ŹäÿłéČÍåĹĘçżŚåőŽäÿ■ïijÑä¡ääŔŕäžěæşÍæĎŔåĹřçijžåď'śçŽĎåŔĆæŢřèćńå£¡çŢěäžĘ(æŕŤåęĆåźűæš
äÿ■è£ĞæIJĂéĞ■èęĄçŽĎæŸŕåĹŻåżžäžĘäÿĂäÿłæIJL'åžŔå■ŮåĚÿ `bound_types.arguments` ãĂĆ è£ŹäÿłåŮåĚÿäijŽåřĘåŔĆæŢřåŘ■äžěåĞ¡æŢřç■¿åŘ■äÿ■çŻÿåŘŇéąžåžŔæŸäářĎåĹřæŇĞå
åIJÍæĹŚäžñçŽĎèčĚéěřåŹÍä¿Ńå■ŘäÿïijÑè£ŹäÿłæŸäářĎåŇĚåŘńäžĘæĹŚäžñèęĄåijžåĹűæŇĞåőŽçŽĎçśżåđŃ

åIJÍèčĚéěřåŹÍåĹŻåżžçŽĎåőđéŹĚåŇĚèčĚåĞ¡æŢřäÿ■ä¡£çŤÍåĹřäžĘ `sig.bind()` æŰźæşŢãĂĆ `bind()` èů§ `bind_partial()` çśżäijijïijÑä¡ĘæŸŕåőČäÿ■åĚĄèőÿå£¡çŢěäżżä¡ŢåŔĆæŢřãĂĆåŻäæ■d'æIJL'äžĘäÿŃéÍćçŽĎçżŞæđIJïijŽ

```
>>> bound_values = sig.bind(1, 2, 3)
>>> bound_values.arguments
OrderedDict([('x', 1), ('y', 2), ('z', 3)])
>>>
```

ä¡£çŤÍè£ŹäÿłæŸäářĎæĹŚäžñåŔŕäžěå¿Ĺè¡żæİ¿çŽĎåőđçŐřæĹŚäžñçŽĎåijžåĹűçśżåđŃæčĂæ§ěïijŽ

```
>>> for name, value in bound_values.arguments.items():
...     if name in bound_types.arguments:
...         if not isinstance(value, bound_types.arguments[name]):
...             raise TypeError()
...
>>>
```

äÿ■è£Ğè£ŹäÿłæŰźæąĹè£ŸæIJL'çĆźářŔçŚŢçŰţïijÑåőČářźäžŐæIJL'éżŸèőď'åĂijçŽĎåŔĆæŢřåźűäÿ■éĂĆ
æŕŤåęĆäÿŃéÍćçŽĎäžčçăĄåŔŕäžěæ■čåÿÿåůěä¡IJïijÑář¡çőąitemsçŽĎçśżåđŃæŸŕéŤŹèŕŕçŽĎïijŽ

```
>>> @typeassert(int, list)
... def bar(x, items=None):
...     if items is None:
...         items = []
...     items.append(x)
...     return items
>>> bar(2)
[2]
>>> bar(2,3)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "contract.py", line 33, in wrapper
TypeError: Argument items must be <class 'list'>
>>> bar(4, [1, 2, 3])
[1, 2, 3, 4]
>>>
```

æIJĂåŘŐäÿĂçĆźæŸŕåĚşäžŐéĂĆçŤÍèčĚéěřåŹÍåŔĆæŢřåŠŇåĞ¡æŢřæşÍèğčäźŃéŮť'çŽĎäžL'èőžãĂĆ
ä¿ŃåęĆïijÑäÿžäżĂäźĹäÿ■åČŔäÿŃéÍćè£ŹæăůåĘŹäÿĂäÿłèčĚéěřåŹÍæÍěæ§ěæL'¿åĞ¡æŢřäÿ■çŽĎæşÍèğčåŚćïij§

```
@typeassert
def spam(x:int, y, z:int = 42):
```

```
    print(x,y,z)
```

äÿĂäÿłåŔŕèČ¡çŽĎåŐ§åŻăæŸŕåęĆæđIJä¡£çŤĺäžĘåĞ¡æŢřåŔĆæŢřæşĺèğčïijŇéĆčäźĹåřšèćńéŹŘåĹűäžĘãĄ
åęĆæđIJæşĺèğčèćńçŤĺæİěåĄŽçśżåđŃæčĂæ§ěåřšäÿ■èČ¡åĄŽåĚűäżŰäžŃæČĚäžĘãĂĆèĂŇäÿŤ
`@typeassert` äÿ■èČ¡åĘ■çŤĺäžŐä¡£çŤĺæşĺèğčåĄŽåĚűäżŰäžŃæČĚçŽĎåĞ¡æŢřäžĘãĂĆ
èĂŇä¡£çŤĺäÿŁéİćçŽĎèčĚéěřåŹĺåŔĆæŢřçĄţæť´żæĂğåď´ğåď´ŽäžĘïijŇäź§æŻť´åŁäéĂŽçŤĺãĂĆ

åŔŕäžěåIJÍPEP 362äžěåŔŁ `inspect` æĺąåİŮäÿ■æL´¿åĹřæŻť´åď´ŽåĚşäžŐåĞ¡æŢřåŔĆæŢřåŕźèśąçŽĎä£ąä

9.8 åřĘèčĚéěřåŹĺåőŽäźL´äÿžçśżçŽĎäÿĂéČĺåĹĘ

éŮőécŸ

ä¡ăæČşåIJÍçśżäÿ■åőŽäźL´èčĚéěřåŹĺïijŇåźűåřĘåĚűä¡IJçŤĺåIJĺåĚűäżŰåĞ¡æŢřæĹŰæŰźæşŢäÿŁãĂĆ

èğčåĘşæŰźæąĹ

åIJÍçśżéĞŇéİćåőŽäźL´èčĚéěřåŹĺå¿ĹçőĂå■ŢïijŇä¡ĘæŸŕä¡äéęŰåĚĹèęĄçąőèőď´åőČçŽĎä¡£çŤĺæŰźåijŔãĂ
äÿŃéİćæĹŚäżňçŤĺä¿Ńå■ŘæİěéŸŘè£řåőČäżňçŽĎäÿ■åŘŇïijŽ

```
from functools import wraps

class A:
    # Decorator as an instance method
    def decorator1(self, func):
        @wraps(func)
        def wrapper(*args, **kwargs):
            print('Decorator 1')
            return func(*args, **kwargs)
        return wrapper

    # Decorator as a class method
    @classmethod
    def decorator2(cls, func):
        @wraps(func)
        def wrapper(*args, **kwargs):
            print('Decorator 2')
            return func(*args, **kwargs)
        return wrapper
```

äÿŃéİćæŸŕäÿĂä¡£çŤĺä¿Ńå■ŘïijŽ

```
# As an instance method
a = A()
@a.decorator1
def spam():
    pass
# As a class method
```

```
@A.decorator2
def grok():
    pass
```

äžŤçżĘèğĆåŕ§åŔŕäžěåŔŚçŐřäÿĂäÿłæŸŕåőđä¿ŃèřČçŤĺïijŇäÿĂäÿłæŸŕçśżèřČçŤĺãĂĆ

èőĺèőž

åIJĺçśżäÿ■åőŽäźL'èčĚéěřåŹĺåĹİçIJŃäÿŁåŐżåě¡åČŔå¿ĹåěĞæĂłïijŇä¡ĘæŸŕåIJĺæăĞåĞĘåžŞäÿ■æIJL'å¿Ĺ
çL'źåĹńçŽĎïijŇ`@property` èčĚéěřåŹĺåőđéŹĚäÿŁæŸŕäÿĂäÿłçśżïijŇåőČéĞŇéİćåőŽäźL'äžĘäÿL'äÿłæŰźæşŢ
`getter()`, `setter()`, `deleter()`,ærŔäÿĂäÿłæŰźæşŢéČ¡æŸŕäÿĂäÿłèčĚéěřåŹĺãĂĆä¿ŃåęĆïijŽ

```
class Person:
    # Create a property instance
    first_name = property()

    # Apply decorator methods
    @first_name.getter
    def first_name(self):
        return self._first_name

    @first_name.setter
    def first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._first_name = value
```

åőČäÿžäżĂäźĹèęAè£ŹäźĹåőŽäźL'çŽĎäÿżèęAåŐ§åŽäæŸŕåŘĎçğ■äÿ■åŘŇçŽĎèčĚéěřåŹĺæŰźæşŢäijŽåI
`property` åőđä¿ŃäÿŁæŞ■ä¡IJåőČçŽĎçŁűæĂAąĂĆ åŽäæ■d'ïijŇäżżä¡ŢæŮűåĂŹåŔłèęAä¡ăçćřåĹřéIJĂèęAå

åIJĺçśżäÿ■åőŽäźL'èčĚéěřåŹĺæIJL'äÿłéŽ¿çŘĘèğčçŽĎåIJřæŰźåřśæŸŕåŕźäžŐéćİåd'ŰåŔĆæŢř
`self` æĹŰ `cls` çŽĎæ■čçąőä¡£çŤĺãĂĆ åř¡çőąæIJĂåd'ŰåśĆçŽĎèčĚéěřåŹĺåĞ¡æŢřæŕŤåęĆ
`decorator1()` æĹŰ `decorator2()` éIJĂèęAæŔŘä¿ŻäÿĂäÿł `self`
æĹŰ `cls` åŔĆæŢřïijŇ ä¡ĘæŸŕåIJĺäÿd'äÿłèčĚéěřåŹĺåĘĚéČĺèćńåĹŻåżžçŽĎ
`wrapper()` åĞ¡æŢřåźűäÿ■éIJĂèęAąåŇĚåŘńè£ŹäÿłåŔĆæŢřæŸŕåIJĺä¡ăçąőåőđèęAęő£éŮőåŇĚèčĚåŹĺäÿ■è£Źäÿłåőđä¿ŃçŽĎæ§ŘäžŻéČ

åŕźäžŐçśżéĞŇéİćåőŽäźL'çŽĎåŇĚèčĚåŹĺè£ŸæIJL'äÿĂçĆźæŕŤè¿ČéŽ¿çŘĘèğčïijŇåŕśæŸŕåIJĺæűL'åŘŁåĺ
ä¿ŃåęĆïijŇåĄĞèő¿ä¡ăæČşèőľåIJĺAäÿ■åőŽäźL'çŽĎèčĚéěřåŹĺä¡IJçŤĺåIJĺå■ŘçśżBäÿ■ãĂĆä¡ăéIJĂèęAąČŘäÿŇ

```
class B(A):
    @A.decorator2
    def bar(self):
        pass
```

äź§åŕśæŸŕèŕť'ïijŇèčĚéěřåŹĺèęAèćńåőŽäźL'æĹŘçśżæŰźæşŢåźűäÿŤä¡ăå£ĚéążæŸ¿åijŔçŽĎä¡£çŤĺçĹűçśż
ä¡ăäÿ■èČ¡ä¡£çŤĺ `@B.decorator2` ïijŇåŽăäÿžåIJĺæŰźæşŢåőŽäźL'æŮűïijŇè£ŹäÿłçśżBè£ŸæšąæIJL'èćńåĹŽ

9.9 åřĘèčĚéěřåŹĺåőŽäźĽäÿžçśż

éŮőécŸ

ä¡äæČşä¡£çŤĺäÿĂäÿłèčĚéěřåŹĺåŐżåŇĚèčĚåĞ¡æŢřïijŇä¡ĘæŸŕåÿŇæIJŻè£ŤåŻđäÿĂäÿłåŔŕèřČçŤĺçŽĎåő
ä¡äéIJĂèęĄèőľä¡äçŽĎèčĚéěřåŹĺåŔŕäżěåŘŇæŮűåůěä¡IJåIJĺçśżåőŽäźĽçŽĎåĘĚéČĺåŠŇåd'ŰéČĺãĂĆ

èğčåĘşæŰźæąĹ

äÿžäžĘåřĘèčĚéěřåŹĺåőŽäźĽæĹŘäÿĂäÿłåődä¿ŃïijŇä¡äéIJĂèęAçąőä£ÌåőČåődçŐřäžĘ
`__call__()` åŠŇ `__get__()` æŰźæşŢãĂĆ ä¿ŃåęĆïijŇäÿŃéÌćçŽĎäżčçăĄåőŽäźĽäžĘäÿĂäÿłçśżïijŇåőČå

```
import types
from functools import wraps

class Profiled:
    def __init__(self, func):
        wraps(func)(self)
        self.ncalls = 0

    def __call__(self, *args, **kwargs):
        self.ncalls += 1
        return self.__wrapped__(*args, **kwargs)

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return types.MethodType(self, instance)
```

ä¡ääŔŕäžěåřĘåőČå¡ŞåĄŽäÿĂäÿłæŹőéĂŽçŽĎèčĚéěřåŹĺæÌěä¡£çŤĺïijŇåIJĺçśżéĞŇéÌćæĹŰåd'ŰéÌćéČ¡åŔŕ

```
@Profiled
def add(x, y):
    return x + y

class Spam:
    @Profiled
    def bar(self, x):
        print(self, x)
```

åIJĺäžd'äžŠçŐŕáćČäÿ■çŽĎä¡£çŤĺçd'žä¿ŃïijŽ

```
>>> add(2, 3)
5
>>> add(4, 5)
9
>>> add.ncalls
2
```

```
>>> s = Spam()
>>> s.bar(1)
<__main__.Spam object at 0x10069e9d0> 1
>>> s.bar(2)
<__main__.Spam object at 0x10069e9d0> 2
>>> s.bar(3)
<__main__.Spam object at 0x10069e9d0> 3
>>> Spam.bar.ncalls
3
```

èőÍèőž

åřĘèčĚéěřåŹÍåőŽäźL'æĹŘçśżéĂŽåÿÿæŸřå¿ĹçőĂå■ŢçŽĎãĂĆä¡ĘæŸřè£ŹéĞŃè£ŸæŸřæIJL'äÿĂäžŽçżĘ

éęŰåĚĹïijŇä¡£çŤÍ `functools.wraps()` åĞ¡æŢřçŽĎä¡IJçŤÍèů§äźŃåL'■è£ŸæŸřäÿĂæăůïijŇåřĘèćńå

åĚűæňąïijŇéĂŽåÿÿå¿Ĺåőźæŷ$äijŽå£¡èğĘäÿŁéİćçŽĎ `__get__()` æŰźæşŢãĂĆåęĆædIJä¡åå£¡çŢěåőČïijŇä£ÌæŇĄåĚűäżŰäżčçăĄäÿ■åŔŸåĘ■æňąè£ŘèąŇïijŇ ä¡ăäijŽåŔŞçŐřå¡Şä¡ååŐżèřČçŤÍèćńèčĚéěřåőđä¿ŃæŰźæşŢæŮűåĞžçŐřå¿ĹåěĞæĂłçŽĎéŮőécŸãĂĆä¿ŃåęĆï

```
>>> s = Spam()
>>> s.bar(3)
Traceback (most recent call last):
...
TypeError: bar() missing 1 required positional argument: 'x'
```

åĞžéŤŹåŐ§åŻăæŸřå¡ŞæŰźæşŢåĞ¡æŢřåIJÍäÿĂäÿłçśżäÿ■èćńæ§ěæL'¿æŮűïijŇåőČäżňçŽĎ `__get__()` æŰźæşŢä¿İæ■őæŔŔè£řåŹÍå■ŔèőőèćńèřČçŤÍïijŇ åIJÍ8.9åřŔèŁĆåůšçżŔèőšè£řè£ĞæŔŔè£řåŹÍå■ŔèőőäžĘãĂĆåIJÍè£ŹéĞŃïijŇ`__get__()` çŽĎçŽőçŽĎæŸřåĹŻåżžäÿĂäÿłçżŠåőŽæŰźæşŢårźèšą (æIJĂçżĹäijŽçżŹè£ŹäÿłæŰźæşŢäijăéĂŠselfåŔĆæŢř)ă

```
>>> s = Spam()
>>> def grok(self, x):
...     pass
...
>>> grok.__get__(s, Spam)
<bound method Spam.grok of <__main__.Spam object at 0x100671e90>>
>>>
```

`__get__()` æŰźæşŢæŸřäÿžäžĘçąőä£İçżŠåőŽæŰźæşŢårźèšąèČ¡èćńæ■čçąőçŽĎåĹŻåżžãĂĆ `type.MethodType()` æL'ŃåŁĺåĹŻåżžäÿĂäÿłçżŠåőŽæŰźæşŢæİěä¡£çŤÍãĂĆåŔłæIJL'å¡Şåőđä¿Ńèćńä¡£çŤ åęĆædIJè£ŹäÿłæŰźæşŢæŸřåIJÍçśżäÿŁéİćæİèèő£éŮőïijŇ éĆčäźĹ `__get__()` äÿ■çŽĎinstanceåŔĆæŢřäijŽèćńèő¿ç¡őæĹŘNoneåźűçŻt'æŐěè£ŤåŽđ `Profiled` åőđä¿ŃæIJňèžńãĂĆ è£ŹæăůçŽĎèřİæĹŚäžňårśåŔřäżěæŔŘåŔŰåőČçŽĎ `ncalls` åśđæĂğäžĘãĂĆ

åęĆædIJä¡ăæČşéĄ£åĚ■äÿĂäžŽæůůäźśïijŇäź§åŔřäżěèĂČèŹŚåŔęåd'ŰäÿĂäÿłä¡£çŤÍéŮ■åŇĚåŠŇ `nonlocal` åŘŸéĞŘåőđçŐřçŽĎèčĚéěřåŹÍïijŇè£ŹäÿłåIJÍ9.5åřŔèŁĆæIJL'èőšåĹřãĂĆä¿ŃåęĆïijŽ

```
import types
from functools import wraps
```

```
def profiled(func):
    ncalls = 0
    @wraps(func)
    def wrapper(*args, **kwargs):
        nonlocal ncalls
        ncalls += 1
        return func(*args, **kwargs)
    wrapper.ncalls = lambda: ncalls
    return wrapper

# Example
@profiled
def add(x, y):
    return x + y
```

è£ŹäÿłæŰźåijŔèů§äźŃåL’■çŽĎæŢĹæđIJåĞääźŐäÿĂæăůïijŇéŹď’äžĘåŕźäžŐ `ncalls` çŽĎèő£éŮőçŐřåIJĺæŸŕéĂŽè£ĞäÿĂäÿłèćńçżŚåőŽäÿžåśđæĂğçŽĎåĞ¡æŢřæİěåőđçŐřïijŇä¿ŃåęĆïijŽ

```
>>> add(2, 3)
5
>>> add(4, 5)
9
>>> add.ncalls()
2
>>>
```

9.10 äÿžçśżåŠŇéİŹæĂĄæŰźæşŢæŔŘä¿ŻèčĚéěřåŹÍ

éŮőécŸ

ä¡ăæČşçżŹçśżæĹŰéİŹæĂĄæŰźæşŢæŔŘä¿ŻèčĚéěřåŹÍãĂĆ

èğčåĘşæŰźæąĹ

çżŹçśżæĹŰéİŹæĂĄæŰźæşŢæŔŘä¿ŻèčĚéěřåŹÍæŸŕå¿ĹçőĂå■ŢçŽĎïijŇäÿ■è£ĞèęĄçąőä£İèčĚéěřåŹÍåIJ `@classmethod` æĹŰ `@staticmethod` äźŃåL’■ãĂĆä¿ŃåęĆïijŽ

```
import time
from functools import wraps

# A simple decorator
def timethis(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.time()
        r = func(*args, **kwargs)
```

```
        end = time.time()
        print(end-start)
        return r
    return wrapper

# Class illustrating application of the decorator to different␣
↪kinds of methods
class Spam:
    @timethis
    def instance_method(self, n):
        print(self, n)
        while n > 0:
            n -= 1

    @classmethod
    @timethis
    def class_method(cls, n):
        print(cls, n)
        while n > 0:
            n -= 1

    @staticmethod
    @timethis
    def static_method(n):
        print(n)
        while n > 0:
            n -= 1
```

èčĚéěřåŘŐçŽĎçśžåŠŇéİŹæĂĄæŰźæşŢåŔŕæ■čåÿÿåůěä¡IJïijŇåŔłäÿ■è£Ğåćđå£ääžĘéćİåď˙ŰçŽĎèőąæŮ

```
>>> s = Spam()
>>> s.instance_method(1000000)
<__main__.Spam object at 0x1006a6050> 1000000
0.11817407608032227
>>> Spam.class_method(1000000)
<class '__main__.Spam'> 1000000
0.11334395408630371
>>> Spam.static_method(1000000)
1000000
0.11740279197692871
>>>
```

èőÍèőž

åęĆæđIJä¡ăæŁŁèčĚéěřåŹÍçŽĎéąžåžŘåĘŹéŤŹäžĘåřśäijŽåĞžéŤŹãĂĆä¿ŃåęĆïijŇåĄĞèő¿ä¡ääČŔäÿŃéİć

```
class Spam:
    @timethis
    @staticmethod
```

```
    def static_method(n):
        print(n)
        while n > 0:
            n -= 1
```

éĆčäźĹä¡ăèřČçŤĺè£ŹäÿłéİŹæĂĄæŰźæşŢæŮűůřśäijŽæŁěéŤŹïijŽ

```
>>> Spam.static_method(1000000)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "timethis.py", line 6, in wrapper
start = time.time()
TypeError: 'staticmethod' object is not callable
>>>
```

éŮőécŸåIJĺäžŐ `@classmethod` åŠŇ `@staticmethod` åőđéŹĚäÿŁåźűäÿ■äijŽåĹŻåžžåŘŕçŻťæŐěèřČçŤĺçŽĎåŕźèśąïijŇ
èĂŇæŸŕåĹŻåžžçĽźæőŁçŽĎæŔŘè£ŕåŹĺåŕźèśą(åŘĆèĂČ8.9åřŔèŁĆ)ãĂĆåŻäæ■ď¡å¦Şä¡ăèŕŢçİĂåIJĺåĚűäżŰèč
çąőä£Ièź£Źçğ■èčĚéěřåŹĺåĞžçŐřåIJĺèčĚéěřåŹĺéŞ¿äÿ■çŽĎçňnäÿĂäÿłä¡■ç¡őåŔŕäžëä£őåď■è£ŹäÿłéŮőécŸãĂĆ

å¡ŞæĹŚäžňåIJĺæŁ¡èśąå§žçśżäÿ■åőŽäźL'çśżæŰźæşŢåŠŇéİŹæĂĄæŰźæşŢ(åŘĆèĂČ8.12åřŔèŁĆ)æŮűïijŇ
ä¿ŃåęĆïijŇåęĆæđIJä¡ăæČşåőŽäźL'äÿĂäÿłæŁ¡èśąçśżæŰźæşŢïijŇåŔŕäžëä¡£çŤĺçśżäijijäÿŃéİćçŽĎäžčçăĄïijŽ

```
from abc import ABCMeta, abstractmethod
class A(metaclass=ABCMeta):
    @classmethod
    @abstractmethod
    def method(cls):
        pass
```

åIJĺè£Źæőţäžčçă Ąäÿ■ïijŇ`@classmethod` èů§ `@abstractmethod` äÿd'èĂĚçŽĎéąžåžŔæŸŕæIJL'èőšçl'űçŽĎïijŇåęĆæđIJä¡ăèřČæ■ćåőČäžňçŽĎéąžåžŔåřśäijŽåĞžéŤŹãĂĆ

9.11 èčĚéěřåŹĺäÿžèćńåŇĚèčĚåĞ¡æŢřåćđåŁäåŔĆæŢř

éŮőécŸ

ä¡ăæČşåIJĺèčĚéěřåŹĺäÿ■çżŹèćńåŇĚèčĚåĞ¡æŢřåćđåŁäécİåď'ŰçŽĎåŔĆæŢřïijŇä¡ĘæŸŕäÿ■èČ¡å¡śåŞ■è£Ź

èğčåĘşæŰźæąĹ

åŔŕäžëä¡£çŤĺåĚşéŤőå■ŮåŔĆæŢřæİęçżŹèćńåŇĚèčĚåĞ¡æŢřåćđåŁäécİåď'ŰåŔĆæŢřãĂĆèĂČèŹŚäÿŃéİćç

```
from functools import wraps

def optional_debug(func):
    @wraps(func)
    def wrapper(*args, debug=False, **kwargs):
```

```
        if debug:
            print('Calling', func.__name__)
        return func(*args, **kwargs)

    return wrapper
```

```
>>> @optional_debug
... def spam(a,b,c):
... print(a,b,c)
...
>>> spam(1,2,3)
1 2 3
>>> spam(1,2,3, debug=True)
Calling spam
1 2 3
>>>
```

èőÍèőž

éĂŽè£ĞèčĚéěřåŹÍæİěçżŹèćńåŇĚèčĚåĞ¡æŢřåćđåŁăåŔĆæŢřçŽĎåĄŽæşŢåźűäÿ■åÿÿèğĄãĂĆ
ář¡çőąåęĆæ■ď'ïijŇæIJL'æŮűåĂŹåőČåŔŕäžěéĄ£åĚ■äÿĂäžŻéĞ■åď'■äžčçăĄãĂĆä¿ŃåęĆïijŇåęĆæđIJä¡ăæIJL'

```
def a(x, debug=False):
    if debug:
        print('Calling a')

def b(x, y, z, debug=False):
    if debug:
        print('Calling b')

def c(x, y, debug=False):
    if debug:
        print('Calling c')
```

éĆčäźĹä¡ăåŔŕäžěářĘåĚűéĞ■æđĎæĹŘè£ŹæăůïijŽ

```
from functools import wraps
import inspect

def optional_debug(func):
    if 'debug' in inspect.getargspec(func).args:
        raise TypeError('debug argument already defined')

    @wraps(func)
    def wrapper(*args, debug=False, **kwargs):
        if debug:
            print('Calling', func.__name__)
        return func(*args, **kwargs)
    return wrapper
```

```
@optional_debug
def a(x):
    pass

@optional_debug
def b(x, y, z):
    pass

@optional_debug
def c(x, y):
    pass
```

è£Źçğ■åőđçŐřæŰźæąĹäźŃæĽ’Ăäžěèą Ňå¿ŮéĂŽïijŇåIJĺäžŐåijžåĹűåĚşéŤőå■ŮåŔĆæŢřå¿Ĺåőźæ ŸŞèćńa
`*args` åŠŇ `**kwargs` åŔĆæŢřçŽĎåĞ¡æŢřäÿ■ãĂĆ éĂŽè£Ğä¡£çŤĺåijžåĹűåĚşéŤőå■ŮåŔĆæŢřïijŇåőČèćńä
åźűäÿŤæŐěäÿŃæİěäżĚäżĚä¡£çŤĺåĽ’ľä¡ŹçŽĎä¡■ç¡őåŠŇåĚşéŤőå■ŮåŔĆæŢřåŐżèřČçŤĺè£ŹäÿłåĞ¡æŢřæŮűïij
äź§åřśæŸŕèŕť’ïijŇåőČåźűäÿ■äijŽèćńçžşåĚěåĹř `**kwargs` äÿ■åŐżãĂĆ

è£ŸæIJL’äÿĂäÿłéŽ¿çĆźåřśæŸŕåęĆä¡ŢåŐżåď’ĎçŘĘèćńæůżåŁăçŽĎåŔĆæŢřäÿŐèćńåŇĚèčĚåĞ¡æŢřåŔĆ
ä¿ŃåęĆïijŇåęĆædIJèčĚéěřåŹÍ `@optional_debug` ä¡IJçŤĺåIJĺäÿĂäÿłåůšçżŔæŃěæIJL’äÿĂäÿł
`debug` åŔĆæŢřçŽĎåĞ¡æŢřäŸŁæŮűäijŽæIJL’éŮőécŸãĂĆ è£ŹéĞŇæĹŚäžňáćđåŁääžĘäÿĂæ■ěåŘ■å■ŮæčĂæ

äÿŁéİćçŽĎæŰźæąĹè£ŸåŔŕäžěæŻť åőŇç¿ŐäÿĂçĆźïijŇåŻääÿžçš¿æŸŐçŽĎçÍŃåžŔåŚŸåžŤèŕěåŔŚçŐřäž

```
>>> @optional_debug
... def add(x,y):
...     return x+y
...
>>> import inspect
>>> print(inspect.signature(add))
(x, y)
>>>
```

éĂŽè£ĞåęĆäÿŃçŽĎä£őæŤźïijŇåŔŕäžěèğčåĘşè£ŹäÿłéŮőécŸïijŽ

```
from functools import wraps
import inspect

def optional_debug(func):
    if 'debug' in inspect.getargspec(func).args:
        raise TypeError('debug argument already defined')

    @wraps(func)
    def wrapper(*args, debug=False, **kwargs):
        if debug:
            print('Calling', func.__name__)
        return func(*args, **kwargs)

    sig = inspect.signature(func)
    parms = list(sig.parameters.values())
    parms.append(inspect.Parameter('debug',
```

```
                inspect.Parameter.KEYWORD_ONLY,
                default=False))
    wrapper.__signature__ = sig.replace(parameters=parms)
    return wrapper
```

éĂŽè£Ğè£ŹæăůçŽĎä£őæŤźïijŇåŇĚèčĚåŘŐçŽĎåĞ¡æŢřç■¿åŘ■åřśèČ¡æ■čçąőçŽĎæŸ¿çd'ž debug åŔĆæŢřçŽĎå■ŸåIJÍäžĘãĂĆä¿ŃåęĆïijŽ

```
>>> @optional_debug
... def add(x,y):
...     return x+y
...
>>> print(inspect.signature(add))
(x, y, *, debug=False)
>>> add(2,3)
5
>>>
```

åŔĆèĂČ9.16åřŔèŁĆèŐůåŔŰæŻt'åd'ŽåĚşäžŐåĞ¡æŢřç■¿åŘ■çŽĎä£ąæĄŕãĂĆ

9.12 ä¡£çŤÍèčĚéěřåŹÍæĽľåĚĚçśżçŽĎåŁ§èČ¡

éŮőécŸ

ä¡ăæČşéĂŽè£ĞåŔ■çIJĄæĹŰèĂĚéĞ■åĘŹçśżåőŽäźL'çŽĎæ§ŘéČĺåĹĘæİëä£őæŤźåőČçŽĎèąŇäÿžïijŇä¡Ę

èğčåĘşæŰźæąĹ

è£Źçğ■æČĚåĘţåŔŕèČ¡æŸŕçśżèčĚéěřåŹÍæIJĂåě¡çŽĎä¡£çŤÍåIJžæŹŕäžĘãĂĆä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿĂä __getattribute__ çŽĎçśżèčĚéěřåŹÍïijŇ åŔŕäžěæĽŞå■řæŮěå£ŮïijŽ

```
def log_getattribute(cls):
    # Get the original implementation
    orig_getattribute = cls.__getattribute__

    # Make a new definition
    def new_getattribute(self, name):
        print('getting:', name)
        return orig_getattribute(self, name)

    # Attach to the class and return
    cls.__getattribute__ = new_getattribute
    return cls

# Example use
@log_getattribute
class A:
    def __init__(self,x):
```

```
        self.x = x
    def spam(self):
        pass
```

äÿŃéÌćæŸŕä¡£çŤÍæŢĹæđIJïijŽ

```
>>> a = A(42)
>>> a.x
getting: x
42
>>> a.spam()
getting: spam
>>>
```

èőÍèőž

çśžèčĚéěřåŹÍéĂŽåÿÿåŔŕäžěä¡IJäÿžåĚűäžŰéńŸçžğæŁĂæIJŕæŕŤåęĆæůůåĚěæĹŰåĚČçśžçŽĎäÿĂçğ■éİđă
æŕŤåęĆïijŇäÿŁéİćçđ'žä¿Ńäÿ■çŽĎåŔęåđ'ŰäÿĂçğ■åőđçŐřä¡£çŤÍåĹŕçžğæL'£ïijŽ

```
class LoggedGetattribute:
    def __getattribute__(self, name):
        print('getting:', name)
        return super().__getattribute__(name)

# Example:
class A(LoggedGetattribute):
    def __init__(self,x):
        self.x = x
    def spam(self):
        pass
```

è£Źçğ■æŰźæąĹäź§èąŇå¿ŮéĂŽïijŇä¡ĘæŸŕäÿžäžĘåŐżçŘĘèğčåőČïijŇä¡ăåřśå£Ěéążç§ěéĄŞæŰźæşŢèřČ
äžěåŔŁåĚűåőČ8.7åřŔèŁĆäżŃçż■çŽĎçžğæL'£ç§ěèřĘãĂĆ æ§Řçğ■çÍŃåžęäÿŁæİèèőšïijŇçśžèčĚéěřåŹÍæŰźæą
åŻääÿžäžŰåźűäÿ■ä¿İètŰ `super()` åĞ¡æŢřãĂĆ

åęĆæđIJä¡ăçşżæČşåIJÍäÿĂäÿłçśžäÿŁéİćä¡£çŤÍåđ'ŽäÿłçśžèčĚéěřåŹÍïijŇéĆčäźĹåřśéIJĂèęĄæşÍæĎŔäÿŃé
ä¿ŃåęĆïijŇäÿĂäÿłèčĚéěřåŹÍAäijŽåřĘåĚűèčĚéěřçŽĎæŰźæşŢåőŇæŢt'æŻ£æ■ćæĹŘåŘęäÿĂçğ■åőđçŐřïijŇ
èĂŇåŔęäÿĂäÿłèčĚéěřåŹÍBåŔłæŸŕçőĂå■ŢçŽĎåIJÍåĚűèčĚéěřçŽĎæŰźæşŢäÿ■æůžåŁăçĆźéĆİåđ'ŰéĂżè¿ŚãĂĆ
éĆčäźĹè£ŹæŮűåĂŹèčĚéěřåŹÍAåřśéIJĂèęĄæŤ¿åIJÍèčĚéěřåŹÍBçŽĎåL'■éİćãĂĆ

ä¡ăè£ŸåŔŕäžěåŻđéą¿äÿĂäÿŃ8.13åřŔèŁĆåŘęåđ'ŰäÿĂäÿłåĚşäžŐçśžèčĚéěřåŹÍçŽĎæIJL'çŤÍçŽĎä¿Ńå■Ř

9.13 ä¡£çŤÍåĚČçśżæŐğåĹűåőđä¿ŃçŽĎåĹŻåżž

éŮőéćŸ

ä¡ăæČşéĂŽè£ĞæŤźåŔŸåőđä¿ŃåĹŻåżžæŰźåijŔæİèåőđçŐřå■Ţä¿ŃãĂĄçijŞå■ŸæĹŰåĚűäžŰçśžäijijçŽĎ

èğčåĘşæŰźæąĹ

PythonçÍŃåžŔåŚŸéČ¡ç§ěéĄŞïijŇåęĆæđIJä¡ăåőŽäźĽäžĘäÿĂäÿłçśżïijŇářśèČ¡åČŔåĞ¡æŢřäÿĂæăůçŽĎè

```
class Spam:
    def __init__(self, name):
        self.name = name

a = Spam('Guido')
b = Spam('Diana')
```

åęĆæđIJä¡ăæČşèĞłåőŽäźĽè£Źäÿłæ■ěéłd'ïijŇä¡ăåŔŕäžěåőŽäźĽäÿĂäÿłåĚČçśżåźűèĞłåůśåőđçŐř __call__() æŰźæşŢãĂĆ

äÿžäžĘæijŤçd'žïijŇåĄĞèő¿ä¡ăäÿ■æČşäżżä¡ŢäžžåĹŻåżžè£Źäÿłçśżç ŽĎåőđä¿ŃïijŽ

```
class NoInstances(type):
    def __call__(self, *args, **kwargs):
        raise TypeError("Can't instantiate directly")

# Example
class Spam(metaclass=NoInstances):
    @staticmethod
    def grok(x):
        print('Spam.grok')
```

è£ŹæăůçŽĎèŕÍïijŇçŤÍæĹůåŔłèČ¡èřČçŤÍè£Źäÿłçśżç ŽĎéİŹæĂĄæŰźæşŢïijŇèĂŇäÿ■èČ¡ä¡£çŤÍéĂŽåÿÿç

```
>>> Spam.grok(42)
Spam.grok
>>> s = Spam()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "example1.py", line 7, in __call__
        raise TypeError("Can't instantiate directly")
TypeError: Can't instantiate directly
>>>
```

çŐřåIJÍïijŇåĄĞåęĆä¡ăæČşåőđçŐřå■Ţä¿ŃæÍąåijŔïijĹåŔłèČ¡åĹŻåżžåŤŕäÿĂåőđä¿ŃçŽĎçśżïijL'ïijŇåőđçĆ

```
class Singleton(type):
    def __init__(self, *args, **kwargs):
        self.__instance = None
        super().__init__(*args, **kwargs)

    def __call__(self, *args, **kwargs):
        if self.__instance is None:
            self.__instance = super().__call__(*args, **kwargs)
            return self.__instance
        else:
            return self.__instance
```

```
# Example
class Spam(metaclass=Singleton):
    def __init__(self):
        print('Creating Spam')
```

éĆčäźĹSpamçśżåřśåŘłèČ¡åĹŻåżžåŤŕäÿĂçŽĎåőđä¿ŃäžĘïijŇæijŤçd’žåęĆäÿŃïijŽ

```
>>> a = Spam()
Creating Spam
>>> b = Spam()
>>> a is b
True
>>> c = Spam()
>>> a is c
True
>>>
```

æIJĂåŘŐïijŇåĄĞèő¿ä¡äæČşåĹŻåżž8.25åřŔèŁĆäÿ■éĆčæăůçŽĎçijŞå■ŸåőđäżŃãĂĆäÿŃéİćæĹŚäžňåŔŕä

```
import weakref

class Cached(type):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.__cache = weakref.WeakValueDictionary()

    def __call__(self, *args):
        if args in self.__cache:
            return self.__cache[args]
        else:
            obj = super().__call__(*args)
            self.__cache[args] = obj
            return obj

# Example
class Spam(metaclass=Cached):
    def __init__(self, name):
        print('Creating Spam({!r})'.format(name))
        self.name = name
```

çĎűåŘŐæĹŚäź§æİěætŃèŕŢäÿĂäÿŃïijŽ

```
>>> a = Spam('Guido')
Creating Spam('Guido')
>>> b = Spam('Diana')
Creating Spam('Diana')
>>> c = Spam('Guido') # Cached
>>> a is b
False
>>> a is c # Cached value returned
```

```
True
>>>
```

èőÍèőž

åĹľçŤĺåĚČçśżåőđçŐřåď'Žçğ■åőđä¿ŃåĹŻåżžæÍąåijŔéĂŽåÿÿèęĄæŕŤäÿ■ä¡£çŤĺåĚČçśżçŽĎæŰźåijŔäijŸ

åĄĞèő¿ä¡ääÿ■ä¡£çŤĺåĚČçśżïijÑä¡ăåŔřèČ¡éIJĂèęĄåřĘçśżéŽŘèŮŔåIJÍæ§ŘäžŻåůěåŐĆåĞ¡æŢřåŘŐéİćãĂ
æŕŤåęĆäÿžäžĘåőđçŐřäÿĂäÿłå■Ţä¿ŃïijÑä¡ää¡ååŔřèČ¡äijŽåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
class _Spam:
    def __init__(self):
        print('Creating Spam')

_spam_instance = None

def Spam():
    global _spam_instance

    if _spam_instance is not None:
        return _spam_instance
    else:
        _spam_instance = _Spam()
        return _spam_instance
```

ař¡çőąä¡£çŤĺåĚČçśżåŔřèČ¡äijŽæűL'åŘŁåĹřæŕŤè¿ČéńŸçžğçĆźçŽĎæŁĂæIJŕïijÑä¡ĘæŸŕåőČçŽĎäżčçăĄ

æŻť'åď'ŽåĚşäžŐåĹŻåżžçijŞå■Ÿåőđä¿ŃãĂĄåijśåijŢçŤĺç■L'åĘĚåőźïijÑèŕůåŘĆèĂČ8.25ařŘèŁĆãĂĆ

9.14 æ■ŢèŐůçśżçŽĎåśđæĂğåőŽäźL'éąžåžŔ

éŮőécŸ

ä¡ăæČşèĞłåŁĺèőřå¡ŢäÿĂäÿłçśżäÿ■åśđæĂğåŠÑæŰźæşŢåőŽäźL'çŽĎéąžåžŔïijÑ
çĎűåŘŐåŘŕäżěåĹľçŤĺåőČæİěåĄŽå¿Ĺåď'ŽæŞ■ä¡IJïijĹæŕŤåęĆåžŘåĹŮåŇŰãĂĄæŸäărĎåĹřæŢřæ■őåžŞç■L'ç

èğčåĘşæŰźæąĹ

åĹľçŤĺåĚČçśżåŘŕäżěå¿ĹåőźæŸŞçŽĎæ■ŢèŐůçśżçŽĎåőŽäźL'ä£ąæĄřãĂĆäÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijÑ

```
from collections import OrderedDict

# A set of descriptors for various types
class Typed:
    _expected_type = type(None)
    def __init__(self, name=None):
        self._name = name
```

```
    def __set__(self, instance, value):
        if not isinstance(value, self._expected_type):
            raise TypeError('Expected ' + str(self._expected_type))
        instance.__dict__[self._name] = value

class Integer(Typed):
    _expected_type = int

class Float(Typed):
    _expected_type = float

class String(Typed):
    _expected_type = str

# Metaclass that uses an OrderedDict for class body
class OrderedMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        d = dict(clsdict)
        order = []
        for name, value in clsdict.items():
            if isinstance(value, Typed):
                value._name = name
                order.append(name)
        d['_order'] = order
        return type.__new__(cls, clsname, bases, d)

    @classmethod
    def __prepare__(cls, clsname, bases):
        return OrderedDict()
```

åIJÍè£ŹäýłåĚČçśżäÿ■ïijŇæL’ğèąŇçśżäÿżä¡ŞæÙűæŔŔè£řåŹÍçŽĎåőŽäźL’éąžåžŔäijŽèćńäÿĂäýł
OrderedDict``æ■ŢèŐůåĹřïijŇ çŤ$æĹŘçŽĎæIJL’åžŔåŘ■çğřäżŐå■ŮåĚÿäÿ■æŔŘåŔŰåĞžæİě
``_order äÿ■ãĂĆè£Źæăůç ŽĎèŕÍçśżäÿ■çŽĎæŰźæşŢåŔřäżěéĂŽè£Ğåď’Žçğ■æŰźåijŔæİěä¡£çŤĺåőČãĂĆ
ä¿ŇåęĆïijŇäÿŇéİćæŸŕäÿĂäýłçőĂå■ŢçŽĎçśżïijŇä¡£çŤĺè£ŹäýłæŐŠåžŔå■ŮåĚÿæİěåőđçŐřårĘäÿĂäýłçśżåőđă

```
class Structure(metaclass=OrderedMeta):
    def as_csv(self):
        return ','.join(str(getattr(self,name)) for name in self._
↪order)

# Example use
class Stock(Structure):
    name = String()
    shares = Integer()
    price = Float()

    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
```

```
        self.price = price
```

æĹŚäżňåIJĺäžďäžŠåijŘçŐŕåćČäÿ■æţŃèŕŢäÿĂäÿŃè£ŹäÿłStockçśżïijŽ

```
>>> s = Stock('GOOG',100,490.1)
>>> s.name
'GOOG'
>>> s.as_csv()
'GOOG,100,490.1'
>>> t = Stock('AAPL','a lot', 610.23)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "dupmethod.py", line 34, in __init__
TypeError: shares expects <class 'int'>
>>>
```

èőĺèőž

æIJňèŁĆäÿĂäÿłåĚşéŤőçĆźåřśæŸŕOrderedMetaåĚČçśżäÿ■åőŽäźL’çŽĎ ‘‘ __prepare__()‘‘ æŰźæşŢãĂĆ è£ŹäÿłæŰźæşŢäijŽåIJĺåijĂåğŃåőŽäźL’çśżåŠŇåőČçŽĎçĹűçśżçŽĎæŮűåĂŹèćńæL’ğè æĹŚäżňè£ŹéĞŇéĂŽè£Ğè£ŤåŻđäžĘäÿĂäÿłOrderedDictèĂŇäÿ■æŸŕäÿĂäÿłæŹőéĂŽçŽĎå■ŮåĚÿïijŇåŘŕäżěå

åęĆæđIJä¡ăæČşæđĎéĂăèĞłåůśçŽĎçśżå■ŮåĚÿåŕźèśąïijŇåŘŕäżěå¿Ĺåőźæ ŸŞçŽĎæL’l’åśŢè£ŹäÿłåŁ§èČ¡ă

```
from collections import OrderedDict

class NoDupOrderedDict(OrderedDict):
    def __init__(self, clsname):
        self.clsname = clsname
        super().__init__()
    def __setitem__(self, name, value):
        if name in self:
            raise TypeError('{} already defined in {}'.format(name,␣
↪self.clsname))
        super().__setitem__(name, value)

class OrderedMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        d = dict(clsdict)
        d['_order'] = [name for name in clsdict if name[0] != '_']
        return type.__new__(cls, clsname, bases, d)

    @classmethod
    def __prepare__(cls, clsname, bases):
        return NoDupOrderedDict(clsname)
```

äÿŃéÍćæĹŚäżňæţŃèŕŢéĞ■åď■çŽĎåőŽäźL’äijŽåĞžçŐřäżĂäźĹæČĚåĘţïijŽ

```
>>> class A(metaclass=OrderedMeta):
... def spam(self):
... pass
... def spam(self):
... pass
...
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "<stdin>", line 4, in A
    File "dupmethod2.py", line 25, in __setitem__
        (name, self.clsname))
TypeError: spam already defined in A
>>>
```

æIJĂåŘŐè£ŸæIJL'äÿĂçĆźå¿ĹéĞ■èęĄïijŇåřśæŸřåIJĺ `__new__()` æŰźæşŢäÿ■åŕźäžŐåĚČçśżäÿ■èćńä£őæŤźå■ŮåĚÿçŽĎåď'ĎçŘĘãĂĆ åř¡çőąçśżä¡£çŤĺäžĘåŔęåď'ŰäÿĂäÿłå■ŮåĚÿæİěåőŽäźL'ïijŇåIJĺæđĎéĂăæIJĂçźĹçŽĎ `class` åŕźèśąçŽĎæŮűåĂŹïijŇ æĹŚäžňäż■çĎűéIJĂèęĄåřĘè£Źäÿłå■ŮåĚÿè¡ňæ■ćäÿžäÿĂäÿłæ■čçąőçŽĎ `dict` åőđä¿ŃãĂĆ éĂŽè£Ğèŕ■åŘě `d = dict(clsdict)` æİěåőŇæĹŘè£ŹäÿłæŢĹæđIJãĂĆ

åŕźäžŐå¿Ĺåď'ŽåžŤçŤĺçÍŃåžŔèĂŇåůšïijŇèČ¡åď'§æ■ŢèŐůçśżåőŽäźL'çŽĎéąžåžŔæŸřäÿĂäÿłçIJŃäijijäÿ■ ä¿ŃåęĆïijŇåIJĺåŕźèśąåĚşçşżæŸăåřĎäÿ■ïijŇæĹŚäžňéĂŽåÿÿäijŽçIJŃåĹřäÿŃéÍćè£ŹçğæŰźåijŘåőŽäźL'çŽĎç

```
class Stock(Model):
    name = String()
    shares = Integer()
    price = Float()
```

åIJĺæąĘæđűåžŢåśĆïijŇæĹŚäžňå£Ěéążæ■ŢèŐůåőŽäźL'çŽĎéąžåžŔæİěåřĘåŕźèśąæŸăåřĎåĹřåĚČçżĎæĹÚ `as_csv()` çŽĎåŁ§èČ¡ïijL'ãĂĆ è£ŹèŁĆæijŤçď'žçŽĎæŁĂæIJŕéİđåÿÿçőĂå■ŢïijŇåźűäÿŤéĂŽåÿÿäijŽæŕŤåĚú

9.15 åőŽäźL'æIJL'åŘŕéĂL'åŘĆæŢřçŽĎåĚČçśż

éŮőécŸ

ä¡ăæČşåőŽäźL'äÿĂäÿłåĚČçśżïijŇåĚĄèőÿçśżåőŽäźL'æŮűæŘŘä¿ŻåŘŕéĂL'åŘĆæŢřïijŇè£ŹæăůåŘŕäžěæĆ

èğčåĘşæŰźæąĹ

åIJĺåőŽäźL'çśżçŽĎæŮűåĂŹïijŇPythonåĚĄèőÿæĹŚäžňä¡£çŤĺ "metaclass""åĚşéŤőå■ŮåŘĆæŢřæİěæŇĞåőŽçL'źåőŽçŽĎåĚČçśżãĂĆ ä¿ŃåęĆä¡£çŤĺæŁ¡èśąå§žçśżïijŽ

```
from abc import ABCMeta, abstractmethod
class IStream(metaclass=ABCMeta):
    @abstractmethod
```

```
    def read(self, maxsize=None):
        pass

    @abstractmethod
    def write(self, data):
        pass
```

çĎűèĂŇïijŇåIJÍèĞłåőŽäźL’åĚČçśżäÿ■æĹŚäžňè£ŸåŔŕäżěæŔŘä¿ŻåĚűäžŰçŽĎåĚşéŤőå■ŮåŔĆæŢřïijŇå

```
class Spam(metaclass=MyMeta, debug=True, synchronize=True):
    pass
```

äÿžäžĘä¡£åĚČçśżæŤŕæŇĄè£ŹäžŻåĚşéŤőå■ŮåŔĆæŢřïijŇä¡ăå£Ěéążçąőä£ÍåIJÍ
__prepare__() , __new__() åŠŇ __init__() æŰźæşŢäÿ■
éČ¡ä¡£çŤĺåijžåĹűåĚşéŤőå■ŮåŔĆæŢřãĂĆåŕśåČŔäÿŃéÍćè£ŹæăůïijŽ

```
class MyMeta(type):
    # Optional
    @classmethod
    def __prepare__(cls, name, bases, *, debug=False,␣
↪synchronize=False):
        # Custom processing
        pass
        return super().__prepare__(name, bases)

    # Required
    def __new__(cls, name, bases, ns, *, debug=False,␣
↪synchronize=False):
        # Custom processing
        pass
        return super().__new__(cls, name, bases, ns)

    # Required
    def __init__(self, name, bases, ns, *, debug=False,␣
↪synchronize=False):
        # Custom processing
        pass
        super().__init__(name, bases, ns)
```

èőÍèőž

çżŹäÿĂäÿłåĚČçśżæůżåŁăåŔŕéĂL’åĚşéŤőå■ŮåŔĆæŢřéIJĂèęĄä¡ăåőŇåĚĺåijĎæĞĆçśżåĹŻåžžçŽĎæL’Ăa
åŽääÿžè£ŹäžŻåŔĆæŢřäijŽèćńäijăéĂŠçżŹæŕŘäÿĂäÿłçŻÿåĚşçŽĎæŰźæşŢãĂĆ
__prepare__() æŰźæşŢåIJÍæL’ĂæIJL’çśżåőŽäźL’åijĂåğŇæL’ğèąŇåL’■éęŰåĚĹèćńèřČçŤĺïijŇçŤĺæİèåĹŻ
éĂŽåÿÿæİèèőšïijŇè£ŹäÿłæŰźæşŢåŔłæŸŕçőĂå■ŢçŽĎè£ŤåŽđäÿĂäÿłå■ŮåĚÿæĹŰåĚűäžŰæŸăářĎářźèśąãĂĆ
__new__() æŰźæşŢèćńçŤĺæİèåőđä¿ŃåŇŰæIJĂçżĹçŽĎçśżåřźèśąãĂĆåőČåIJÍçśżçŽĎäÿżä¡Şèćńæ L’ğèąŇåő
__init__() æŰźæşŢæIJĂåŘŐèćńèřČçŤĺïijŇçŤĺæİèæL’ğèąŇåĚűäžŰçŽĎäÿĂäžŽåĹİåğŇåŇŰåůěä¡IJãĂĆ

å¡ŞæĹŚäžňæđĎéĂäåĚČçśżçŽĎæŮűůåĂŹïijŇéĂŽåÿÿåŔłéIJĂèęĄåőŽäźL’äÿĂäÿł

__new__() æĹŰ __init__() æŰźæşŢïijŇä¡Ęäÿ■æŸŕäÿď'äÿłéČ¡åőŽäźĽ'ãĂĆ
ä¡ĘæŸŕïijŇåęĆæđIJéIJĂèęĄæŐěåŔŮåĚűäžŰçŽĎåĚşéŤőå■ŮåŔĆæŢřçŽĎèŕİïijŇè£Źäÿď'äÿłæŰźæşŢåřśèęĄå
éžŸèőď'çŽĎ __prepare__() æŰźæşŢæŐěåŔŮäżżæĎŔçŽĎåĚşéŤőå■ŮåŔĆæŢřïijŇä¡ĘæŸŕäijŽå£¡çŢěåő
æĽ'Ăäżěå Ŕłæ IJL'å¡Şè£ŹäžŽécİåď'ŰçŽĎåŔĆæŢřåŔŕèČ¡äijŽå¡śåŞ■åĽřçśżåŚ¡åŘ■çľ'žéŮť'çŽĎåĹŽäżżæŮűä¡ăa
__prepare__() æŰźæşŢãĂĆ

éĂŽè£Ğä¡£çŤĺåijżåĹűåĚşéŤőå■ŮåŔĆæŢřïijŇåIJÍçśżçŽĎåĹŽäżżè£ĞçÍŇäÿ■æĹŚäžňå£ĚéąžéĂŽè£ĞåĚş

ä¡£çŤĺåĚşéŤőå■ŮåŔĆæŢřéĚ■ç¡őäÿĂäÿłåĚČçśżè£ŸåŕŕäžěèğĘä¡IJåŕźçśżåŔŸéĞŔçŽĎäÿĂçğ■æŻ£äžčæU

```
class Spam(metaclass=MyMeta):
    debug = True
    synchronize = True
    pass
```

åřĘè£ŹäžŽåśđæĂğåőŽäźĽäÿžåŔĆæŢřçŽĎåě¡åď'ĎåIJÍäžŐåőČäžňäÿ■äijŽæśąæ§Şçśżç ŽĎåŘ■çğřçľ'žéŮt'
è£ŹäžŽåśđæĂğäžĚäžĚåŔłäžŐåśđäžŐçśżçŽĎåĹŽäżżéŸűæőţïijŇèĂŇäÿ■æŸŕçśżäÿ■çŽĎèŕ■åŘěæĽ'ğèąŇéŸűa
åŔęåď'ŰïijŇåőČäžňåIJÍ __prepare__() æŰźæşŢäÿ■æŸŕåŕŕäżěèćńèő£éŰőçŽĎïijŇåŽääÿžè£ŹäÿłæŰźæşŢ
ä¡ĘæŸŕçśżåŔŸéĞŔåŔłèČ¡åIJÍåĚČçśżçŽĎ __new__() åŠŇ __init__()
æŰźæşŢäÿ■åŕŕèğĄãĂĆ

9.16 *argsåŠŇ**kwargsçŽĎåijżåĹűåŔĆæŢřç■¿åŘ■

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłåĞ¡æŢřæĹŰæŰźæşŢïijŇåőČä¡£çŤĺ*argsåŠŇ**kwargsä¡IJäÿžåŔĆæŢřïijŇè£Źæăůä¡£å¿Ů
ä¡ĘæIJL'æŮűåĂŹä¡ăæČşæčĂæ§ëäijăéĂŠè£ŽæİěçŽĎåŔĆæŢræŸŕäÿ■æŸŕæ§ŘäÿłäžăæČşèęĄçŽĎçśżåđŇãĂĆ

èğčåĘşæŰźæąĹ

åŕźäżżä¡ŢæűL'åŔŁåĹřæŞ■ä¡IJåĞ¡æŢřèřČçŤĺç■¿åŘ■çŽĎéŮőécŸïijŇä¡äéČ¡åžŤèŕëä¡£çŤĺ
inspect æĺąåİŮäÿ■çŽĎç■¿åŘ■çĽźæĂğãĂĆ æĹŚäżňæIJĂäÿżèęĄåĚşæşĺäÿď'äÿłçśżïijŽSignature
åŠŇ Parameter ãĂĆäÿŇéİćæŸŕäÿĂäÿłåĹŽäżżåĞ¡æŢřåL'■éİçŽĎäžď'äžŠä¿Ňå■ŘïijŽ

```
>>> from inspect import Signature, Parameter
>>> # Make a signature for a func(x, y=42, *, z=None)
>>> parms = [ Parameter('x', Parameter.POSITIONAL_OR_KEYWORD),
...           Parameter('y', Parameter.POSITIONAL_OR_KEYWORD,␣
 ↪default=42),
...           Parameter('z', Parameter.KEYWORD_ONLY, default=None) ]
>>> sig = Signature(parms)
>>> print(sig)
(x, y=42, *, z=None)
>>>
```

äÿĂæŮęä¡ăæIJL'äžĘäÿĂäÿłç■¿åŘ■åŕźèśąïijŇä¡ăåřšåŔŕäžěä¡£çŤĺåőČçŽĎ bind()
æŰźæşŢå¿Ĺåőźæ ŸŞçŽĎåřĘåőČçżŚåőŽåĹř *args åŠŇ **kwargs äÿŁåŐżãĂĆ
äÿŇéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎæijŤçd'žïijŽ

```
>>> def func(*args, **kwargs):
...     bound_values = sig.bind(*args, **kwargs)
...     for name, value in bound_values.arguments.items():
...         print(name,value)
...
>>> # Try various examples
>>> func(1, 2, z=3)
x 1
y 2
z 3
>>> func(1)
x 1
>>> func(1, z=3)
x 1
z 3
>>> func(y=2, x=1)
x 1
y 2
>>> func(1, 2, 3, 4)
Traceback (most recent call last):
...
    File "/usr/local/lib/python3.3/inspect.py", line 1972, in _bind
        raise TypeError('too many positional arguments')
TypeError: too many positional arguments
>>> func(y=2)
Traceback (most recent call last):
...
    File "/usr/local/lib/python3.3/inspect.py", line 1961, in _bind
        raise TypeError(msg) from None
TypeError: 'x' parameter lacking default value
>>> func(1, y=2, x=3)
Traceback (most recent call last):
...
    File "/usr/local/lib/python3.3/inspect.py", line 1985, in _bind
        '{arg!r}'.format(arg=param.name))
TypeError: multiple values for argument 'x'
>>>
```

åŘŕäžěçIJŃåĞžæİěïijŇéĂŽè£ĞåřEç■¿åŘ■åŠŇäijăéĂŠçŽĎåŔĆæŢřçżŠåőŽèţůæİěïijŇåŔŕäžěåijžåĹűåĞ¡ä

äÿŃéİćæŸŕäÿĂäÿłåijžåĹűåĞ¡æŢřç■¿åŘ■æŻťåĚůä¡ŞçŽĎä¿Ńå■ŘãĂĆåIJĺäžčçăĄäÿ■ïijŇæĹŚäžňåIJĺå§žç
`__init__()` æŰźæşŢïijŇ çĎűåŘŐæĹŚäžňåijžåĹűæĽĂæIJL'çŽĎå■Řçśżå£Ěéążæŕŕä¿ŻäÿĂäÿłçL'źåőŽçŽĹ

```
from inspect import Signature, Parameter

def make_sig(*names):
    parms = [Parameter(name, Parameter.POSITIONAL_OR_KEYWORD)
             for name in names]
    return Signature(parms)

class Structure:
```

```
    __signature__ = make_sig()
    def __init__(self, *args, **kwargs):
        bound_values = self.__signature__.bind(*args, **kwargs)
        for name, value in bound_values.arguments.items():
            setattr(self, name, value)

# Example use
class Stock(Structure):
    __signature__ = make_sig('name', 'shares', 'price')

class Point(Structure):
    __signature__ = make_sig('x', 'y')
```

äÿŃéÌćæŸŕä¡£çŤÍè£ŹäÿŁ `Stock` çśžçŽĎçd'žä¿ŃïijŽ

```
>>> import inspect
>>> print(inspect.signature(Stock))
(name, shares, price)
>>> s1 = Stock('ACME', 100, 490.1)
>>> s2 = Stock('ACME', 100)
Traceback (most recent call last):
...
TypeError: 'price' parameter lacking default value
>>> s3 = Stock('ACME', 100, 490.1, shares=50)
Traceback (most recent call last):
...
TypeError: multiple values for argument 'shares'
>>>
```

èőÍèőž

åIJÍæĹŚäžňéIJĂèęĄæđĎåžžéĂŽçŤÍåĞ¡æŢřåžŞãĂĄçijŰåĘŹèčĚéěřåŹÍæĹŰåőđçŐřäžčçŘĘçŽĎæŮűåĂŹ
`*args` åŠŇ `**kwargs` çŽĎä¡£çŤÍæŸŕå¿ĹæŹőéĄ■çŽĎãĂĆ
ä¡ĘæŸŕïijŇè£ŹæăůçŽĎåĞ¡æŢřæIJL'äÿĂäÿłçijžçĆźåřśæŸŕå¡Şä¡ăæČşèęĄåőđçŐřèĞłåůśçŽĎåŔĆæŢřæčĂéłŇ
è£ŹæŮűåĂŹæĹŚäžňåŔŕäžěéĂŽè£ĞäÿĂäÿłç■¿åŘ■åŕźèśąæİěçőĂåŇŰåőČãĂĆ

åIJÍæIJĂåŘŐçŽĎäÿĂäÿłæŰźæąĹåőđä¿Ńäÿ■ïijŇæĹŚäžňè£ŸåŔŕäžěéĂŽè£Ğä¡£çŤÍèĞłåőŽäźL'åĚČçśžæİ

```
from inspect import Signature, Parameter

def make_sig(*names):
    parms = [Parameter(name, Parameter.POSITIONAL_OR_KEYWORD)
             for name in names]
    return Signature(parms)

class StructureMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        clsdict['__signature__'] = make_sig(*clsdict.get('_fields',
↪[]))
```

```
        return super().__new__(cls, clsname, bases, clsdict)

class Structure(metaclass=StructureMeta):
    _fields = []
    def __init__(self, *args, **kwargs):
        bound_values = self.__signature__.bind(*args, **kwargs)
        for name, value in bound_values.arguments.items():
            setattr(self, name, value)

# Example
class Stock(Structure):
    _fields = ['name', 'shares', 'price']

class Point(Structure):
    _fields = ['x', 'y']
```

å¡ŞæĹŚäžňèĞłåőŽäźL'ç■¿åŘ■çŽĎæŮűåĂŹïijŇářĘç■¿åŘ■å■ŸåĆĺåIJĺçL'źåőŽçŽĎåšđæĂğ `__signature__` äÿ■éĂŽåÿÿæŸŕå¿ĹæIJL'çŤĺçŽĎãĂĆ è£ŹæăůçŽĎèŕİïijŇåIJĺä¡£çŤĺ `inspect` æĺąåİŮæL'ğèąŇåĘĚçIJAçŽĎäžčçăĄåŕśèČ¡åŔŚçŐřç■¿åŘ■åźűåřĘåőČä¡IJäÿžèřČçŤĺçžęåőŽãĂĆ

```
>>> import inspect
>>> print(inspect.signature(Stock))
(name, shares, price)
>>> print(inspect.signature(Point))
(x, y)
>>>
```

9.17 åIJĺçśżäÿŁåijžåĹűä¡£çŤĺçijŰçĺŃèğĎçžę

éŮőécŸ

ä¡ăçŽĎçĺŃåžŔåŇĚåŘńäÿĂäÿłå¿Ĺåď'ğçŽĎçśżçżğæL'£ä¡ŞçşżïijŇä¡åäÿŇæIJŻåijžåĹűæL'ğèąŇæ§ŘäžŻçij

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăæČşçŻŚæŐğçśżçŽĎåőŽäźL'ïijŇéĂŽåÿÿåŔŕäżěéĂŽè£ĞåőŽäźL'äÿĂäÿłåĚČçśżãĂĆäÿĂäÿłå§ `type` åźűéĞ■åőŽäźL'åőČçŽĎ `__new__()` æŰźæşŢ æĹŰèĂĚæŸŕ `__init__()` æŰźæşŢãĂĆæŕŤåęĆïijŽ

```
class MyMeta(type):
    def __new__(self, clsname, bases, clsdict):
        # clsname is name of class being defined
        # bases is tuple of base classes
        # clsdict is class dictionary
        return super().__new__(cls, clsname, bases, clsdict)
```

åŘęäÿĂçğ■æŸŕïijŇåőŽäźL' `__init__()` æŰźæşŢïijŽ

```
class MyMeta(type):
    def __init__(self, clsname, bases, clsdict):
        super().__init__(clsname, bases, clsdict)
        # clsname is name of class being defined
        # bases is tuple of base classes
        # clsdict is class dictionary
```

äÿžäžĘä¡£çŤĺè£ŹäÿłåĚČçśżïijŇä¡äéĂŽåÿÿèęĄåřĘåőČæŤ¿åĹřåĹřäÿĂäÿłéąűçžğçĹűçśżåőŽäźL’äÿ■ïijŇçĴ

```
class Root(metaclass=MyMeta):
    pass

class A(Root):
    pass

class B(Root):
    pass
```

åĚČçśżçŽĎäÿĂäÿłåĚşéŤőçL’źçĆźæŸŕåőČåĚĄèőÿä¡ääIJĺåőŽäźL’çŽĎæŮűåĂŹæčĂæ§ěçśżçŽĎåĘĚåőźã
__init__() æŰźæşŢäÿ■ïijŇ ä¡ääŔŕäżěå¿Ĺè¡żæİ¿çŽĎæčĂæ§ěçśżå■ŮåĚÿãĂĄçĹűçśżç■L’ç■L’ãĂĆåźűäÿŤï
åŽäæ■d’ïijŇäÿĂäÿłæąĘæđűçŽĎæđĎåżžèĂĚåřśèČ¡åIJĺåd’ğåđŇçŽĎçżğæL’£ä¡Şçşżäÿ■éĂŽè£ĞçźŹäÿĂäÿłéąű

ä¡IJäÿžäÿĂäÿłåĚůä¡ŞçŽĎåžŤçŤĺä¿Ńå■ŘïijŇäÿŃéİćåőŽäźL’äžĘäÿĂäÿłåĚČçśżïijŇåőČäijŽæŃŠçżİäżżä¡

```
class NoMixedCaseMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        for name in clsdict:
            if name.lower() != name:
                raise TypeError('Bad attribute name: ' + name)
        return super().__new__(cls, clsname, bases, clsdict)

class Root(metaclass=NoMixedCaseMeta):
    pass

class A(Root):
    def foo_bar(self): # Ok
        pass

class B(Root):
    def fooBar(self): # TypeError
        pass
```

ä¡IJäÿžæŻť énŸçžğåŠŇåőđçŤĺçŽĎä¿Ńå■ŘïijŇäÿŃéİćæIJL’äÿĂäÿłåĚČçśżïijŇåőČçŤĺæİěæčĂæţŃéĞ■è¡¡

```
from inspect import signature
import logging

class MatchSignaturesMeta(type):

    def __init__(self, clsname, bases, clsdict):
        super().__init__(clsname, bases, clsdict)
```

```
        sup = super(self, self)
        for name, value in clsdict.items():
            if name.startswith('_') or not callable(value):
                continue
            # Get the previous definition (if any) and compare the␣
↪signatures
            prev_dfn = getattr(sup,name,None)
            if prev_dfn:
                prev_sig = signature(prev_dfn)
                val_sig = signature(value)
                if prev_sig != val_sig:
                    logging.warning('Signature mismatch in %s. %s !
↪= %s',
                                    value.__qualname__, prev_sig,␣
↪val_sig)

# Example
class Root(metaclass=MatchSignaturesMeta):
    pass

class A(Root):
    def foo(self, x, y):
        pass

    def spam(self, x, *, z):
        pass

# Class with redefined methods, but slightly different signatures
class B(A):
    def foo(self, a, b):
        pass

    def spam(self,x,z):
        pass
```

åęĆæđIJä¡ãè£Řèąǹè£Źæőţäžčçă Ąïij Ňåřšäij Žå¿ŮåĹřäÿŃéİćè£Źæăůç ŽĎè¿ŞåĞžçżŞæđIJïij Ž

```
WARNING:root:Signature mismatch in B.spam. (self, x, *, z) != (self,
↪ x, z)
WARNING:root:Signature mismatch in B.foo. (self, x, y) != (self, a,␣
↪b)
```

è£Źçğ■è■ęåŚŁä£ąæĄŕåŕźäžŐæ■ŢèŐůäÿĂäžŻå¿őåęŹçŽĎçÍŃåžŔbugæŸŕå¿ĹæIJL'çŤÍçŽĎãĂĆä¿ŃåęĆïij éĆčäźĹå¡Şå■ŘçśżæŤźåŔŸåŔĆæŢřåŘ■å■ŮçŽĎæŮűåĂŹåršäijŽèřČçŤÍåĞžéŤŹãĂĆ

èőÍèőž

åIJÍåď ğåđŃéİćåŘŚåŕźèśąçŽĎçÍŃåžŔäÿ■ïijŇéĂŽåÿÿåřĘçśżçŽĎåőŽäźL'æŤ¿åIJÍåĚČçśżäÿ■æŐğåĹűæŸŕå åĚČçśżåŔŕäžěçŻŚæŐğçśżçŽĎåőŽäźL'ïijŇè■ęåŚŁçijŰçÍŃäžžåŚŸæ§ŘäžŻæšąæIJL'æşÍæĎŘåĹřçŽĎåŔŕèČ¡åĞ

æIJL'äžžåŘŕèČ¡äijŽèŕť'ïijŇåČŔè£Źæăůç ŽĎéŤŹèŕŕåŘŕäžěéĂŽè£ĞçÍŃåžŘåĹĘæđŘåůěåĚůæĹŰIDEåŐżå
ä¡ĘæŸŕïijŇåęĆæđIJä¡ăåIJÍæđĎåžžäÿĂäÿłæąĘæđűæĹŰåĞ¡æŢřåžŞä¿ŻåĚűäžŰäžžä¡£çŤÍïijŇéĆčäźĹä¡ăæšąåŁ
åŻăæ■ď'ïijŇåŕźäžŐè£Źçğ■çśžåđŃçŽĎçÍŃåžŔïijŇåęĆæđIJåŘŕäžěåIJĺåĚČçśžäÿ■åĄŽæčĂæţŃæĹŰèőÿåŘŕäžěă

åIJÍåĚČçśžäÿ■éĂL'æŃl'éĞ■æŰřåőŽäźL' __new__() æŰźæşŢè£ŸæŸŕ __init__() æŰźæşŢåŘŰåĘşäžŐä¡ăæČşæĂŐæăůä¡£çŤÍçżŞæđIJçśžãĂĆ __new__() æŰźæşŢåIJÍçśžåĹŻåžžäźŃåL'■èćńèřČçŤÍïijŇéĂŽåÿÿçŤÍäžŐéĂŽè£ĞæŞçğ■æŰźåijŔïijĹæŕŤåęĆéĂŽè£ĞæŤ èĂŇ __init__() æŰźæşŢæŸŕåIJÍçśžèćńåĹŻåžžäźŃåŘŐèćńèřČçŤÍïijŇå¡Şä¡ăéIJĂèęĄåőŇæŢt'æđĎåžžçśžä åIJÍæIJĂåŘŐäÿĂäÿłä¿Ńå■Řäÿ■ïijŇè£ŹæŸŕå£ĚèęĄçŽĎïijŇåŻăäÿžåőČä¡£çŤÍäžĘ super() åĞ¡æŢřæİěæŘIJçt'ćäźŃåL'■çŽĎåőŽäźL'ãĂĆ åőČåŔłèČ¡åIJÍçśžçŽĎåőđä¿Ńèćńå ĹŻåžžäźŃåŘŐïijŇåźűäÿŤçŻ

æIJĂåŘŐäÿĂäÿłä¿Ńå■Řè£ŸæijŤçď'žäžĘPythonçŽĎåĞ¡æŢřç■¿åŘ■åŕźèśąçŽĎä¡£çŤÍãĂĆ åőđéŹĚäÿŁïijŇåĚČçśžäijŽçőąçŘĘäÿ■æŕŔäÿłäÿĂäÿłèřČçŤÍåőŽäźL'ïijŇæŘIJçt'ćåL'■äÿĂäÿłåőŽäźL'ïijĹåęĆæđ çĎűåŘŐéĂŽè£Ğä¡£çŤÍ inspect.signature() æİěçőĂå■ŢçŽĎæŕŤè¿ČåőČäžňçŽĎèřČçŤÍç■¿åŘ■ãĂĆ

æIJĂåŘŐäÿĂçĆźïijŇäžčçăĄäÿ■æIJL'äÿĂèąŇä¡£çŤÍäžĘ super(self, self) åźűäÿ■æŸŕæŐŠçL'ĹéŤŹèŕŕãĂĆ å¡Şä¡£çŤÍåĚČçśžçŽĎæŮűåĂŹïijŇæĹŚäžňèęĄæŮűåĹžèőŕä¡ŔäÿĂçĆźåŕśæŸŕ self åőđéŹĚäÿŁæŸŕäÿĂäÿłçśžåŕźèśąãĂĆ åŻăæ■ď'ïijŇè£Źæİąèŕ■åŘěåĚűåőđåřśæŸŕçŤÍæİěåŕżæL'¿ä¡■äžŐçż self çĹűçśžçŽĎåőŽäźL'ãĂĆ

9.18 äžěçijŰçÍŃæŰźåijŔåőŽäźL'çśž

éŮőécŸ

ä¡ăåIJÍåĘŹäÿĂæőţäžčçăĄïijŇæIJĂçżĹéIJĂèęĄåĹŻåžžäÿĂäÿłæŰřçŽĎçśžåŕźèśąãĂĆä¡äèĂČèŹŚåřĘçśžçŻ åźűäÿŤä¡£çŤÍåĞ¡æŢřæŕŤåęĆ exec() æİěæL'ğèąŇåőČïijŇä¡ĘæŸŕä¡äæČşåŕżæL'¿äÿĂäÿłæŻt'åŁääijŸéŻĚçŽ

èğčåĘşæŰźæąĹ

ä¡ăåŘŕäžěä¡£çŤÍåĞ¡æŢř types.new_class() æİěåĹİåğŃåŇŰæŰřçŽĎçśžåŕźèśąãĂĆ ä¡äéIJĂèęĄåĄŽçŽĎåŘłæŸŕæŘŘä¿ŻçśžçŽĎåŘ■å■ŮãĂĄçĹűçśžåĚČçżĎãĂĄåĚşéŤőå■ŮåŘĆæŢřïijŇäžěåŘŁ

```
# stock.py
# Example of making a class manually from parts

# Methods
def __init__(self, name, shares, price):
    self.name = name
    self.shares = shares
    self.price = price
def cost(self):
    return self.shares * self.price

cls_dict = {
    '__init__' : __init__,
    'cost' : cost,
}
```

```
# Make a class
import types

Stock = types.new_class('Stock', (), {}, lambda ns: ns.update(cls_
↪dict))
Stock.__module__ = __name__
```

è£Źçğ■æŰźåijŔäijŽæđĎåżžäÿĂäÿłæŹőéĂŽçŽĎçśžåŕźèśąïijŇåźűäÿŤæŇĽçĚğä¡ăçŽĎæIJ§æIJŻåůěä¡IJïj

```
>>> s = Stock('ACME', 50, 91.1)
>>> s
<stock.Stock object at 0x1006a9b10>
>>> s.cost()
4555.0
>>>
```

è£Źçğ■æŰźæşŢäÿ■ïijŇäÿĂäÿłæŕŤè¿ČéŽ¿çŘĘèğčçŽĎåIJřæŰźæŸŕåIJĺèřČçŤĺåőŇ
types.new_class() åŕź Stock.__module__ çŽĎèţŇåĂijãĂĆ
æŕŔæňąå¡ŞäÿĂäÿłçśžèćńåőŽäźĽåŘŐïijŇåőČçŽĎ __module__
åśđæĂğåŇĚåŘńåőŽäźĽåőČçŽĎæĺąåİŮåŘ■ãĂĆ è£ŹäÿłåŘ■å■ŮçŤĺäžŐçŤ§æĹŘ
__repr__() æŰźæşŢçŽĎè¿ŞåĞžãĂĆåőČåŘŇæăůäź§ècńçŤĺäžŐå¿Ĺåď˙ŽåžŞïijŇæŕŤåęĆ
pickle ãĂĆ åŻăæ■ď˙ïijŇäÿžäžĘèőľä¡ăåĹŽåżžçŽĎçśžæŸŕâĂIJæ■čçąőâĂİçŽĎïijŇä¡ăéIJĂèęĄçąőä£Ìè£ŹäÿłŽ

åęĆæđIJä¡ăæČşåĹŽåżžçŽĎçśžéIJĂèęĄäÿĂäÿłäÿ■åŘŇçŽĎåĚČçśžïijŇåŕäžěéĂŽè£Ğ
types.new_class() çňňäÿĽäÿłåŔĆæŢřäijăéĂŠçżŹåőČãĂĆä¿ŃåęĆïijŽ

```
>>> import abc
>>> Stock = types.new_class('Stock', (), {'metaclass': abc.ABCMeta},
...                         lambda ns: ns.update(cls_dict))
...
>>> Stock.__module__ = __name__
>>> Stock
<class '__main__.Stock'>
>>> type(Stock)
<class 'abc.ABCMeta'>
>>>
```

çňňäÿĽäÿłåŔĆæŢřè£ŸåŕäžěåŇĚåŘńåĚűäžŰçŽĎåĚşéŤőå■ŮåŔĆæŢřãĂĆæŕŤåęĆïijŇäÿĂäÿłçśžçŽĎåő

```
class Spam(Base, debug=True, typecheck=False):
    pass
```

éĆčäźĹåŕäžěåřĘåĚűç£žèŕŚæĹŘåęĆäÿŃçŽĎ new_class() èřČçŤĺå¡ćåijŕïijŽ

```
Spam = types.new_class('Spam', (Base,),
                       {'debug': True, 'typecheck': False},
                       lambda ns: ns.update(cls_dict))
```

new_class() çňňåŻŽäÿłåŔĆæŢřæIJĂçěđçğŸïijŇåőČæŸŕäÿĂäÿłçŤĺæİěæŐěåŘŮçśžåŚ¡åŘ■çľžéŮť˙çŽ
éĂŽåÿÿè£ŹæŸŕäÿĂäÿłæŹőéĂŽçŽĎå■ŮåĚÿïijŇä¡ĘæŸŕåőČåőđéŹĚäÿŁæŸŕ
__prepare__() æŰźæşŢè£ŤåŻđçŽĎäżžæĎŘåŕźèśąïijŇè£ŹäÿłåIJĺ9.14åřŔèŁĆåůšçżŘäžŃçż■è£ĞäžĘąĂĆ

è£Źäÿłå Ğ¡æŢřéIJĂèęĄä¡£çŤĺäÿŁéİćæijŤçd'žçŽĎ update()
æŰźæşŢçżŹåŚ¡åŘ■çl'žéŮt'åćđåŁååĘĚåőźãĂĆ

èőĺèőž

å¿Ĺåd'ŽæŮűåĂŹåęĆæđIJèČ¡æđĎéĂăæŰřçŽĎçśżårźèśąæŸŕå¿ĹæIJL'çŤĺçŽĎãĂĆ
æIJL'äÿłå¿ĹçĘ§æĆL'çŽĎä¿Ńå■ŘæŸŕèřČçŤĺ collections.namedtuple()
åĞ¡æŢřïijŇä¿ŃåęĆïijŽ

```
>>> Stock = collections.namedtuple('Stock', ['name', 'shares',
↪'price'])
>>> Stock
<class '__main__.Stock'>
>>>
```

namedtuple() ä¡£çŤĺ exec() èĂŇäÿ■æŸŕäÿŁéİćäżŃçż■çŽĎæŁĂæIJŕãĂĆä¡ĘæŸŕïijŇäÿŃéİćéĂŽè
æĹŚäžňçŻt'æŐěåĹŽåžžäÿĂäÿłçśżïijŽ

```
import operator
import types
import sys

def named_tuple(classname, fieldnames):
    # Populate a dictionary of field property accessors
    cls_dict = { name: property(operator.itemgetter(n))
                 for n, name in enumerate(fieldnames) }

    # Make a __new__ function and add to the class dict
    def __new__(cls, *args):
        if len(args) != len(fieldnames):
            raise TypeError('Expected {} arguments'.
↪format(len(fieldnames)))
        return tuple.__new__(cls, args)

    cls_dict['__new__'] = __new__

    # Make the class
    cls = types.new_class(classname, (tuple,), {},
                          lambda ns: ns.update(cls_dict))

    # Set the module to that of the caller
    cls.__module__ = sys._getframe(1).f_globals['__name__']
    return cls
```

è£Źæőtäžčçă ĄçŽĎæIJĂåŘŐéČĺåĹĘä¡£çŤĺäžĘäÿĂäÿłæL'ĂèřŞçŽĎ"æąĘæđűé■ŤæşŢ"ïijŇéĂŽè£ĞèřČçŤ
sys._getframe() æİěèŐůåŔŰèřČçŤĺèĂĚçŽĎæĺąåİŮůåŘ■ãĂĆ
åŔęåd'ŰäÿĂäÿłæąĘæđűé■ŤæşŢä¿Ńå■ŘåIJĺ2.15åřŔèŁĆäÿ■æIJL'äžŃçż■è£ĞãĂĆ

äÿŇéİćçŽĎä¿Ńå■ŘæijŤçd'žäžĘąL'■éİćçŽĎäžčçăĄæŸŕåęĆä¡Ţåůěä¡IJçŽĎïijŽ

```
>>> Point = named_tuple('Point', ['x', 'y'])
>>> Point
<class '__main__.Point'>
>>> p = Point(4, 5)
>>> len(p)
2
>>> p.x
4
>>> p.y
5
>>> p.x = 2
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>> print('%s %s' % p)
4 5
>>>
```

è£ŹéąźæĂæIJŕäÿĂäÿłå¿ĹéĞ■èęĄçŽĎæŰźéİćæŸŕåŐČåŕźäžŐåĚČçśżçŽĎæ■čçąőä¡£çŤĺãĂĆ ä¡ăåŔŕèČ¡åČŔéĂŽè£ĞçŻť'æŐěåőđä¿ŃåŇŰäÿĂäÿłåĚČçśżæİěçŻť'æŐěåŻåžžäÿĂäÿłçśżïijŽ

```
Stock = type('Stock', (), cls_dict)
```

è£Źçğ■æŰźæşŢçŽĎéŮőécŸåIJĺäžŐåőČå£¡çŢëäžĘäÿĂäžŻåĚşéŤőæ■ěéłd'ïijŇæŕŤåęĆåŕźäžŐåĚČçśżäÿ■ `__prepare__()` æŰźæşŢçŽĎèŕČçŤĺãĂĆ éĂŽè£Ğä¡£çŤĺ `types.new_class()` ïijŇä¡ăåŔŕäžëä£ÌèŕĄæL'ĂæIJL'çŽĎå£ĚèęĄåĹİåğŃåŇŰæ■ěéłd'éČ¡èČ¡å¿ŮåĹřæL'ğèąŇãĂĆ æŕŤåęĆïijŇ`types.new_class()` çňňåŻŻäÿłåŔĆæŢřçŽĎåŻđèŕČåĞ¡æŢřæŐěåŔŮ `__prepare__()` æŰźæşŢè£ŤåŻđçŽĎæŸăărĎåŕźèśąãĂĆ

åęĆæđIJä¡ăäžĚäžĚåŔŁæŸŕæČşæL'ğèąŇåĞĘåd'Ğæ■ěéłd'ïijŇåŔŕäžëä¡£çŤĺ `types.prepare_class()` ãĂĆä¿ŃåęĆïijŽ

```
import types
metaclass, kwargs, ns = types.prepare_class('Stock', (), {'metaclass
↪': type})
```

åőČäijŽæ§ěæL'¿åŘĹéĂĆçŽĎåĚČçśżåźűèŕČçŤĺåőČçŽĎ `__prepare__()` æŰźæşŢãĂĆ çĎűåŘŐè£ŹäÿłåĚČçśżä£Ìå■ŸåőČçŽĎåĚşéŤőå■ŮåŔĆæŢřijŇåĞĘåd'ĞåŚ¡åŘ■çl'žéŮt'åŘŐèćńè

æŻt'åd'Žä£qæĄf, èŕůåŔĆèĂČ PEP 3115 , äžěåŘŁ Python documentation .

9.19 åIJĺåőŽäźL'çŽĎæŮűåĂŹåĹİåğŃåŇŰçśżçŽĎæĹŘåŚŸ

éŮőécŸ

ä¡ăæČşåIJĺçśżèćńåőŽäźL'çŽĎæŮűåĂŹåŕśåĹİåğŃåŇŰäÿĂéČĺåĹĘçśżçŽĎæĹŘåŚŸïijŇèĂŇäÿ■æŸŕèęĄç

èğčåĘşæŰźæąĹ

åIJÍçśžåőŽäźĽ'æŮűářśæĽ'ğèąŇåĹİåğŃåŇŰæĹŰèő¿ç¡őæŞ■ä¡IJæŸŕåĚČçśżçŽĎäÿĂäÿłåĚÿåđŃåžŤçŤÍåIJ
è£ŹæŮűåĂŹä¡ääŔŕäżěæĽ'ğèąŇäÿĂäžŽéćİåď'ŰçŽĎæŞ■ä¡IJãĂĆ

äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijŇåĹl'çŤÍè£ŹäÿłæĂİèůŕæİěåĹŻåžžçśżäijijäžŐ
`collections` æÍąåİŮäÿ■çŽĎåŚ¡åŘ■åĚČçżĎçŽĎçśżïijŽ

```
import operator

class StructTupleMeta(type):
    def __init__(cls, *args, **kwargs):
        super().__init__(*args, **kwargs)
        for n, name in enumerate(cls._fields):
            setattr(cls, name, property(operator.itemgetter(n)))

class StructTuple(tuple, metaclass=StructTupleMeta):
    _fields = []
    def __new__(cls, *args):
        if len(args) != len(cls._fields):
            raise ValueError('{} arguments required'.format(len(cls.
↪_fields)))
        return super().__new__(cls,args)
```

è£Źæőţäžčçă ĄåŕŕäžěçŤÍæİěåőŽäźĽ'çőĂå■ŢçŽĎå§žäžŐåĚČçżĎçŽĎæŢřæ■őçżŞæđĎïijŇąęĆäÿŃæĽ'Ăç

```
class Stock(StructTuple):
    _fields = ['name', 'shares', 'price']

class Point(StructTuple):
    _fields = ['x', 'y']
```

äÿŃéİćæijŤçď'žåőČåęĆä¡Ţåůěä¡IJïijŽ

```
>>> s = Stock('ACME', 50, 91.1)
>>> s
('ACME', 50, 91.1)
>>> s[0]
'ACME'
>>> s.name
'ACME'
>>> s.shares * s.price
4555.0
>>> s.shares = 23
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>>
```

èőÍèőž

è£ŹäÿĂåřŔèŁĆäÿ■ïijŇçśż `StructTupleMeta` èŐůåŔŰåĹřçśżåśđæĂğ `_fields` äÿ■çŽĎåśđæĂğåŘ■å■ŮåĹŮèąĺïijŇ çĎűåŘŐåřĘåőĆäžňè¡ňæ■ćæĹŘçŻÿåžŤçŽĎåŔŕèő£éŮőçĽźåőŽåĚČçżĎæ `operator.itemgetter()` åĹŻåżžäÿĂäÿłèő£éŮőåŹĺåĞ¡æŢřïijŇ çĎűåŘŐ `property()` åĞ¡æŢřåřĘåĚűè¡ňæ■ćæĹŘäÿĂäÿłåśđæĂğãĂĆ

æIJňèŁĆæIJĂéŽ¿æĞĆçŽĎéČĺåĹĘæŸŕç§ěéĄŞäÿ■åŘŇçŽĎåĹİåğŃåŇŰæ■ěéłď'æŸŕäżĂäźĹæŮűåĂŹåŔŞ `StructTupleMeta` äÿ■çŽĎ `__init__()` æŰźæşŢåŔłåIJĺæŕŔäÿłçśżèćńåőŽäźĽ'æŮűèćńèřČçŤĺäÿĂæňąãĂ `cls` åŔĆæŢřåřśæŸŕéĆčäÿłèćńåőŽäźĽ'çŽĎçśżãĂĆåőđéŹĚäÿŁïijŇäÿŁè£řäżčçăĄä¡£çŤĺäžĘ `_fields` çśżåŔŸéĞŔæİèä£İå■ŸæŰřçŽĎèćńåőŽäźĽ'çŽĎçśżïijŇ çĎűåŘŐçżŹåőČåĘ■æůżåŁääÿĂçĆźæŰřçŽĎäÿIJèě£ãĂĆ

`StructTuple` çśżä¡IJäÿžäÿĂäÿłæŹőéĂŽçŽĎå§žçśżïijŇä¿ŻåĚűäżŰä¡£çŤĺèĂĚæİěçżğæĽ'£ãĂĆ è£Źäÿłçśżäÿ■çŽĎ `__new__()` æŰźæşŢçŤĺæİèæđĎéĂäæŰřçŽĎåőđä¿ŃãĂĆ è£ŹéĞŇä¡£çŤĺ `__new__()` åźűäÿ■æŸŕå¿Ĺåÿÿèğ Ąïij Ňäÿżèę Ąæ Ÿŕå Žääÿžæ ĹŚäžňèę Ąä£őæŤźåĚČçżĎçŽĎèŕČçŤĺç■¿åŘ■ïijŇ ä¡£å¿ŮæĹŚäžňåŔŕäżèåČŔæŹőéĂŽçŽĎåőđä¿ŃèŕČçŤĺéĆčæăůåĹŻåżžåőđä¿ŃãĂĆåŕśåČŔäÿŃéİćè£Źæăůïijž

```
s = Stock('ACME', 50, 91.1) # OK
s = Stock(('ACME', 50, 91.1)) # Error
```

èů§ `__init__()` äÿ■åŘŇçŽĎæŸŕïijŇ `__new__()` æŰźæşŢåIJĺåőđä¿ŃèćńåĹŻåżžäźŃåĽ■èćńèğęåŔŚçŤśäžŐåĚČçżĎæŸŕäÿ■åŔŕä£őæŤźçŽĎïijŇæĽ'Ääžěäÿ ĂæŮęåőČäžňèćńåĹŻåżžäžĘåŕśäÿ■åŔŕèČ¡åŕźåőČåĄŽä `__init__()` äijŽåIJĺåőđä¿ŃåĹŻåżžçŽĎæIJĂåŘŐèćńèğęåŔŚïijŇ è£Źæăůç ŽĎèŕİæĹŚäžňåŕśåŔŕäžèåĄŽæĹŚäžňæČşåĄŽçŽĎäžĘãĂĆè£Źäź§æŸŕäÿžäžĂäźĹ `__new__()` æŰźæşŢåůšçżŔèćńåőŽäźĽ'äžĘãĂĆ

åř¡çőąæIJňèŁĆå¿Ĺç§■ïijŇè£ŸæŸŕéIJĂèęĄä¡äèČ¡äżŤçżĘçăŤèŕżïijŇæůśåĚěæĂİèĂČPythonçśżæŸŕåęĆä è£ŸæIJĽ'åŕśæŸŕåĚČçśżåŠŇçśżçŽĎåŘĎäÿłäÿ■åŘŇçŽĎæŰźæşŢçl'űçń§åIJĺäżĂäźĹæŮűåĂŹècńèřČçŤĺãĂĆ

PEP 422 æŔŘä¿ŽäžĘäÿĂäÿłèğčåĘşæIJňèŁĆéŮőécŸçŽĎåŔęåd'ŰäÿĂçğ■æŰźæşŢãĂĆ ä¡ĘæŸŕïijŇæĹłæ■ćåĽřæĹŚåĘŹè£ŹæIJñäźęçŽĎæŮűåĂŹïijŇåőČè£ŸæšąèćńéĞĞçžşåŠŇæŐěåŔŮãĂĆ åř¡çőąåęĆæ■d'ïijŇåęĆædIJä¡ää¡£çŤĺçŽĎæŸŕPython 3.3æĹŰæŽť énŸçŽĎçĽĹæIJñïijŇéĆčäźĹè£ŸæŸŕåĂijå

9.20 åĹľçŤĺåĞ¡æŢřæşÍèğčåőđçŐřæŰźæşŢéĞ■è¡¡

éŮőécŸ

ä¡ăåůšçżŔå■ęè£ĞæĂŐæăůä¡£çŤĺåĞ¡æŢřåŔĆæŢřæşÍèğčïijŇéĆčäźĹä¡ăåŔŕèČ¡äijŽæČşåĹľçŤĺåőČæİěåőđ ä¡ĘæŸŕä¡ääÿ■çąőåőŽåžŤèŕěæĂŐæăůåŐżåőđçŐŕïijĹæĹŰèĂĚåĹřåžŢèąŇå¿ŮéĂŽäÿ■ïijĽ'ãĂĆ

èğčåĘşæŰźæąĹ

æIJňåřŔèŁĆçŽĎæŁĂæIJŕæŸŕå§žäžŐäÿĂäÿłçőĂå■ŢçŽĎæŁĂæIJŕïijŇéĆčåŕśæŸŕPythonåĚĄèőÿåŔĆæŢ

```
class Spam:
    def bar(self, x:int, y:int):
        print('Bar 1:', x, y)
```

```
    def bar(self, s:str, n:int = 0):
        print('Bar 2:', s, n)

s = Spam()
s.bar(2, 3) # Prints Bar 1: 2 3
s.bar('hello') # Prints Bar 2: hello 0
```

äÿŃéÌćæŸŕæĹŚäżñçňňäÿĂæ■ěçŽĎåřÌèŕŢïijŇä¡£çŤĺåĹřäžĘäÿĂäÿłåĚČçśżåŠŇæŔŘè£řåŹĺïijŽ

```
# multiple.py
import inspect
import types

class MultiMethod:
    '''
    Represents a single multimethod.
    '''
    def __init__(self, name):
        self._methods = {}
        self.__name__ = name

    def register(self, meth):
        '''
        Register a new method as a multimethod
        '''
        sig = inspect.signature(meth)

        # Build a type signature from the method's annotations
        types = []
        for name, parm in sig.parameters.items():
            if name == 'self':
                continue
            if parm.annotation is inspect.Parameter.empty:
                raise TypeError(
                    'Argument {} must be annotated with a type'.
↪format(name)
                )
            if not isinstance(parm.annotation, type):
                raise TypeError(
                    'Argument {} annotation must be a type'.
↪format(name)
                )
            if parm.default is not inspect.Parameter.empty:
                self._methods[tuple(types)] = meth
            types.append(parm.annotation)

        self._methods[tuple(types)] = meth

    def __call__(self, *args):
```

```
        '''
        Call a method based on type signature of the arguments
        '''
        types = tuple(type(arg) for arg in args[1:])
        meth = self._methods.get(types, None)
        if meth:
            return meth(*args)
        else:
            raise TypeError('No matching method for types {}'.
↪format(types))

    def __get__(self, instance, cls):
        '''
        Descriptor method needed to make calls work in a class
        '''
        if instance is not None:
            return types.MethodType(self, instance)
        else:
            return self

class MultiDict(dict):
    '''
    Special dictionary to build multimethods in a metaclass
    '''
    def __setitem__(self, key, value):
        if key in self:
            # If key already exists, it must be a multimethod or␣
↪callable
            current_value = self[key]
            if isinstance(current_value, MultiMethod):
                current_value.register(value)
            else:
                mvalue = MultiMethod(key)
                mvalue.register(current_value)
                mvalue.register(value)
                super().__setitem__(key, mvalue)
        else:
            super().__setitem__(key, value)

class MultipleMeta(type):
    '''
    Metaclass that allows multiple dispatch of methods
    '''
    def __new__(cls, clsname, bases, clsdict):
        return type.__new__(cls, clsname, bases, dict(clsdict))

    @classmethod
    def __prepare__(cls, clsname, bases):
        return MultiDict()
```

äÿžäžĘä¡£çŤĺè£ŹäÿłçśżïijŇä¡ääŔŕäžěåČŔäÿŇéİćè£ŹæăůåĘŹïijŽ

```
class Spam(metaclass=MultipleMeta):
    def bar(self, x:int, y:int):
        print('Bar 1:', x, y)

    def bar(self, s:str, n:int = 0):
        print('Bar 2:', s, n)

# Example: overloaded __init__
import time

class Date(metaclass=MultipleMeta):
    def __init__(self, year: int, month:int, day:int):
        self.year = year
        self.month = month
        self.day = day

    def __init__(self):
        t = time.localtime()
        self.__init__(t.tm_year, t.tm_mon, t.tm_mday)
```

äÿŇéİćæŸŕäÿĂäÿłäžd'äžŠçd'žä¿ŇæİěéłŇèŕĄåőČèČ¡æ■čçąőçŽĎåůěä¡IJïijŽ

```
>>> s = Spam()
>>> s.bar(2, 3)
Bar 1: 2 3
>>> s.bar('hello')
Bar 2: hello 0
>>> s.bar('hello', 5)
Bar 2: hello 5
>>> s.bar(2, 'hello')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "multiple.py", line 42, in __call__
        raise TypeError('No matching method for types {}'.
 ↪format(types))
TypeError: No matching method for types (<class 'int'>, <class 'str
 ↪'>)
>>> # Overloaded __init__
>>> d = Date(2012, 12, 21)
>>> # Get today's date
>>> e = Date()
>>> e.year
2012
>>> e.month
12
>>> e.day
3
>>>
```

èőÍèőž

åÌęçŹ¡æÌèèőšïijÑçŻÿåŕźäžŐéĂŽåÿÿçŽĎäžčçăĄèĂŇåůšæIJňèŁĆä¡£çŤÍåĹřäžĘå¿Ĺåď'ŽçŽĎé■ŤæşŢäžčçă
ä¡ĘæŸŕïijŇåőČå■ť'èČ¡èől'æĹŚäžňæůśåĚěçŘĘèğčåĚČçśżåŠÑæŔŔè£řåŹÍçŽĎåžŢåśĆåůěä¡IJåŐ§çŘĘïijÑ
åźűèČ¡åŁäæůśåŕźè£ŹäžŻæęĆå£ţçŽĎå■řèśąãĂĆåŻäæ■d'ïijÑåřśçőŮä¡ääźűäÿ■äijŽçńÑå■şåŐżåžŤçŤÍæIJňèŁĆ
åőČçŽĎäÿĂäžŻåžŢåśĆæĂİæČşå■ť'äijŽå¡śåŞ■åĹřåĚűåőČæűL'åŔŁåĹřåĚČçśżãĂĄæŔŔè£řåŹÍåŠÑåĞ¡æŢřæş

æIJňèŁĆçŽĎåőđçŐřäÿ■çŽĎäÿżèęĄæĂİèůřåĚűåőđæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆMutipleMeta
åĚČçśżä¡£çŤÍåőČçŽĎ __prepare__() æŰźæşŢ æİěæŔŘä¿ŻäÿĂäÿłä¡IJäÿž MultiDict
åőđä¿ŃçŽĎèĞłåőŽäźL'å■ŮåĚÿãĂĆè£Źäÿłèů§æŹőéĂŽå■ŮåĚÿäÿ■äÿĂæäůçŽĎæŸŕïijÑ
MultiDict äijŽåIJÍåĚČçt'äèćńèő¿ç¡őçŽĎæŮűåĂŹæčĂæ§ěæŸŕåŘęåůšçżŔå■ŸåIJÍïijÑåęĆæđIJå■ŸåIJÍçŽĎ
MultiMethod åőđä¿Ńäÿ■åŘĹåźűãĂĆ

MultiMethod åőđä¿ŃéĂŽè£ĞæđĎåžżäżŐçśżåđÑç■¿åŘ■åĹřåĞ¡æŢřçŽĎæŸäåřĎæİěæŤűéŻĘæŰźæşŢ
åIJÍè£ŹäÿłæđĎåžżè£ĞçÍÑäÿ■ïijÑåĞ¡æŢřæşÍèğčèćńçŤÍæİěæŤűéŻĘè£ŹäžŻç■¿åŘ■çĎůåŘŐæđĎåžżè£ŹäÿłæŸ
è£Źäÿłè£ĞçÍÑåIJÍ MultiMethod.register() æŰźæşŢäÿ■åőđçŐřãĂĆ
è£Źçğ■æŸäåřĎçŽĎäÿĂäÿłåĚşéŤőçL'źçĆźæŸŕåŕźäžŐåď'ŽäÿłæŰźæşŢïijÑæL'ĂæIJL'åŔĆæŢřçśżåđÑéČ¡å£ĚéĆ

äÿžäžĘèől' MultiMethod åőđä¿ŃæÍąæÑ§äÿĂäÿłèřĆçŤÍïijÑåőČçŽĎ
__call__() æŰźæşŢèćńåőđçŐřäžĘãĂĆ è£ŹäÿłæŰźæşŢäżŐæL'ĂæIJL'æŐŠéŹď slef
çŽĎåŔĆæŢřäÿ■æđĎåžżäÿĂäÿłçśżåđÑåĚČçžĎïijÑåIJÍåĘĚéČÍmapäÿ■æ§ěæL'¿è£ŹäÿłæŰźæşŢïijÑ
çĎůåŘŐèřČçŤÍçŽÿåžŤçŽĎæŰźæşŢãĂĆäÿžäžĘèČ¡èől' MultiMethod
åőđä¿ŃåIJÍçśżåőŽäźL'æŮűæ■čçąőæŞ■ä¡IJïijÑ__get__() æŸŕå£Ěéążå¿ŮåőđçŐřçŽĎãĂĆ
åőČèćńçŤÍæİěæđĎåžżæ■čçąőçŽĎçżŠåőŽæŰźæşŢãĂĆæŕŤåęĆïijŽ

```
>>> b = s.bar
>>> b
<bound method Spam.bar of <__main__.Spam object at 0x1006a46d0>>
>>> b.__self__
<__main__.Spam object at 0x1006a46d0>
>>> b.__func__
<__main__.MultiMethod object at 0x1006a4d50>
>>> b(2, 3)
Bar 1: 2 3
>>> b('hello')
Bar 2: hello 0
>>>
```

äÿ■è£ĞæIJňèŁĆçŽĎåőđçŐřè£ŸæIJL'äÿĂäžŻéŹŘåĹűïijÑåĚűäÿ■äÿĂäÿłæŸŕåőČäÿ■èČ¡ä¡£çŤÍåĚşéŤőå■

```
>>> s.bar(x=2, y=3)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: __call__() got an unexpected keyword argument 'y'

>>> s.bar(s='hello')
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: __call__() got an unexpected keyword argument 's'
>>>
```

äź§èőÿæIJL’åĚűäżŰçŽĎæŰźæşŢèČ¡æůżåŁäè£Źçğ■æŤŕæŇĄïijŇä¡ĘæŸŕåőČéIJĂèęĄäÿĂäÿłåőŇåĚĺäÿ■
éŮőécŸåIJĺäžŐåĚşéŤőå■ŮåŔĆæŢřçŽĎåĞžçŐřæŸŕæšąæIJL’éąžåžŔçŽĎãĂĆå¡ŞåőČèů§ä¡■ç¡őåŔĆæŢřæůůåĚ
éĆčä¡äçŽĎåŔĆæŢřářšäijŽåŔŸå¿ŮæŕŤè¿ČæůůäźśäžĘïijŇè£ŹæŮűåĂŹä¡ääÿ■å¿Ůäÿ■åIJĺ
`__call__()` æŰźæşŢäÿ■åĚĹåŐżåĄŽäÿłæŐŠåžŔãĂĆ

åŘŇæăůåŕźäžŐçżğæL’£äź§æŸŕæIJL’éŹŘåĹűçŽĎïijŇä¿ŃåęĆïijŇçśżäijijäÿŃéİćè£Źçğ■äżčçăĄåřšäÿ■èČ¡

```
class A:
    pass

class B(A):
    pass

class C:
    pass

class Spam(metaclass=MultipleMeta):
    def foo(self, x:A):
        print('Foo 1:', x)

    def foo(self, x:C):
        print('Foo 2:', x)
```

åŐ§åŻäæŸŕåŻääÿž `x:A` æşĺèğčäÿ■èČ¡æĹŘåŁ§åŇźéĚ■å■Řçśżåőđä¿ŃïijĹæŕŤåęĆBçŽĎåőđä¿ŃïijL’ïijŇå

```
>>> s = Spam()
>>> a = A()
>>> s.foo(a)
Foo 1: <__main__.A object at 0x1006a5310>
>>> c = C()
>>> s.foo(c)
Foo 2: <__main__.C object at 0x1007a1910>
>>> b = B()
>>> s.foo(b)
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "multiple.py", line 44, in __call__
        raise TypeError('No matching method for types {}'.
 ↪format(types))
TypeError: No matching method for types (<class '__main__.B'>,)
>>>
```

ä¡IJäÿžä¡£çŤĺåĚČçśżåŠŇæşĺèğčçŽĎäÿĂçğ■æŻ£äžčæŰźæąĹïijŇåŘŕäžěéĂŽè£ĞæŔŘè£ŕåŹĺæİěåőđçŐřç

```
import types

class multimethod:
    def __init__(self, func):
        self._methods = {}
        self.__name__ = func.__name__
        self._default = func
```

```
    def match(self, *types):
        def register(func):
            ndefaults = len(func.__defaults__) if func.__defaults__␣
↪else 0
            for n in range(ndefaults+1):
                self._methods[types[:len(types) - n]] = func
            return self
        return register

    def __call__(self, *args):
        types = tuple(type(arg) for arg in args[1:])
        meth = self._methods.get(types, None)
        if meth:
            return meth(*args)
        else:
            return self._default(*args)

    def __get__(self, instance, cls):
        if instance is not None:
            return types.MethodType(self, instance)
        else:
            return self
```

äÿžäžĘä¡£çŤÍæŔŔè£řåŹÍçL’ĹæIJňïijŇä¡ăéIJĂèęĄåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
class Spam:
    @multimethod
    def bar(self, *args):
        # Default method called if no match
        raise TypeError('No matching method for bar')

    @bar.match(int, int)
    def bar(self, x, y):
        print('Bar 1:', x, y)

    @bar.match(str, int)
    def bar(self, s, n = 0):
        print('Bar 2:', s, n)
```

æŔŔè£řåŹÍæŰźæąĹåŘŇæăůäź§æIJL’åL’■éİćæŔŔåĹřçŽĎéŹŘåĹűïijĹäÿ■æŤŕæŇĄåĚşéŤőå■ŮåŔĆæŢřåŠ

æL’ĂæIJL’äžŃçL’ľéČ¡æŸŕåźşç■L’çŽĎïijŇæIJL’åě¡æIJL’åİŔïijŇäź§èőÿæIJĂåě¡çŽĎåŁđæşŢåřśæŸŕåIJÍæŹ
äÿ■è£ĞæIJL’äžŻçL’źæőŁæČĚåĘţäÿŃè£ŸæŸŕæIJL’æĎŔäźL’çŽĎïijŇærŤåęĆå§žäžŐæĺąåijŔåŇźéĚ■çŽĎæŰźä
äÿ¿äÿłä¿Ńå■ŘïijŇ8.21åřŔèŁĆäÿ■çŽĎèő£éŮőèĂĚæĺąåijŔåŔŕäżëä£őæŤźäÿžäÿĂäÿłä¡£çŤÍæŰźæşŢéĞ■è¡¡çŽ
ä¡ĘæŸŕïijŇéŹđ’äžĘè£Źäÿłäžëåđ’ŰïijŇéĂŽåÿÿäÿ■åžŤèŕëä¡£çŤÍæŰźæşŢéĞ■è¡¡ïijĹåřśçőĂå■ŢçŽĎä¡£çŤÍäÿ■å

åIJÍPythonçđ’¿åŇžåŕźäžŐåőđçŐřæŰźæşŢéĞ■è¡¡çŽĎèőÍèőžåůšçżŔçŤśæİëåůšäźĚãĂĆ
åŕźäžŐåijŢåŔŚè£ŹäÿłäžL’èőžçŽĎåŐ§åŻäïijŇåŔŕäżëåŔĆèĂČäÿŃGuido van RossumçŽĎè£ŹçŕĞå■Žåőćïij Five-Minute Multimethods in Python

9.21 éĄ£åĚ■éĞ■åď■çŽĎåśđæĂğæŰźæşŢ

éŮőécŸ

ä¡ăåIJĺçśżäÿ■éIJĂèęĄéĞ■åď■çŽĎåőŽäźL'äÿĂäžŻæL'ğèąŇçŻÿåŘŇéĂżè¿ŚçŽĎåśđæĂğæŰźæşŢïijŇæŕŤ

èğčåĘşæŰźæąĹ

èĂČèŹŚäÿŃäÿĂäÿłçőĂå■ŢçŽĎçśżïijŇåőČçŽĎåśđæĂğçŤśåśđæĂğæŰźæşŢåŇĚèčĚïijŽ

```
class Person:
    def __init__(self, name ,age):
        self.name = name
        self.age = age

    @property
    def name(self):
        return self._name

    @name.setter
    def name(self, value):
        if not isinstance(value, str):
            raise TypeError('name must be a string')
        self._name = value

    @property
    def age(self):
        return self._age

    @age.setter
    def age(self, value):
        if not isinstance(value, int):
            raise TypeError('age must be an int')
        self._age = value
```

åŔŕäžěçIJŃåĹřïijŇäÿžäžĘåőđçŐřåśđæĂğåĂijçŽĎçśżåđŃæčĂæ§ěæĹŚäżňåĘŹäžĘå¿Ĺåď'ŽçŽĎéĞ■åď■ä
åŔłèęĄä¡ăäžěåŘŐçIJŃåĹřçśżäijijè£ŹæăůçŽĎäżčçăĄïijŇä¡ăéČ¡åžŤèřěæČşåŁđæşŢåŐżçőĂåŇŰåőČãĂĆ
äÿĂäÿłåŔŕèąŇçŽĎæŰźæşŢæŸŕåĹŻåżžäÿĂäÿłåĞ¡æŢřçŤĺæİěåőŽäźL'åśđæĂğåźűè£ŤåŽđåőČãĂĆä¿ŃåęĆïijŽ

```
def typed_property(name, expected_type):
    storage_name = '_' + name

    @property
    def prop(self):
        return getattr(self, storage_name)

    @prop.setter
    def prop(self, value):
        if not isinstance(value, expected_type):
```

```
            raise TypeError('{} must be a {}'.format(name, expected_
↪type))
        setattr(self, storage_name, value)

    return prop

# Example use
class Person:
    name = typed_property('name', str)
    age = typed_property('age', int)

    def __init__(self, name, age):
        self.name = name
        self.age = age
```

èőÍèőž

æIJňèŁĆæĹŚäżňæijŤçď žåĘĚéČĺåĞ¡æŢřæĹŰèĂĚéŮ■åŇĚçŽĎäÿĂäÿłéĞ■èęĄçL’źæĂğïijŇåőČäżňå¿ĹåC
typed_property() çIJŃäÿŁåŐżæIJL’çĆźéŽ¿çŘĘèğčïijŇåĚűåőđåőČæL’ĂåĄŽçŽĎäžĚäżĚåřśæŸřäÿžä¡ăç
åŻăæ■ď’ïijŇå¡ŞåIJĺäÿĂäÿłçśżäÿ■ä¡£çŤĺåőČçŽĎæŮűåĂŹïijŇæŢĹæđIJè§åřĘåőČéĞŇéİćçŽĎäžčçăĄæŤ¿åĹř
åř¡çőąåśđæĂğçŽĎ getter åŠŇ setter æŰźæşŢèő£éŮőäžĘæIJňåIJřåŔŸéĞŔąęĆ name ,
expected_type äžěåŘŁ storate_name ïijŇè£Źäÿłå¿Ĺæ■čåÿÿïijŇè£ŹäžŻåŔŸéĞŔçŽĎåĂijäijŽä£İå■Ÿ

æĹŚäżňè£ŸåŔŕäžěä¡£çŤĺ functools.partial() æİěçĺ■çĺ■æŤźåŔŸäÿŃè£Źäÿłäż¿Ńå■ŘïijŇå¿ĹæIJI

```
from functools import partial

String = partial(typed_property, expected_type=str)
Integer = partial(typed_property, expected_type=int)

# Example:
class Person:
    name = String('name')
    age = Integer('age')

    def __init__(self, name, age):
        self.name = name
        self.age = age
```

åĚűåőđä¡ăåŔŕäžěåŔŚçŐřïijŇè£ŹéĞŇçŽĎäżčçăĄèů§8.13åřŔèŁĆäÿ■çŽĎçśżåđŃçşżçż§æŔŔè£řåŹĺäżčçă

9.22 åőŽäźĽäÿŁäÿŃæŰĞçőąçŘĘåŹĺçŽĎçőĂå■ŢæŰźæşŢ

éŮőécŸ

ä¡ăæČşèĞłåůšåŐżåőđçŐřäÿĂäÿłæŰřçŽĎäÿŁäÿŃæŰĞçőąçŘĘåŹĺïijŇäžěä¿£ä¡£çŤĺwithèŕ■åŘěãĂĆ

èğčåĘşæŰźæąĹ

åőđçŐřäÿĂäÿłæŰřçŽĎäÿŁäÿŃæŰĞçőąçŘĘåŹĺçŽĎæIJĂçőĂå■ŢçŽĎæŰźæşŢåřśæŸřä¡£çŤĺ contexlib æĺąåİŮäÿ■çŽĎ @contextmanager èčĚéěřåŹĺãĂĆ äÿŃéİćæŸřäÿĂäÿłåőđçŐřäžĘäžčçăĄåİŮèőąæŮűåŁ§èČ¡çŽĎäÿŁäÿŃæŰĞçőąçŘĘåŹĺä¿Ńå■ŘïijŽ

```
import time
from contextlib import contextmanager

@contextmanager
def timethis(label):
    start = time.time()
    try:
        yield
    finally:
        end = time.time()
        print('{}: {}'.format(label, end - start))

# Example use
with timethis('counting'):
    n = 10000000
    while n > 0:
        n -= 1
```

åIJĺåĞ¡æŢř timethis() äÿ■ïijŇyield äźŃåĽ’■çŽĎäžčçăĄäijŽåIJĺäÿŁäÿŃæŰĞçőąçŘĘåŹĺäÿ■ä¡IJäÿ __enter__() æŰźæşŢæĽ’ğèąŇïijŇ æĽ’ĂæIJĽåIJĺ yield äźŃåŘŐçŽĎäžčçăĄäijŽä¡IJäÿž __exit__() æŰźæşŢæĽ’ğèąŇãĂĆ åęĆæđIJåĞžçŐřäžĘåijĆåÿÿïijŇåijĆåÿÿäijŽåIJĺyield-èŕ■åŘěéĆčéĞŇæŁŻåĞžãĂĆ

äÿŃéİćæŸřäÿĂäÿłæŻť’åŁäéńŸçžğäÿĂçĆźçŽĎäÿŁäÿŃæŰĞçőąçŘĘåŹĺïijŇåőđçŐřäžĘåĹŮèąĺåŕźèśąäÿŁç

```
@contextmanager
def list_transaction(orig_list):
    working = list(orig_list)
    yield working
    orig_list[:] = working
```

è£Źæőţäžčçă ĄçŽĎä¡IJçŤĺæŸřäżżä¡Ţåŕźå Ĺ ŮèąĺçŽĎä£őæŤźåŘłæIJĽå¡ŞæĽ’ĂæIJĽäžčçăĄè£ŘèąŇåőŇæ äÿŃéİćæĹŚäžňæİěæijŤçd’žäÿĂäÿŃïijŽ

```
>>> items = [1, 2, 3]
>>> with list_transaction(items) as working:
...     working.append(4)
...     working.append(5)
...
>>> items
[1, 2, 3, 4, 5]
>>> with list_transaction(items) as working:
...     working.append(6)
...     working.append(7)
...     raise RuntimeError('oops')
```

```
...
Traceback (most recent call last):
    File "<stdin>", line 4, in <module>
RuntimeError: oops
>>> items
[1, 2, 3, 4, 5]
>>>
```

èőÍèőž

éĂŽåÿÿæČĚåĘţäÿŃĭijŇåęĆæđIJèęĄåĘŹäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺĭijŇä¡ăéIJĂèęĄåőŽäźL’äÿĂäÿłç
`__enter__()` åŠŇäÿĂäÿł `__exit__()` æŰźæşŢĭijŇåęĆäÿŃæL’Ăçd’žĭijŽ

```
import time

class timethis:
    def __init__(self, label):
        self.label = label

    def __enter__(self):
        self.start = time.time()

    def __exit__(self, exc_ty, exc_val, exc_tb):
        end = time.time()
        print('{}: {}'.format(self.label, end - self.start))
```

åř¡çőąè£Źäÿłäź§äÿ■éŽ¿åĘŹĭijŇä¡ĘæŸŕçŻÿæŕŤè¿ČåĘŹäÿĂäÿłçőĂå■ŢçŽĎä¡£çŤÍ
`@contextmanager` æşÍèğčçŽĎåĞ¡æŢřèĂŇèÍĂè£ŸæŸŕçÍ■æŸ¿äźŔåŚşãĂĆ

`@contextmanager` åžŤèŕëäżĚäżĚçŤÍæİěåĘŹèĞłåŇĚåŘńçŽĎäÿŁäÿŃæŰĞçőąçŘĘåĞ¡æŢřãĂĆ
åęĆæđIJä¡ăæIJL’äÿĂäžŽåŕźèšą(æŕŤåęĆäÿĂäÿłæŰĞäźűãĂĄç¡ŚçżIJè£đæŐěæĹŰéŤĄ)ïijŇéIJĂèęĄæŤŕæŇĄ
`with` èŕ■åŘëïijŇéĆčäźĹä¡ăåřšéIJĂèęĄå■ŢçŃňåðđçŐř `__enter__()` æŰźæşŢåŠŇ
`__exit__()` æŰźæşŢãĂĆ

9.23 åIJĺåšĂéČĺåŔŸéĞŔå§§äÿ■æL’ğèąŇäžčçăĄ

éŮőécŸ

ä¡ăæČşåIJĺä¡£çŤĺèŇČåŻť åĘĚæL’ğèąŇæ§Řäÿłäžčçă ĄçL’Ğæőţïij Ňåźűäÿŧåÿ ŇæIJŻåIJĺæL’ğèąŇåŘŐæL’Ă

èğčåĘşæŰźæąĹ

äÿžäžĘçŘĘèğčè£ŹäÿłéŮőécŸïijŇåĚĹèŕŢèŕŢäÿĂäÿłçőĂå■ŢåIJžæŹŕãĆéęŰåĚĹïijŇåIJĺåĚĺåśĂåŚ¡åŘ■ç

```
>>> a = 13
>>> exec('b = a + 1')
```

```
>>> print(b)
14
>>>
```

çĎűåŘŐïijŇåĘ■åIJÍäÿĂäÿłåĞ¡æŢřäÿ■æL’ğèąŇåŘŇæăůçŽĎäžčçăĄïijŽ

```
>>> def test():
...     a = 13
...     exec('b = a + 1')
...     print(b)
...
>>> test()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
    File "<stdin>", line 4, in test
NameError: global name 'b' is not defined
>>>
```

åŘŕäžěçIJŇåĞžïijŇæIJĂåŘŐæŁŻåĞžäžĘäÿĂäÿłNameErrorråijĆåÿÿïijŇåřśèů§åIJÍ `exec()` èŕ■åŘěäžŐæšąæL’ğèąŇè£ĞäÿĂæăůãĂĆ èęĄæŸŕä¡äæČşåIJÍåŘŐéÍćçŽĎèőąçőŮäÿ■ä¡£çŤÍåĹř `exec()` æL’ğèąŇçżŞæđIJçŽĎèŕÍåřśäijŽæIJL’éŮőécŸäžĘãĂĆ

äÿžäžĘä£őæ■čè£ŹæăůçŽĎéŤŹèŕŕïijŇä¡ăéIJĂèęĄåIJÍèřČçŤÍ `exec()` äźŇåL’■ä¡£çŤÍ `locals()` åĞ¡æŢřæÍěå¿ŮåĹřäÿĂäÿłåśĂéČÍåŔŸéĞŔå■ŮåĚÿãĂĆ äźŇåŘŐä¡ääřśèČ¡äžŐåśĂéČÍå■ŮåĚÿäÿ■èŐůåŔŮä£őæŤźè£ĞåŘŐçŽĎåŔŸéĞŔåĂijäžĘãĂĆä¿ŇåęĆïijŽ

```
>>> def test():
...     a = 13
...     loc = locals()
...     exec('b = a + 1')
...     b = loc['b']
...     print(b)
...
>>> test()
14
>>>
```

èőÍèőž

åőđéŹĚäÿŁåŕźäžŐ `exec()` çŽĎæ■čçąőä¡£çŤÍæŸŕæŕŤè¿ČéŽ¿çŽĎãĂĆåď’ğåď’ŽæŢřæČĚåĘţäÿŇå¡Şä¡ãè `exec()` çŽĎæŮűåĂŹïijŇ è£ŸæIJL’åŔęåď’ŰæŻť’åě¡çŽĎèğčåĘşæŰźæąĹïijĹæŕŤåęĆèčĚéěřåŹÍãĂĄéŮ■åŇĚã

çĎűèĂŇïijŇåęĆædIJä¡ääž■çĎűèęĄä¡£çŤÍ `exec()` ïijŇæIJňèŁĆåĹŮåĞžäžĘäÿĂäžŻåęĆä¡Ţæ■čçąőä¡£ç éżŸèőď’æČĚåĘţäÿŇïijŇ`exec()` äijŽåIJÍèřČçŤÍèĂĚåśĂéČÍåŠŇåĚÍåśĂèŇČåŻť’åĘĚæL’ğèąŇäžčçăĄãĂĆçĎűäijäéĂŠçżŹ `exec()` çŽĎåśĂéČÍèŇČåŻť’æŸŕæŮůèť’Íåőđé ŹĚåśĂéČÍåŔŸéĞŔçżĎæĹŔçŽĎäÿĂäÿłå■ŮåĚÿãĂ åŻăæ■ď’ïijŇåęĆædIJ `exec()` åęĆædIJæL’ğèąŇäžĘä£őæŤźæŞ■ä¡IJïijŇè£ŹçğäŁőæŤźåŘŐçŽĎçżŞædIJåŕźåčäÿŇéÍćæŸŕåŔęåď’ŰäÿĂäÿłæijŤçď’žåőČçŽĎä¿Ňå■ŘïijŽ

```
>>> def test1():
...     x = 0
...     exec('x += 1')
...     print(x)
...
>>> test1()
0
>>>
```

äÿŁéİćäżčçăĄéĞŇïijŇå¡Şä¡ăèřČçŤÍ `locals()` èŐůåŔŰåśĂéČĺåŔŸéĞŔæŮűïijŇä¡ăèŐůå¿ŮçŽĎæŸŕäi
`exec()` çŽĎåśĂéČĺåŔŸéĞŔçŽĎäÿĂäÿłæŃůèť´ÍãĂĆ éĂŽè£ĞåIJĺäżčçăĄæL'ğèąŇåŘŐåőąæ§ěè£Źäÿłå■ŮåĚÿ

```
>>> def test2():
...     x = 0
...     loc = locals()
...     print('before:', loc)
...     exec('x += 1')
...     print('after:', loc)
...     print('x =', x)
...
>>> test2()
before: {'x': 0}
after: {'loc': {...}, 'x': 1}
x = 0
>>>
```

äżŤçżĘèğĆåŕ§æIJĂåŘŐäÿĂæ■ęçŽĎè¿ŞåĞžïijŇéŹď´éİđä¡ăåřĘ `loc`
äÿ■èćńä£őæŤźåŘŐçŽĎåĂijæĽŃåŁĺèţŃåĂijçżŹxïijŇåŘęåĹŹxåŔŸéĞŔåĂijæŸŕäÿ■äijŽåŔŸçŽĎãĂĆ

åIJĺä¡£çŤÍ `locals()` çŽĎæŮűåĂŹïijŇä¡ăéIJĂèęĄæşĺæĎŔæŞ■ä¡IJéąžåžŔãĂĆærŔæňąåőČèćńèřČçŤĺçŽ
`locals()` äijŽèŐůåŔŰåśĂéČĺåŔŸéĞŔåĂijäÿ■çŽĎåĂijåźűèęĘçŻŰå■ŮåĚÿäÿ■çŻÿåžŤçŽĎåŔŸéĞŔãĂĆ
èŕůæşĺæĎŔèğĆåŕ§äÿŃäÿŃéİćè£ŹäÿłèŕŢéłŃçŽĎè¿ŞåĞžçżŞæđIJïijŽ

```
>>> def test3():
...     x = 0
...     loc = locals()
...     print(loc)
...     exec('x += 1')
...     print(loc)
...     locals()
...     print(loc)
...
>>> test3()
{'x': 0}
{'loc': {...}, 'x': 1}
{'loc': {...}, 'x': 0}
>>>
```

æşĺæĎŔæIJĂåŘŐäÿĂæňąèřČçŤÍ `locals()` çŽĎæŮűåĂŹxçŽĎåĂijæŸŕåęĆä¡Ţèćńèę Ęç ŻŰæŐL'çŽĎã

ä¡IJäÿž `locals()` çŽĎäÿĂäÿłæŻ£äżčæŰźæąĹïijŇä¡ăåŔŕäżěä¡£çŤĺä¡ăèĞłåůśçŽĎå■ŮåĚÿïijŇåźűåřĘåő

exec() ãĂĆä¿ŃåęĆïijŽ

```
>>> def test4():
...     a = 13
...     loc = { 'a' : a }
...     glb = { }
...     exec('b = a + 1', glb, loc)
...     b = loc['b']
...     print(b)
...
>>> test4()
14
>>>
```

åď ğéČĺåĹĘæČĚåĘţäÿŃïijŇè£Źçğ■æŰźåijŔæŸŕä¡£çŤĺ exec() çŽĎæIJĂä¡şåőđèůţãĂĆ ä¡ăåŔłéIJĂèęĄä£ÌèŕĄåĚĺåśĂåŠŇåśĂéČĺå■ŮåĚÿåIJĺåŘŐéİćäžčçăĄèő£éŮőæŮűåůšçżŔèćńåĹİågŃåŇŰãĂĆ

è£ŸæIJL'äÿĂçĆźïijŇåIJĺä¡£çŤĺexec() äźŃåL'■ïijŇä¡ăåŔŕèČ¡éIJĂèęĄéŮőäÿŃèĞłåůśæŸŕåŘęæIJL'åĚú åď ğåď ŽæŢřæČĚåĘţäÿŃå¡Şä¡ăèęĄèĂČèŹŚä¡£çŤĺ exec() çŽĎæŮűåĂŹïijŇ è£ŸæIJL'åŘęåď ŰæŻt'ăě¡çŽĎèğčåĘşæŰźæąĹïijŇæŕŤåęĆèčĚéěřåŹĺãĂĄéŮ■åŇĚãĂĄåĚČçśżïijŇæĹŰåĚűäżŰ

9.24 èğčædŘäÿŐåĹĘædŘPythonæžŘçăĄ

éŮőécŸ

ä¡ăæČşåĘŹèğčædŘåźűåĹĘædŘPythonæžŘäżčçăĄçŽĎçĺŃåžŘãĂĆ

èğčåĘşæŰźæąĹ

åď ğéČĺåĹĘçĺŃåžŘåŚŸç§ěéĄŞPythonèČ¡åď §èőąçőŮæĹŰæL'ğèąŇå■Ůçňęäÿšå¡ćåijŔçŽĎæžŘäżčçăĄãĂ

```
>>> x = 42
>>> eval('2 + 3*4 + x')
56
>>> exec('for i in range(10): print(i)')
0
1
2
3
4
5
6
7
8
9
>>>
```

ař¡çőąåęĆæ■d'ïijŇast æĺąåİŮèČ¡èćńçŤĺæİěářĘPythonæžŘçăĄçijŰèŕŚæĹŘäÿĂäÿłåŔŕèćńåĹĘædŘçŽĎa

```
>>> import ast
>>> ex = ast.parse('2 + 3*4 + x', mode='eval')
>>> ex
<_ast.Expression object at 0x1007473d0>
>>> ast.dump(ex)
"Expression(body=BinOp(left=BinOp(left=Num(n=2), op=Add(),
right=BinOp(left=Num(n=3), op=Mult(), right=Num(n=4))), op=Add(),
right=Name(id='x', ctx=Load())))"

>>> top = ast.parse('for i in range(10): print(i)', mode='exec')
>>> top
<_ast.Module object at 0x100747390>
>>> ast.dump(top)
"Module(body=[For(target=Name(id='i', ctx=Store()),
iter=Call(func=Name(id='range', ctx=Load()), args=[Num(n=10)],
keywords=[], starargs=None, kwargs=None),
body=[Expr(value=Call(func=Name(id='print', ctx=Load()),
args=[Name(id='i', ctx=Load())], keywords=[], starargs=None,
kwargs=None))], orelse=[])])"
>>>
```

åĹĘæđŘæžŘçăĄæăŚéIJÄèęĄä¡äèĞłåůśæŻt’åd’ŽçŽĎå■ęäźăïijŇåőČæŸŕçŤśäÿĂçşżåĹŮASTèŁĆçĆźçżĎ
åĹĘæđŘè£ŹäžŻèŁĆçĆźæIJÄçőĂå■ĘçŽĎæŰźæşĘářśæŸŕåőŽäźL’äÿĂäÿłèő£éŮőèĂĚçśżïijŇåőđçŐřå¿Łåd’Ž
`visit_NodeName()` æŰźæşĘïijŇ `NodeName()` åŇźéĚ■éĆčäžŻä¡äæĎ§åĚt’èűčçŽĎèŁĆçĆźãĂĆäÿŃéİćæ

```
import ast

class CodeAnalyzer(ast.NodeVisitor):
    def __init__(self):
        self.loaded = set()
        self.stored = set()
        self.deleted = set()

    def visit_Name(self, node):
        if isinstance(node.ctx, ast.Load):
            self.loaded.add(node.id)
        elif isinstance(node.ctx, ast.Store):
            self.stored.add(node.id)
        elif isinstance(node.ctx, ast.Del):
            self.deleted.add(node.id)

# Sample usage
if __name__ == '__main__':
    # Some Python code
    code = '''
    for i in range(10):
        print(i)
    del i
    '''
```

```
    # Parse into an AST
    top = ast.parse(code, mode='exec')

    # Feed the AST to analyze name usage
    c = CodeAnalyzer()
    c.visit(top)
    print('Loaded:', c.loaded)
    print('Stored:', c.stored)
    print('Deleted:', c.deleted)
```

åęĆæđIJä¡ăè£ŘèąŇè£ŹäŷłçÍŇåžŔïijŇä¡ăäijŽå¿ŮåĹřäŷŇéİćè£ŹæăůçŽĎè¿ŞåĞžïijŽ

```
Loaded: {'i', 'range', 'print'}
Stored: {'i'}
Deleted: {'i'}
```

æIJĂåŘŐïijŇASTåŔŕäžěéĂŽè£Ğ `compile()` åĞ¡æŢřæİěçijŰèŕŚåźűæL'ğèąŇãĂĆä¿ŃåęĆïijŽ

```
>>> exec(compile(top,'<stdin>', 'exec'))
0
1
2
3
4
5
6
7
8
9
>>>
```

èőĺèőž

å¡Şä¡ăèČ¡åd'§åĹĘæđŘæžŘäžčçăĄåźűäžŐäÿ■èŐůåŔŰä£ąæĄŕçŽĎæŮűåĂŹïijŇä¡ăåřśèČ¡åĘŹå¿Ĺåd'Žäžč
ä¿ŃåęĆïijŇçŻÿæŕŢçŻšçŻőçŽĎäijăéĂŠäÿĂäžŻäžčçăĄçL'ĞæőţåĹřçśżäijij
`exec()` åĞ¡æŢřäÿ■ïijŇä¡ăåŔŕäžěåĚĹåřĘåőČè¡ňæ■ćæĹŘäÿĂäÿłASTïijŇ
çĎűåŘŐèğĆår§åőČçŽĎçżĘèŁĆçIJŃåőČåĹřåžŢæŸŕæĂŐæăůåĄŽçŽĎãĂĆ
ä¡ăè£ŸåŔŕäžěåĘŹäÿĂäžŻåůěåĚůæİěæ§ěçIJŃæ§ŘäÿłæĺąåİŮçŽĎåĚĺéČĺæžŘçăĄïijŇåźűäÿŤåIJĺæ■d'å§žçąĂäÿ

éIJĂèęĄæşĺæĎŘçŽĎæŸŕïijŇåęĆæđIJä¡ăç§ěéĄŞèĞłåůšåIJĺåźšåŢěïijŇä¡ăè£ŸèČ¡åd'§éĞ■åĘŹASTæİěèą
äÿŃéİćæŸŕäŸĂäÿłèčĚéěřåŹĺä¿Ńå■ŘïijŇåŔŕäžěéĂŽè£ĞéĞ■æŰřèğčæđŘåĞ¡æŢřä¡ŞæžŘçăĄãĂĄ
éĞ■åĘŹASTåźűéĞ■æŰřåĹŽåžžåĞ¡æŢřäžčçăĄåŕźèśąæİěåřĘåĚĺåśĂèő£éŮőåŔŸéĞŔéŹ■äÿžåĞ¡æŢřä¡Şä¡IJçŤ

```
# namelower.py
import ast
import inspect

# Node visitor that lowers globally accessed names into
# the function body as local variables.
```

```
class NameLower(ast.NodeVisitor):
    def __init__(self, lowered_names):
        self.lowered_names = lowered_names

    def visit_FunctionDef(self, node):
        # Compile some assignments to lower the constants
        code = '__globals = globals()\n'
        code += '\n'.join("{0} = __globals['{0}']".format(name)
                          for name in self.lowered_names)
        code_ast = ast.parse(code, mode='exec')

        # Inject new statements into the function body
        node.body[:0] = code_ast.body

        # Save the function object
        self.func = node

# Decorator that turns global names into locals
def lower_names(*namelist):
    def lower(func):
        srclines = inspect.getsource(func).splitlines()
        # Skip source lines prior to the @lower_names decorator
        for n, line in enumerate(srclines):
            if '@lower_names' in line:
                break

        src = '\n'.join(srclines[n+1:])
        # Hack to deal with indented code
        if src.startswith((' ','\t')):
            src = 'if 1:\n' + src
        top = ast.parse(src, mode='exec')

        # Transform the AST
        cl = NameLower(namelist)
        cl.visit(top)

        # Execute the modified AST
        temp = {}
        exec(compile(top,'','exec'), temp, temp)

        # Pull out the modified code object
        func.__code__ = temp[func.__name__].__code__
        return func
    return lower
```

äÿžäžĘä¡£çŤĺè£Źäÿłäżčçă Ąïij Ňä¡ăåŔŕäžěåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
INCR = 1
@lower_names('INCR')
def countdown(n):
```

```
    while n > 0:
        n -= INCR
```

èčĚéěřåŹÍäijŽåřĘ `countdown()` åĞ¡æŢřéĞ■åĘŹäÿžçśżäijijäÿŃéÍćè£Źæăůå■ŘïijŽ

```
def countdown(n):
    __globals = globals()
    INCR = __globals['INCR']
    while n > 0:
        n -= INCR
```

åIJÍæĂğèČ¡æţŃèŕŢäÿ■ïijŇåőČäijŽèőľåĞ¡æŢřè£ŘèąŇå£ń20%

çŐřåIJÍïijŇä¡ăæŸŕäÿ■æŸŕæČşäÿžä¡ăæL'ĂæIJL'çŽĎåĞ¡æŢřéČ¡åŁääÿŁè£ŹäÿłèčĚéěřåŹÍåŚćïij§æĹŰèőÿä
ä¡ĘæŸŕïijŇè£Źå■t'æŸŕåŕźäžŐäÿĂäžŻéńŸçžğæŁĂæIJŕæŕŤåęĆASTæŞ■ä¡IJãĂĄæžŘçăĄæŞ■ä¡IJç■L'ç■L'çŽĎ

æIJňèŁĆåŔŮåŔęåd'ŰäÿĂäÿłåIJÍ `ActiveState` äÿ■åd'ĎçŘĘPythonå■ŮèŁĆçăĄçŽĎçńäèŁĆçŽĎåŘŕç
ä¡£çŤÍASTæŸŕäÿĂäÿłæŻt'åŁäéńŸçžğçĆźçŽĎæŁĂæIJŕïijŇåźűäÿŤäź§æŻt'çőĂå■ŢäžŻãĂĆåŔĆèĂČäÿŃéÍćäÿ

9.25 æŃĘèğčPythonå■ŮèŁĆçăĄ

éŮőécŸ

ä¡ăæČşéĂŽè£ĞářĘä¡ăçŽĎäżčçăĄåŔ■çijŰèŕŚæĹŘä¡ŐçžğçŽĎå■ŮèŁĆçăĄæÍěæ§ěçIJŃåőČåžŢåśĆçŽĎå

èğčåĘşæŰźæąĹ

`dis` æÍąåİŮåŔŕäżèèćńçŤÍæÍěè¿ŞåĞžäżżä¡ŢPythonåĞ¡æŢřçŽĎåŔ■çijŰèŕŚçżŞæđIJãĂĆä¿ŃåęĆïijŽ

```
>>> def countdown(n):
... while n > 0:
...     print('T-minus', n)
...     n -= 1
... print('Blastoff!')
...
>>> import dis
>>> dis.dis(countdown)
...
>>>
```

èőÍèőž

å¡Şä¡ăæČşèęAç§ěéĄŞä¡ăçŽĎçÍŃåžŔåžŢåśĆçŽĎè£ŘèąŇæIJžåĹűçŽĎæŮűåĂŹïijŃdis
æÍąåİŮæŸŕå¿ĹæIJL'çŤÍçŽĎãĂĆæŕŤåęĆåęĆæđIJä¡ăæČşèŕŢçİĂçŘĘèğčæĂğèČ¡çL'źå¿ĄãĂĆ
èćń `dis()` åĞ¡æŢřèğčæđŘçŽĎåŐ§åğŃå■ŮèŁĆçăĄåęĆäÿŃæL'Ăçd'žïijŽ

```
>>> countdown.__code__.co_code
b"x
 ↪'\x00|\x00\x00d\x01\x00k\x04\x00r)\x00t\x00\x00d\x02\x00|\x00\x00\x83
\x02\x00\x01|\x00\x00d\x03\x008}
 ↪\x00\x00q\x03\x00Wt\x00\x00d\x04\x00\x83
\x01\x00\x01d\x00\x00S"
>>>
```

åęĆædIJä¡ăæČşèĞłåůśèğčéĞŁè£Źæőţäžčçă Ąïij Ñä¡ăéIJĂèęĄä¡£çŤĺäÿĂäžŻåIJÍ `opcode` æĺąåİŮäÿ■åőŽäźL'çŽĎåÿÿéĞŔãĂĆä¿ŃåęĆïijŽ

```
>>> c = countdown.__code__.co_code
>>> import opcode
>>> opcode.opname[c[0]]
>>> opcode.opname[c[0]]
'SETUP_LOOP'
>>> opcode.opname[c[3]]
'LOAD_FAST'
>>>
```

åěĞæĂłçŽĎæŸŕïijŇåIJÍ `dis` æĺąåİŮäÿ■åźűæšąæIJL'åĞ¡æŢřèől'ä¡ăäžěçijŰçÍŃæŰźåijŔå¿Ĺåőźæ ŸŞçŽĎäÿ■è£ĞïijŇäÿŃéİćçŽĎçŤ§æĹŘåŹÍåĞ¡æŢřåŔŕäžěåřĘåŐ§ågŃå■ŮèŁĆçăĄåžŔåĹŮè¡ňæ■ćæĹŘ `opcodes` åŠŇåŔĆæŢřãĂĆ

```
import opcode

def generate_opcodes(codebytes):
    extended_arg = 0
    i = 0
    n = len(codebytes)
    while i < n:
        op = codebytes[i]
        i += 1
        if op >= opcode.HAVE_ARGUMENT:
            oparg = codebytes[i] + codebytes[i+1]*256 + extended_arg
            extended_arg = 0
            i += 2
            if op == opcode.EXTENDED_ARG:
                extended_arg = oparg * 65536
                continue
        else:
            oparg = None
        yield (op, oparg)
```

ä¡£çŤĺæŰźæşŢåęĆäÿŃïijŽ

```
>>> for op, oparg in generate_opcodes(countdown.__code__.co_code):
...     print(op, opcode.opname[op], oparg)
```

è£Źçğ■æŰźåijŔå¿ĹåřŚæIJL'äžžç§ěéĄŞïijÑä¡ăåŔŕäžěåĹl'çŤĺáőĆæŻ£æ■ćäżä¡Ţä¡ăæČşèęĄæŻ£æ■ćçŽĎ äÿŃéİćæĹŚäžñçŤĺäÿĂäÿłçd'žä¿ŃæİěæijŤçd'žæŢt'äÿłè£ĞçĺŃïijŽ

```
>>> def add(x, y):
...     return x + y
...
>>> c = add.__code__
>>> c
<code object add at 0x1007beed0, file "<stdin>", line 1>
>>> c.co_code
b'|\x00\x00|\x01\x00\x17S'
>>>
>>> # Make a completely new code object with bogus byte code
>>> import types
>>> newbytecode = b'xxxxxxx'
>>> nc = types.CodeType(c.co_argcount, c.co_kwonlyargcount,
...     c.co_nlocals, c.co_stacksize, c.co_flags, newbytecode, c.co_
↪consts,
...     c.co_names, c.co_varnames, c.co_filename, c.co_name,
...     c.co_firstlineno, c.co_lnotab)
>>> nc
<code object add at 0x10069fe40, file "<stdin>", line 1>
>>> add.__code__ = nc
>>> add(2,3)
Segmentation fault
```

ä¡ăåŔŕäžěåČŔè£ŹæăůèĂ■åđ'ğæŃŻèőľ'èğčéĞŁåŹĺåěŤæžČãĂĆä¡ĘæŸŕïijŇåŕźäžŐçijŰåĘŹæŻť'énŸçžğäi
äžŰäžňåŔŕèČ¡çIJ§çŽĎéIJĂèęĄéĞ■åĘŹå■ŮèŁĆçăĄãĂĆæIJňèŁĆæIJĂåŘŐçŽĎéČĺåĹĘæijŤçđ'žäžĘè£ŹäÿłæŸ
this code on ActiveState

çňňå■ĄçńäïijžæĺąåİŮäÿŐåŇĚ

æĺąåİŮäÿŐåŇĚæŸŕäżżä¡Ţåđ'ğåđŃçĺŃåžŔçŽĎæäÿå£ČïijŇåřśè£đPythonåőL'èčĚçĺŃåžŔæIJňèžńäź§æŸŕ

Contents:

10.1 æđĎåżžäÿĂäÿłæĺąåİŮçŽĎåśĆçžğåŇĚ

éŮőécŸ

ä¡ăæČşåřĘä¡ăçŽĎäžčçăAçżĎçżĞæĹŘçŤśå¿Ĺåđ'ŽåĹĘåśĆæĺąåİŮæđĎæĹŘçŽĎåŇĚãĂĆ

èğčåĘşæŰźæąĹ

åřĄèčĚæĹŘåŇĚæŸŕå¿ĹçőĂå■ŢçŽĎãĂĆåIJĺæŰĞäżűçşżçż§äÿŁçżĎçżĞä¡ăçŽĎäžčçăĄïijŇåźűçąőä£Ìæŕ
ä¿ŃåęĆïijŽ

```
graphics/
    __init__.py
    primitive/
        __init__.py
        line.py
        fill.py
        text.py
    formats/
        __init__.py
        png.py
        jpg.py
```

äÿĂæŮęä¡ăåĄŽåĹřäžĘè£ŹäÿĂçĆźïijŇä¡ăåžŤèŕěèČ¡åď§æĽğèąŇåŘĎçğ■importèŕ■åŔěïijŇåęĆäÿŃïijŽ

```
import graphics.primitive.line
from graphics.primitive import line
import graphics.formats.jpg as jpg
```

èőÍèőž

åőŽäźĽæĺąåİŮçŽĎåśĆæňąçżŞæđĎåřšåČŔåIJĺæŰĞäžűçşżçż§äÿŁåžžçńŅçŽőå¡ŢçżŞæđĎäÿĂæăůåőźæŸ
æŰĞäžű__init__.pyçŽĎçŽőçŽĎæŸŕèęĄåŇĚåŘńäÿ■åŘŇè£ŘèąŇçžğåĹńçŽĎåŇĚçŽĎåŔŕéĂĽçŽĎåĹİåğŇåŇ
äÿ¿äÿłä¿Ňå■ŘïijŇåęĆæđIJä¡ăæĽğèąŇäžĘèŕ■åŔěimport graphicsïijŇ æŰĞäžűgraph-
ics/__init__.pyåřĘèćńåŕijåĚě,åżžçńŃgraphicsåŚ¡åŘ■çľžéŮť çŽĎåĘĚåőźãĂĆåČŔimport
graphics.format.jpgè£ŹæăůåŕijåĚěïijŇæŰĞäžűgraphics/__init__.pyåŠŇæŰĞäžűgraphics/graphics/formats/__

çżİåď ğéČĺåĹĘæŮűåĂŹèőľ__init__.pyçľžçİĂåřšåě¡ãĂĆä¡ĘæŸŕæIJL'äžŻæČĚåĘţäÿŃåŔŕèČ¡åŇĚåŘńäżč
äÿ¿äÿłä¿Ňå■ŘïijŇ__init__.pyèČ¡åď§çŤĺæİěèĞłåŁĺåŁăè¡¡å■ŘæĺąåİŮ:

```
# graphics/formats/__init__.py
from . import jpg
from . import png
```

åČŔè£ŹæăůäÿĂäÿłæŰĞäžű,çŤĺæĹůåŔŕäžěäżĚäžĚéĂŽè£Ğimport grah-
pics.formatsæİěäžčæŻ£import graphics.formats.jpgäžěåŔŁimport graphics.formats.pngãĂĆ

__init__.pyçŽĎåĚűäžŮåÿÿçŤĺçŤĺæşŢåŇĚæŃňåřĘåď ŽäÿłæŰĞäžűåŘĹåźűåĹřäÿĂäÿłéĂżè¿ŚåŚ¡åŘ■çľže

æŢŔéŤŘçŽĎçĺŃåžŘåŚŸäijŽåŘŚçŐřïijŇå■şä¡£æšąæIJL'__init__.pyæŰĞäžűå■ŸåIJĺïijŇpythonäż■çĎűä

10.2 æŐğåĹűæĺąåİŮèćńåĚĺéČĺåŕijåĚěçŽĎåĘĚåőź

éŮőécŸ

å¡Şä¡£çŤĺ'from module import *' èŕ■åŔěæŮűïijŇåÿŇæIJŽåŕźäžŐæĺąåİŮæĹŮåŇĚåŕijåĞžçŽĎçňęåŘůè£Z

èğčåĘşæŰźæąĹ

åIJĺä¡äçŽĎæĺąåİŮäÿ■åőŽäźĽäÿĂäÿłåŔŸéĞŔ __all__ æİěæŸŐçąőåIJřåĹŮåĞžéIJĂèęĄåŕijåĞžçŽĎåĘĚå

äÿ¿äÿłä¿Ńå■Ř:

```
# somemodule.py
def spam():
    pass

def grok():
    pass

blah = 42
# Only export 'spam' and 'grok'
__all__ = ['spam', 'grok']
```

èőĺèőž

åř¡çőąåijžçČĹåŔ■åŕźä¡£çŤĺ ‘from module import *‘, ä¡ĘæŸŕåIJĺåőŽäźĽäžĘåď ğéĞŔåŔŸéĞŔåŘ■çŽĎæĺąåİŮäÿ■écŚçźĄä¡£çŤĺãĂĆ åęĆæđIJä¡ăäÿ■åĄŽäżżä¡Ţäž Ń, è£ŹæăůçŽĎåŕijåĚěåřĘäijŽåŕijåĚěæĽĂæIJL’äÿ■äżěäÿŃåĹŠçž£åijĂåď’ţçŽĎã åŔęäÿĂæŰźéİć,åęĆæđIJåőŽäźĽäžĘ __all__ , éĆčäźĹåŔłæIJL’èćńåĹŮäÿ¿åĞžçŽĎäÿIJèě£äijŽèćńåŕijåĞžãĂĆ

åęĆæđIJä¡ăåřĘ __all__ åőŽäźĽ’æĹŘäÿĂäÿłçľžåĹŮèąĺ, æšąæIJL’äÿIJèě£åřĘèćńåŕijåĚěãĂĆ åęĆæđIJ __all__ åŇĚåŘńæIJłåőŽäźĽ’çŽĎåŘ■å■Ů, åIJĺåŕijåĚěæŰűåijŢètůAttributeErrorãĂĆ

10.3 ä¡£çŤĺçŻÿåŕźèůŕå¿ĎåŘ■åŕijåĚěåŇĚäÿ■å■ŘæĺąåİŮ

éŮőécŸ

åřĘäžčçăĄçżĎçżĞæĹŘåŇĚ,æČşçŤĺimportèŕ■åŔěäžŐåŔęäÿĂäÿłåŇĚåŘ■æšąæIJL’çąňçijŰçăĄè£ĞçŽĎå

èğčåĘşæŰźæąĹ

ä¡£çŤĺåŇĚçŽĎçŻÿåŕźåŕijåĚëijŇä¡£äÿĂäÿłçŽĎæĺąåİŮåŕijåĚěåŘŇäÿĂäÿłåŇĚçŽĎåŔęäÿĂäÿłæĺąåİŮ äÿ¿äÿłä¿Ńå■ŘëijŇåĄĞèő¿åIJĺä¡äçŽĎæŰĞäžűçşżçż§äÿŁæIJL’mypackageåŇĚëijŇçżĎçżĞåęĆäÿŇëijŽ

```
mypackage/
    __init__.py
    A/
        __init__.py
        spam.py
        grok.py
    B/
```

```
        __init__.py
        bar.py
```

åęĆæđIJæĺąåİŮmypackage.A.spamèęĄåŕijåĚěåŘŇçŻőå¡ŢäÿŃçŽĎæĺąåİŮgrokïijŇåőČåžŤèŕěåŇĚæŃňçŽ

```
# mypackage/A/spam.py
from . import grok
```

åęĆæđIJæĺąåİŮmypackage.A.spamèęĄåŕijåĚëäÿ■åŘŇçŻőå¡ŢäÿŃçŽĎæĺąåİŮB.barïijŇåőČåžŤèŕěä¡£çŤĺ

```
# mypackage/A/spam.py
from ..B import bar
```

äÿd'äÿłimportèŕ■åŔěéČ¡æšąåŇĚåŘńéąűåśĆåŇĚåŘ■ïijŇèĂŇæŸŕä¡£çŤĺäžĘspam.pyçŽĎçŻÿåŕźèůŕå¿ĎãĂ

èőÍèőž

åIJĺåŇĚåĘĚïijŇæŮćåŔŕäžěä¡£çŤĺçŻÿåŕźèůŕå¿Ďäź§åŔŕäžěä¡£çŤĺçżÌåŕźèůŕå¿ĎæÌěåŕijåĚěãĂĆ
äÿ¿äÿłä¿Ńå■ŘïijŽ

```
# mypackage/A/spam.py
from mypackage.A import grok # OK
from . import grok # OK
import grok # Error (not found)
```

åČŔmypackage.Aè£Źæăůä¡£çŤĺçżÌåŕźèůŕå¿ĎåŘ■çŽĎäÿ■åĹl'äźŃåd'ĎæŸŕè£ŹåřĘéąűåśĆåŇĚåŘ■çąñçijU
äÿ¿äÿłä¿Ńå■ŘïijŇåęĆæđIJä¡ăæŤźåŔŸäžĘåŇĚåŘ■ïijŇä¡ăåřśå£ĚéążæčĂæ§ěæL'ĂæIJL'æŰĞäžűæÌěä£őæ■čæ
åŘŇæăůïijŇçąñçijŰçăĄçŽĎåŘ■çğřäijŽä¡£çğżåŁĺäžčçăĄåŔŸå¿ŮåŽŕéŽ¿ãĂĆäÿ¿äÿłä¿Ńå■ŘïijŇäź§èőÿæIJL'ä
åęĆæđIJä¡£çŤĺçŻÿåŕźåŕijåĚěïijŇéĆčäÿĂåĹĞéČ¡okïijŇçĎűèĂŇä¡£çŤĺçżÌåŕźèůŕå¿ĎåŘ■å¿ĹåŔŕèČ¡äijŽåĞžéŮ

importèŕ■åŔěçŽĎ . åŠŇ "..''çIJŃètůæÌěå¿ĹæżŚçĺ¡,
ä¡ĘåőČæŇĞåőŽçŻőå¡ŢåŘ■.äÿžå¡ŞåL'■çŻőå¡ŢïijŇ..BäÿžçŻőå¡Ţ../BãĂĆè£Źçğ■èŕ■æşŢåŔłéĂĆçŤĺäžŐimport
äÿ¿äÿłä¿Ńå■ŘïijŽ

```
from . import grok # OK
import .grok # ERROR
```

åř¡çőąä¡£çŤĺçŻÿåŕźåŕijåĚěçIJŃètůæÌěåČŔæŸŕæţŔèğĹæŰĞäžűçşżçż§ïijŇä¡ĘæŸŕäÿ■èČ¡åĹřåőŽäźL'åŇĚ

æIJĂåŘŐïijŇçŻÿåŕźåŕijåĚěåŔłéĂĆçŤĺäžŐåIJĺåŘĹéĂĆçŽĎåŇĚäÿ■çŽĎæĺąåİŮãĂĆård'åĚűæŸŕåIJĺéąűås
ä¿ŃåęĆïijŽ

```
% python3 mypackage/A/spam.py # Relative imports fail
```

åŔęäÿĂæŰźéÍćïijŇåęĆæđIJä¡ăä¡£çŤÍPythonçŽĎ-méĂL'éąźæÌěæL'ğèąŇåĚĹåL'■çŽĎèĎŽæIJñïijŇçŻÿåŕ
ä¿ŃåęĆïijŽ

```
% python3 -m mypackage.A.spam # Relative imports work
```

æŻt'åd'ŽçŽĎåŇĚçŽĎçŻÿåŕźåŕijåĚěçŽĎèČŇæŹŕç§ěèŕE,èŕůçIJŃ PEP 328 .

10.4 åřĘæĺąåİŮåĹĘåL'šæĹŘåď'Žäÿ łæŰĞäžű

éŮőécŸ

ä¡ăæČşåřĘäÿĂäÿłæĺąåİŮåĹĘåL'šæĹŘåď'Žäÿ łæŰĞäžűãĂĆä¡ĘæŸŕä¡ăäÿ■æČşåřĘåĹĘçęžçŽĎæŰĞäžűçż

èğčåĘşæŰźæąĹ

çĺŃåžŔæĺąåİŮåŔŕäžěéĂŽè£ĞåŔŸæĹŘåŃĚæİěåĹĘåL'šæĹŘåď'Žäÿ łçŃňçńŃçŽĎæŰĞäžűãĂĆèĂČèŹŚäÿ

```
# mymodule.py
class A:
    def spam(self):
        print('A.spam')

class B(A):
    def bar(self):
        print('B.bar')
```

åĄĞèő¿ä¡ăæČşmymodule.pyåĹĘäÿžäÿď'äÿłæŰĞäžűĭijŇæŕŔäÿłåőŽäźL'çŽĎäÿĂäÿłçśžãĂĆèęĄåĄŽåĹřè
è£Źè£ŹäÿłçŽőå¡ŢäÿŃĭijŇåĹŻåžžäžěäÿŃæŰĞäžűĭijŽ

```
mymodule/
    __init__.py
    a.py
    b.py
```

åIJĺa.pyæŰĞäžűäÿ■æŔŠåĚěäžěäÿŃäžčçăĄĭijŽ

```
# a.py
class A:
    def spam(self):
        print('A.spam')
```

åIJĺb.pyæŰĞäžűäÿ■æŔŠåĚěäžěäÿŃäžčçăĄĭijŽ

```
# b.py
from .a import A
class B(A):
    def bar(self):
        print('B.bar')
```

æIJĂåŘŐĭijŇåIJĺ __init__.py äÿ■ĭijŇářĘ2äÿłæŰĞäžűçšŸåŘĹåIJĺäÿĂèţůĭijŽ

```
# __init__.py
from .a import A
from .b import B
```

åęĆædIJæŇL'çĚğè£ŹäžŻæ■ěéłď'ĭijŇæL'Ăäžğç Ť§çŽĎåŇĚMyModuleářĘä¡IJäÿžäÿĂäÿłå■ŢäÿĂçŽĎéĂż

```
>>> import mymodule
>>> a = mymodule.A()
>>> a.spam()
A.spam
>>> b = mymodule.B()
>>> b.bar()
B.bar
>>>
```

èőÍèőž

åIJÍè£ŹäýłçńäèŁĆäý■çŽĎäýżèęĄéŮőécŸæŸŕäýĂäýłèő¿èőąéŮőécŸïijŇäý■çőąä¡ăæŸŕåŘęåýŇæIJŻçŤÍæ

```
from mymodule.a import A
from mymodule.b import B
...
```

è£ŹæăůèČ¡åůëä¡IJïijŇä¡Ęè£ŹèőľçŤÍæŁůæL'£åŔŮæŻť'åd'ŽçŽĎèť§æŃĚïijŇçŤÍæŁůèęĄç§ěéĄŞäý■åŘŇ

```
from mymodule import A, B
```

åŕźåŘŐèĂĚèĂŇèÍĂïijŇèőľ mymoduleæŁŘäýžäýĂäýłåd'ğçŽĎæžŘæŰĞäżűæŸŕæIJĂåýÿèğĄçŽĎãĂĆä¡Ì
è£ŹæăůåĄŽçŽĎåĚşéŤőæŸŕåŁŻåżžäýĂäýłåŇĚçŻőå¡ŢïijŇä¡£çŤÍ __init__.py
æŰĞäżűæİěářĘæŕŔéČÍåŁĘçšŸåŘŁåIJÍäýĂèţůãĂĆ

å¡ŞäýĂäýłæÍąåİŮèćńåŁĘåL'šïijŇä¡äéIJĂèęĄçL'źåĹńæşÍæĎŔäžd'åŔL'åijŢçŤÍçŽĎæŰĞäżűåŘ■ãĂĆäÿ¿äý
from .a import A æİěèŐůåŔŰãĂĆ

æŢt'äýłçńäèŁĆéČ¡ä¡£çŤÍåŇĚçŽĎçŻÿåŕźåŕijåĚěæİěéĄ£åĚ■åřĘéąűåśĆæÍąåİŮåŘ■çąňçijŰçăĄåĹřæžŘäżč

ä¡IJäÿžè£ŹäÿĂçńäèŁĆçŽĎåžűäijÿïijŇåřĘäżŃçż■åžűè£§åŕijåĚěãĂĆåęĆåŻ¿æL'Äçd'žïijŇ__init__.pyæŰ
èęĄåĄŽåĹřè£ŹäÿĂçĆźïijŇ__init__.pyæIJL'çżĘå¿őçŽĎåŔŸåŇŰïijŽ

```
# __init__.py
def A():
    from .a import A
    return A()

def B():
    from .b import B
    return B()
```

åIJÍè£ŹäýłçL'ĹæIJñäý■ïijŇçśżAåŠŇçśżBècńæŻ£æ■ćäýžåIJÍçňňäýĂæňąèő£éŮőæŮűåŁăè¡¡æL'ĂéIJĂçŽĎ
ä¿ŃåęĆïijŽ

```
>>> import mymodule
>>> a = mymodule.A()
>>> a.spam()
A.spam
>>>
```

åžűè£§åŁäè¡¡çŽĎäÿžèęAçijžçĆźæŸŕçżğæL'£åŠŇçśžåđŃæčĂæ§ěåŘŕèČ¡äijŽäÿ■æŰ■ãĂĆä¡ăåŘŕèČ¡äijŽ

```
if isinstance(x, mymodule.A): # Error
...

if isinstance(x, mymodule.a.A): # Ok
...
```

åžűè£§åŁäè¡¡çŽĎçIJ§åőđä¿Ńå■Ř, èğĄæăĞåĞĘåžŞ multiprocessing/__init__.py çŽĎæžŘçăĄ.

10.5 åĹľçŤĺåŚ¡åŘ■çľžéŮť'åŕijåĚěçŻőå¡ŢåĹĘæŢčçŽĎäżčçăĄ

éŮőécŸ

ä¡ăåŘŕèČ¡æIJL'åď'ğéĞŘçŽĎäżčçăĄïijŇçŤśäÿ■åŘŇçŽĎäžžæİěåĹĘæŢčåIJřçżť'æŁd'ãĂĆæŕŘäÿłéČĺåĹĘè

èğčåĘşæŰźæąĹ

äžŐæIJñèť'ĺäÿŁèőšïijŇä¡ăèęĄåőŽäźL'äÿĂäÿłéąűçžğPythonåŇĚïijŇä¡IJäÿžäÿĂäÿłåď'ğéŻĘåŘĹåĹĘåijĂç

åIJÍçż§äÿĂäÿ■åŘŇçŽĎçŻőå¡ŢéĞŇçż§äÿĂçŻÿåŘŇçŽĎåŚ¡åŘ■çľžéŮť'ïijŇä¡ĘæŸŕèęĄåĹăăŐżçŤĺæİěåřĘ

```
foo-package/
    spam/
        blah.py

bar-package/
    spam/
        grok.py
```

åIJÍè£Ź2äÿłçŻőå¡ŢéĞŇïijŇéČ¡æIJL'çİĂåĚśåŘŇçŽĎåŚ¡åŘ■çľžéŮť'spamãĂĆåIJÍäżżä¡ŢäÿĂäÿłçŻőå¡ŢéĞ

èőľ'æĹŚäžňçIJŃçIJŃïijŇåęĆæđIJåřĘfoo-packageåŠŇbar-packageéČ¡åŁăĺŕpythonæĺąåİŮèůŕå¿ĎåźűåřİèŕŢåŕijåĚěäijŽåŘŚçŤ§äżĂäźĹ

```
>>> import sys
>>> sys.path.extend(['foo-package', 'bar-package'])
>>> import spam.blah
>>> import spam.grok
>>>
```

äÿd'äÿłäÿ■åŘŇçŽĎåŇĚçŻőå¡ŢèćńåŕĺåźűåĹřäÿĂèţůïijŇä¡ăåŘŕäżěåŕijåĚěspam.blahåŠŇspam.grokïijŇå

èőÍèőž

åIJÍè£ŹéĞŇåůěä¡IJçŽĎæIJžåĹűèćńçğřäÿžâĂIJåŇĚåŚ¡åŘ■çľžéŮť'âĂİçŽĎäÿĂäÿłçL'źå¿ĄãĂĆäżŐæIJñè

åŇĚåŚ¡åŘ■çľžéŮťçŽĎåĚşéŤőæŸŕçąőä£ÌéąűçžğçŻőå¡Ţäÿ■æšąæIJL’__init__.pyæŰĞäżűæİěä¡IJäÿžåĚśäÿ¿äÿłä¿Ńå■ŘïijŽ

```
>>> import spam
>>> spam.__path__
_NamespacePath(['foo-package/spam', 'bar-package/spam'])
>>>
```

åIJÍåőŽä¡■åŇĚçŽĎå■ŘçżĎäžűæŮűïijŇçŻőå¡Ţ__path__åřĘèćńçŤÍåĹř(ä¿ŃåęĆ, å¡Şåŕijå Ěěspam.grokæĹŰèĂĚspam.blahçŽĎæŮűåĂŹ).

åŇĚåŚ¡åŘ■çľžéŮťçŽĎäÿĂäÿłéĞ■èęĄçL’źçĆźæŸŕäżżä¡ŢäžžéČ¡åŔŕäżěçŤÍèĞłåůśçŽĎäžčçăĄæİěæL’ľåś

```
my-package/
    spam/
        custom.py
```

åęĆæđIJä¡ăářĘä¡ăçŽĎäžčçăĄçŻőå¡ŢåŠŇåĚűäżŰåŇĚäÿĂèţůæůżåŁăåĹřsys.pathïijŇè£ŹářĘæŮăçijİåIJřå

```
>>> import spam.custom
>>> import spam.grok
>>> import spam.blah
>>>
```

äÿĂäÿłåŇĚæŸŕåŘęèćńä¡IJäÿžäÿĂäÿłåŇĚåŚ¡åŘ■çľžéŮťçŽĎäÿżèęĄæŰźæşŢæŸŕæčĂæ§ěåĚű__file__åś

```
>>> spam.__file__
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
AttributeError: 'module' object has no attribute '__file__'
>>> spam
<module 'spam' (namespace)>
>>>
```

æŻťåď ŽçŽĎåŇĚåŚ¡åŘ■çľžéŮťä£ąæĄŕåŔŕäżěæ§ěçIJŃ PEP 420.

10.6 éĞ■æŰřåŁăè¡¡æĺąåİŮ

éŮőécŸ

ä¡ăæČşéĞ■æŰřåŁăè¡¡åůšçżŔåŁăè¡¡çŽĎæĺąåİŮïijŇåŻăäÿžä¡ăářźåĚűæžŘçăĄè£ŻèąŇäžĘä£őæŤźãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤÍimp.reload()æİěéĞ■æŰřåŁăè¡¡åĚĹåL’■åŁăè¡¡çŽĎæĺąåİŮãĂĆäÿ¿äÿłä¿Ńå■ŘïijŽ

```
>>> import spam
>>> import imp
>>> imp.reload(spam)
```

```
<module 'spam' from './spam.py'>
>>>
```

èőÍèőž

éĞ■æŰřåŁäè¡¡æÍąåİŮåIJÍåijĂåŔŠåŠŇèřČèŕŢè£ĞçÍŃäÿ■åÿÿåÿÿå¿ŁæIJL'çŤÍãĂĆä¡ĘåIJÍçŤ§äžğçŐŕåćČä

reload()æŞęéŹđ'äžĘæÍąåİŮåžŢåśĆå■ŮåĚÿçŽĎåĘĚåőźïijŇåźűéĂŽè£ĞéĞ■æŰřæL'ğèąŇæÍąåİŮçŽĎæžŘ

åř¡çőąåęĆæ■đ'ïijŇreload()æšąæIJL'æŻt'æŰřåČŔ"from module import name"è£ŹæăůäĄ£çŤÍimportèŕ■åŘěåŕijåĚěçŽĎåőŽäźL'ãĂĆäÿ¿äÿłä¿Ńå■ŘïijŽ

```
# spam.py
def bar():
    print('bar')

def grok():
    print('grok')
```

çŐřåIJÍåŘŕåŁĺäžđ'äžŠåijŔäijŽèŕİïijŽ

```
>>> import spam
>>> from spam import grok
>>> spam.bar()
bar
>>> grok()
grok
>>>
```

äÿ■éĂĂåĞžPythonä£őæŤźspam.pyçŽĎæžŘçăĄïijŇářĘgrok()åĞ¡æŢřæŤźæĹŘè£Źæăůïijž

```
def grok():
    print('New grok')
```

çŐřåIJÍåŻđåĹřäžđ'äžŠåijŔäijŽèŕİïijŇéĞ■æŰřåŁäè¡¡æÍąåİŮïijŇářİèŕŢäÿŃè£Źäÿłåőđéł№ïijž

```
>>> import imp
>>> imp.reload(spam)
<module 'spam' from './spam.py'>
>>> spam.bar()
bar
>>> grok() # Notice old output
grok
>>> spam.grok() # Notice new output
New grok
>>>
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇä¡ăçIJŃåĹřæIJL'2äÿłçL'ĹæIJńçŽĎgrok()åĞ¡æŢřèćńåŁäè¡¡ãĂĆéĂŽåÿÿæİèèŕť'ï

åŻăæ■đ'ïijŇåIJÍçŤ§äžğçŐŕåćČäÿ■åŘŕèČ¡éIJĂèęĄéĄ£åĚ■éĞ■æŰřåŁäè¡¡æÍąåİŮãĂĆåIJÍäžđ'äžŠçŐŕåćČ

10.7 è£Řèą ŇçŻőå¡ŢæĹŰåŐŃçijľæŰĞäžű

éŮőécŸ

æĆĺæIJL'äÿĂäÿłåůšæĹŘéŢ£äÿžåŇĚåŘńåď'ŽäÿłæŰĞäžűçŽĎåžŤçŤĺïijŇåőČåůšè£IJäÿ■åĘ■æŸŕäÿĂäÿłçő

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăçŽĎåžŤçŤĺçĺŃåžŔåůšçżŔæIJL'åď'ŽäÿłæŰĞäžűïijŇä¡ăåŔŕäžěæŁŁä¡ăçŽĎåžŤçŤĺçĺŃåžŔæŤ¿äÿ¿äÿłä¿Ńå■ŘïijŇä¡ăåŔŕäžěåČŔè£Źæăůå ĹŻåžžçŻőå¡Ţïij Ž

```
myapplication/
    spam.py
    bar.py
    grok.py
    __main__.py
```

åęĆædIJ__main__.pyå■ŸåIJĺïijŇä¡ăåŔŕäžěçőĂå■ŢåIJřåIJĺéąűçžğçŽőå¡Ţè£ŘèąŇPythonèğčéĞŁåŹĺïijŽ

```
bash % python3 myapplication
```

èğčéĞŁåŹĺåřĘæL'ğèąŇ__main__.pyæŰĞäžűä¡IJäÿžäÿżçĺŃåžŔãĂĆ

åęĆædIJä¡ăåřĘä¡ăçŽĎäžčçăĄæL'ŞåŇĚæĹŘzipæŰĞäžűïijŇè£Źçğ■æŁĂæIJŕåŘŇæăůäź§éĂĆçŤĺïijŇäÿ¿ă

```
bash % ls
spam.py bar.py grok.py __main__.py
bash % zip -r myapp.zip *.py
bash % python3 myapp.zip
... output from __main__.py ...
```

èőÍèőž

åĹŻåžžäÿĂäÿłçŽőå¡ŢæĹŰzipæŰĞäžűåźűæůžåŁă__main__.pyæŰĞäžűæİěåřĘäÿĂäÿłæŻť'åď'ğçŽĎPytho

çŤśäžŐçŽőå¡ŢåŠŇzipæŰĞäžűäÿÔæ■čåÿÿæŰĞäžűæIJL'äÿĂçĆźäÿ■åŘŇïijŇä¡ăåŔŕèČ¡è£ŸéIJĂèęĄåćđå

```
#!/usr/bin/env python3 /usr/local/bin/myapp.zip
```

10.8 èŕżåŔŰä¡■äžŐåŇĚäÿ■çŽĎæŢřæ■őæŰĞäžű

éŮőécŸ

ä¡ăçŽĎåŇĚäÿ■åŇĚåŘńäžčçăĄéIJĂèęĄåŐżèŕżåŔŰçŽĎæŢřæ■őæŰĞäžűãĂĆä¡ăéIJĂèęĄåř¡åŔŕèČ¡åIJřçŤ

èğčåĘşæŰźæąĹ

åĄĞèő¿ä¡ăçŽĎåŇĚäÿ■çŽĎæŰĞäžűçżĎçżĞæĹŘåęĆäÿŃïijŽ

```
mypackage/
    __init__.py
    somedata.dat
    spam.py
```

çŐřåIJĺåĄĞèő¿spam.pyæŰĞäžűéIJĂèęĄèŕžåŔŰsomedata.datæŰĞäžűäÿ■çŽĎåĘĚåőźãĂĆä¡ăåŔŕäžěçŤĺ

```
# spam.py
import pkgutil
data = pkgutil.get_data(__package__, 'somedata.dat')
```

çŤśæ■ď'äžğçŤ§çŽĎåŔŸéĞŔæŸŕåŇĚåŘńèřěæŰĞäžűçŽĎåŐ§ågŃåĘĚåőźçŽĎå■ŮèŁĆå■Ůçňęäÿšã ĂĆ

èőĺèőž

èęĄèŕžåŔŰæŢřæ■őæŰĞäžűïijŇä¡ăåŔŕèČ¡äijŽåĂ¿åŘŚäžŐçijŰåĘŹä¡£çŤĺåĘĚç¡őçŽĎI/
OåŁ§èČ¡çŽĎäžčçăĄïijŇåęĆopen()ãĂĆä¡ĘæŸŕè£Źçğ■æŰźæşŢäź§æIJL'äÿĂäžŻéŮőécŸãĂĆ

éęŰåĚĹïijŇäÿĂäÿłåŇĚåŕźèğčéĞŁåŹĺçŽĎå¡ŞåL'■åůëä¡IJçŻőå¡ŢåĞăäźŐæšąæIJL'æŐğåĹűæİČãĂĆåŻăæ

çňňäžŇïijŇåŇĚéĂŽåÿÿåőL'èčĚä¡IJäÿž.zipæĹŰ.eggæŰĞäžűïijŇè£ŹäžŻæŰĞäžűåźűäÿ■åČŔåIJĺæŰĞäžű

pkgutil.get_data()åĞ¡æŢřæŸŕäÿĂäÿłèŕžåŔŰæŢřæ■őæŰĞäžűçŽĎéńŸçžğåůěåĚůïijŇäÿ■çŤĺçőąåŇĚæŸŕå

get_data()çŽĎçňňäÿĂäÿłåŔĆæŢřæŸŕåŇĚåŘńåŇĚåŘ■çŽĎå■Ůçňęäÿšã ĂĆä¡ăåŔŕäžěçŻť'æŐěä¡£çŤĺåŇĚ

10.9 åřĘæŰĞäžűåď'źåŁäåĚěåĹřsys.path

éŮőécŸ

ä¡ăæŮăæşŢåŕijåĚěä¡ăçŽĎPythonäžčçăĄåŻăäÿžåőĈæL'ĂåIJĺçŽĎçŻőå¡ŢäÿåIJĺsys.pathéĞŇãĂĆä¡ăæĈş

èğčåĘşæŰźæąĹ

æIJL'äÿď'çğ■åÿÿçŤĺçŽĎæŰźåijŔåřĘæŰřçŻőå¡ŢæůžåŁäå Ĺřsys.pathãĂĆçňňäÿĂçğ■ïijŇä¡ăåŔŕäžěä¡£çŤĺ

```
bash % env PYTHONPATH=/some/dir:/other/dir python3
Python 3.3.0 (default, Oct 4 2012, 10:17:33)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more␣
 ↪information.
>>> import sys
>>> sys.path
['', '/some/dir', '/other/dir', ...]
>>>
```

åIJÍèĞłåőŽäźL’åžŤçŤĺçÍŃåžŔäy■ïijŇè£Źæăůç ŽĎçŐŕåćČåŔŸéĞŔåŔŕåIJÍçÍŃåžŔåŘŕåŁÍæŮűèő¿ç¡őæĹŰ

çňňäžŇçğ■æŰźæşŢæŸŕåĹŻåžžäyĂäył.pthæŰĞäžűïijŇářĘçŻőå¡ŢåĹŮäÿ¿åĞžæİěïijŇåČŔè£ŹæăůïijŽ

```
# myapplication.pth
/some/dir
/other/dir
```

è£Źäył.pthæŰĞäžűéIJĂèęĄæŤ¿åIJÍæ§ŘäyłPythonçŽĎsite-packagesçŻőå¡ŢïijŇéĂŽåÿÿä¡■äžŐ/usr/local/lib/python3.3/site-packages æĹŰèĂĚ ~/.local/lib/python3.3/sitepackagesãĂĆå¡Şèğčé ĞŁåŹĺåŘŕåŁÍæŮűïijŇ.pthæŰĞäžűéĞŇåĹŮäÿ¿åĞžæİęçŽĎå■ŸåIJ

èőÍèőž

æŕŤèţůèť źåŁŻåIJřæL’¿æŰĞäžűïijŇä¡ăåŔŕèČ¡äijŽåĂ¿åŘŠäžŐåĘŹäÿĂäyłäžčçăĄæL’ŃåŁÍèřČèŁĆsys.pat

```
import sys
sys.path.insert(0, '/some/dir')
sys.path.insert(0, '/other/dir')
```

èŹ¡çĎűè£ŹèČ¡âĂIJåůëä¡IJâĂİïijŇáőČæŸŕåIJÍáőđèůţäÿ■æđĄäÿžèĎĘåijśïijŇážŤář¡éĞŔéĄ£åĚ■ä¡£çŤÍãĂ

```
import sys
from os.path import abspath, join, dirname
sys.path.insert(0, join(abspath(dirname(__file__)), 'src'))
```

è£ŹářĘsrcçŻőå¡ŢæůžåŁăåĹřpathéĞŇïijŇåŠŇæL’ğèąŇæŔŠåĚëæ■ěéłd’çŽĎäžčçăĄåIJÍåŘŇäyĂäyłçŻőå¡

site-packagesçŻőå¡ŢæŸŕçňňäyL’æŰźåŇĚåŠŇæÍąåİŮåőL’èčĚçŽĎçŻőå¡ŢãĂĆåęĆæđIJä¡ăæL’ŃåŁÍåőL’èč packagesçŻőå¡ŢãĂĆèŹ¡çĎűçŤÍäžŐéĚ■ç¡őpathçŽĎ.pthæŰĞäžűå£ĚéąžæŤ¿ç¡őåIJÍsite-packageséĞŇïijŇä¡ĘåőČéĚ■ç¡őçŽĎèůŕå¿ĎåŔŕäžěæŸŕçşżçż§äyŁäžžä¡Ţä¡ăåÿŇæIJŻçŽĎçŻőå¡ŢãĂĆåŽăæ■d’

10.10 éĂŽè£Ğå■Ůçňęäÿšå Ř■åŕijåĚěæÍąåİŮ

éŮőécŸ

ä¡ăæČşåŕijåĚěäyĂäyłæÍąåİŮïijŇä¡ĘæŸŕæÍąåİŮçŽĎåŘ■å■ŮåIJÍå■Ůçňęäyšé ĞŇãĂĆä¡ăæČşåŕźå■Ůçňęäÿ

èğčåĘşæŰźæąĹ

ä¡£çŤÍimportlib.import_module()åĞ¡æŢřæİěæL’ŃåŁÍåŕijåĚěåŘ■å■Ůäÿžå■ŮçňęäyšçżźåĞžçŽĎäyĂäyłæ

```
>>> import importlib
>>> math = importlib.import_module('math')
>>> math.sin(2)
0.9092974268256817
```

```
>>> mod = importlib.import_module('urllib.request')
>>> u = mod.urlopen('http://www.python.org')
>>>
```

import_moduleåŔłæŸŕçőĂå■ŢåIJřæĽğèąŇåŠŇimportçŻÿåŘŇçŽĎæ■ěéłd'ïijŇä¡ĘæŸŕè£ŤåŻđçŤ§æĹŘç

åęĆæđIJä¡ăæ■čåIJĺä¡£çŤĺçŽĎåŇĚïijŇimport_module()äź§åŔŕçŤĺäžŐçŻÿåŕźåŕijåĚěãĂĆä¡ĘæŸŕïijŇä¡äé

```
import importlib
# Same as 'from . import b'
b = importlib.import_module('.b', __package__)
```

èőĺèőž

ä¡£çŤĺimport_module()æĽŃåŁĺåŕijåĚěæĺąåİŮçŽĎéŮőécŸéĂŽåÿÿåĞžçŐřåIJĺäžěæ§Řçğ■æŰźåijŔçijŰå

åIJĺæŮğçŽĎäžčçăĄïijŇæIJL'æŮűä¡ääijŽçIJŃåĹřçŤĺäžŐåŕijåĚěçŽĎåĘĚåžžåĞ¡æŢř__import__()ãĂĆår¡ç
éĂŽåÿÿæŻt'åőźæŸŞä¡£çŤĺãĂĆ

èĞłåőŽäźL'åŕijåĚěè£ĞçĺŃçŽĎéńŸçžğåődä¿ŃèğĄ10.11årŔèŁĆ

10.11 éĂŽè£ĞéŠľå■Řè£IJçĺŃåŁăè¡¡æĺąåİŮ

éŮőécŸ

ä¡ăæČşèĞłåőŽäźL'PythonçŽĎimportèŕ■åŔěïijŇä¡£å¿ŮåőČèČ¡äžŐè£IJçĺŃæIJžåŹĺäÿŁéİćéĂŔæŸŐçŽĎå

èğčåĘşæŰźæąĹ

éęŰåĚĹèęĄæŔŘåĞžæİěçŽĎæŸŕåőL'åĚĺéŮőécŸãĂĆæIJňèŁĆèőĺèőžçŽĎæĂİæČşåęĆæđIJæšąæIJL'äÿĂ
äź§åřśæŸŕèŕť'ïijŇæĹŚäžňçŽĎäÿżèęĄçŽőçŽĎæŸŕæůśåĚěåĹĘæđŘPythonçŽĎimportèŕ■åŔěæIJžåĹűãĂĆ
åęĆæđIJä¡ăçŘĘèğčäžĘæIJňèŁĆåĘĚéČĺåŐ§çŘĘïijŇä¡ăåřśèČ¡åd'§äÿžåĚűäžŰäžžä¡ŢçŽőçŽĎèĂŇèĞłåőŽäźL'i
æIJL'äžĘè£ŹäžŻïijŇèől'æĹŚäžňçżğçż■åŘŚåL'■èţřãĂĆ

æIJňèŁĆæäÿå£ČæŸŕèő¿èőąåŕijåĚěèŕ■åŔěçŽĎæL'l'åśŢåŁ§èČ¡ãĂĆæIJL'å¿Ĺåd'Žçğ■æŰźæşŢåŔŕäžěåĄŽ
äÿ■è£ĞäÿžäžĘæijŤçd'žçŽĎæŰźä¿£ïijŇæĹŚäžňåijĂåğŃåĚĹædĎéĂääÿŃéİćè£ŹäÿłPythonäžčçăĄçżŞædĎïijŽ

```
testcode/
    spam.py
    fib.py
    grok/
        __init__.py
        blah.py
```

è£ŹäžŻæŰĞäžűçŽĎåĘĚåőźåźűäÿ■éĞ■èęĄïijŇäÿ■è£ĞæĹŚäžňåIJĺæŕŘäÿłæŰĞäžűäÿ■æŤ¿åĚěäžĘåřŚéĞ
è£Źæăůä¡ăåŔŕäžěæţŃèŕŢåőČäžňåźűæ§ěçIJŃå¡ŞåőČäžňèćńåŕijåĚěæŰűçŽĎè¿ŞåĞžãĂĆä¿ŃåęĆïijŽ

```
# spam.py
print("I'm spam")

def hello(name):
    print('Hello %s' % name)

# fib.py
print("I'm fib")

def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n-1) + fib(n-2)

# grok/__init__.py
print("I'm grok.__init__")

# grok/blah.py
print("I'm grok.blah")
```

è£ŹéĞŇçŽĎçŽőçŽĎæŸŕåĚĄèőÿè£ŹäžŽæŰĞäżűä¡IJäÿžæÍąåİŮèćńè£IJçÍŃèő£éŮőãĂĆäźş§èőÿæIJĂçőĂå■ŢçŽĎæŰźåijŔåřśæŸŕåřĘåőČäžňåŔŠåÿČåĹřäÿĂäÿłwebæIJ■åŁąåŹÍäÿŁéİćãĂĆåIJÍtestcode

```
bash % cd testcode
bash % python3 -m http.server 15000
Serving HTTP on 0.0.0.0 port 15000 ...
```

æIJ■åŁąåŹÍè£ŘèąŇètůæİěåŘŐåĘ■åŘŕåŁÍäÿĂäÿłå■ŢçŃňçŽĎPythonèğčéĞŁåŹÍãĂĆçąőä£İä¡ăåŔŕäžěä¡£çŤÍ `urllib` èő£éŮőåĹře£IJçÍŃæŰĞäżűãĂĆä¿ŃåęĆïijŽ

```
>>> from urllib.request import urlopen
>>> u = urlopen('http://localhost:15000/fib.py')
>>> data = u.read().decode('utf-8')
>>> print(data)
# fib.py
print("I'm fib")

def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n-1) + fib(n-2)
>>>
```

äžŐè£ŹäÿłæIJ■åŁąåŹÍåŁăè¡¡æžŘäžčçăĄæŸŕæŐěäÿŃæİěæIJňèŁĆçŽĎå§žçąĂãĂĆäÿžäžĘæŻ£äžčæL'ŃåŁÍçŽĎéĂŽè£Ğ `urlopen()` æİěæŤűéŻĘæžŘæŰĞäżűïijŇæĹŚäžňéĂŽè£ĞèĞłåőŽäźL'importèŕ■åŘěæİěåIJÍåŘŐåŘŕèĞłåŁÍåÿőæĹŚäžňåĄŽåĹřãĂĆ

åŁăè¡¡è£IJçÍŃæÍąåİŮçŽĎçňňäÿĂçğ■æŰźæşŢæŸŕåĹŻåžžäÿĂäÿłæŸ¿çd'žçŽĎåŁăè¡¡åĞ¡æŢřæİěåőŇæĹŘ

```
import imp
import urllib.request
import sys

def load_module(url):
    u = urllib.request.urlopen(url)
    source = u.read().decode('utf-8')
    mod = sys.modules.setdefault(url, imp.new_module(url))
    code = compile(source, url, 'exec')
    mod.__file__ = url
    mod.__package__ = ''
    exec(code, mod.__dict__)
    return mod
```

è£Źäÿłå Ğ¡æŢřäijŽäÿŃè¡¡æžŘäžčçăĄïijŇåźűä¡£çŤÍ compile() åřĘåĚűçijŰèŕŠåĽřäÿĂäÿłäžčçăĄåŕźèśąäÿ■ïijŇ çĎűåŘŐåIJÍäÿĂäÿłæŰřåĹŻåžžçŽĎæÍąåİŮåŕźèśąçŽĎå■ŮåĚÿä

```
>>> fib = load_module('http://localhost:15000/fib.py')
I'm fib
>>> fib.fib(10)
89
>>> spam = load_module('http://localhost:15000/spam.py')
I'm spam
>>> spam.hello('Guido')
Hello Guido
>>> fib
<module 'http://localhost:15000/fib.py' from 'http://
 ↪localhost:15000/fib.py'>
>>> spam
<module 'http://localhost:15000/spam.py' from 'http://
 ↪localhost:15000/spam.py'>
>>>
```

æ■čåęĆä¡ăæĽĂèğĄïijŇåŕźäžŐçőĂå■ŢçŽĎæÍąåİŮè£ŹäÿłæŸŕèąŇå¿ŮéĂŽçŽĎãĂĆ äÿ■è£ĞåőČåźűæšąæIJĽåţŇåĚěåĹřéĂŽåÿÿçŽĎimportèŕ■åŘěäÿ■ïijŇåęĆæđIJèęĄæŤŕæŇĄæŻť'énŸçžğçŽĎçż

äÿĂäÿłæŻť'éĚůçŽĎåĄŽæşŢæŸŕåĹŻåžžäÿĂäÿłèĞłåőŽäźĽ'åŕijåĚěåŹÍãĂĆçňňäÿĂçğ■æŰźæşŢæŸŕåĹŻåž

```
# urlimport.py
import sys
import importlib.abc
import imp
from urllib.request import urlopen
from urllib.error import HTTPError, URLError
from html.parser import HTMLParser

# Debugging
import logging
log = logging.getLogger(__name__)

# Get links from a given URL
```

```
def _get_links(url):
    class LinkParser(HTMLParser):
        def handle_starttag(self, tag, attrs):
            if tag == 'a':
                attrs = dict(attrs)
                links.add(attrs.get('href').rstrip('/'))
    links = set()
    try:
        log.debug('Getting links from %s' % url)
        u = urlopen(url)
        parser = LinkParser()
        parser.feed(u.read().decode('utf-8'))
    except Exception as e:
        log.debug('Could not get links. %s', e)
    log.debug('links: %r', links)
    return links

class UrlMetaFinder(importlib.abc.MetaPathFinder):
    def __init__(self, baseurl):
        self._baseurl = baseurl
        self._links = { }
        self._loaders = { baseurl : UrlModuleLoader(baseurl) }

    def find_module(self, fullname, path=None):
        log.debug('find_module: fullname=%r, path=%r', fullname,␣
↪path)
        if path is None:
            baseurl = self._baseurl
        else:
            if not path[0].startswith(self._baseurl):
                return None
            baseurl = path[0]
        parts = fullname.split('.')
        basename = parts[-1]
        log.debug('find_module: baseurl=%r, basename=%r', baseurl,␣
↪basename)

        # Check link cache
        if basename not in self._links:
            self._links[baseurl] = _get_links(baseurl)

        # Check if it's a package
        if basename in self._links[baseurl]:
            log.debug('find_module: trying package %r', fullname)
            fullurl = self._baseurl + '/' + basename
            # Attempt to load the package (which accesses __init__.
↪py)
            loader = UrlPackageLoader(fullurl)
            try:
                loader.load_module(fullname)
```

```
                self._links[fullurl] = _get_links(fullurl)
                self._loaders[fullurl] = UrlModuleLoader(fullurl)
                log.debug('find_module: package %r loaded',␣
↪fullname)
            except ImportError as e:
                log.debug('find_module: package failed. %s', e)
                loader = None
            return loader
        # A normal module
        filename = basename + '.py'
        if filename in self._links[baseurl]:
            log.debug('find_module: module %r found', fullname)
            return self._loaders[baseurl]
        else:
            log.debug('find_module: module %r not found', fullname)
            return None

    def invalidate_caches(self):
        log.debug('invalidating link cache')
        self._links.clear()

# Module Loader for a URL
class UrlModuleLoader(importlib.abc.SourceLoader):
    def __init__(self, baseurl):
        self._baseurl = baseurl
        self._source_cache = {}

    def module_repr(self, module):
        return '<urlmodule %r from %r>' % (module.__name__, module._
↪_file__)

    # Required method
    def load_module(self, fullname):
        code = self.get_code(fullname)
        mod = sys.modules.setdefault(fullname, imp.new_
↪module(fullname))
        mod.__file__ = self.get_filename(fullname)
        mod.__loader__ = self
        mod.__package__ = fullname.rpartition('.')[0]
        exec(code, mod.__dict__)
        return mod

    # Optional extensions
    def get_code(self, fullname):
        src = self.get_source(fullname)
        return compile(src, self.get_filename(fullname), 'exec')

    def get_data(self, path):
        pass
```

```
    def get_filename(self, fullname):
        return self._baseurl + '/' + fullname.split('.')[-1] + '.py'

    def get_source(self, fullname):
        filename = self.get_filename(fullname)
        log.debug('loader: reading %r', filename)
        if filename in self._source_cache:
            log.debug('loader: cached %r', filename)
            return self._source_cache[filename]
        try:
            u = urlopen(filename)
            source = u.read().decode('utf-8')
            log.debug('loader: %r loaded', filename)
            self._source_cache[filename] = source
            return source
        except (HTTPError, URLError) as e:
            log.debug('loader: %r failed. %s', filename, e)
            raise ImportError("Can't load %s" % filename)

    def is_package(self, fullname):
        return False

# Package loader for a URL
class UrlPackageLoader(UrlModuleLoader):
    def load_module(self, fullname):
        mod = super().load_module(fullname)
        mod.__path__ = [ self._baseurl ]
        mod.__package__ = fullname

    def get_filename(self, fullname):
        return self._baseurl + '/' + '__init__.py'

    def is_package(self, fullname):
        return True

# Utility functions for installing/uninstalling the loader
_installed_meta_cache = { }
def install_meta(address):
    if address not in _installed_meta_cache:
        finder = UrlMetaFinder(address)
        _installed_meta_cache[address] = finder
        sys.meta_path.append(finder)
        log.debug('%r installed on sys.meta_path', finder)

def remove_meta(address):
    if address in _installed_meta_cache:
        finder = _installed_meta_cache.pop(address)
        sys.meta_path.remove(finder)
        log.debug('%r removed from sys.meta_path', finder)
```

äÿŃéİćæŸŕäÿĂäÿłäžď äžŠäijŽèŕİïijŇæijŤçď žäžĘåęĆä¡Ţä¡£çŤĺåL’■éİćçŽĎäżčçăĄïijŽ

```
>>> # importing currently fails
>>> import fib
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>> # Load the importer and retry (it works)
>>> import urlimport
>>> urlimport.install_meta('http://localhost:15000')
>>> import fib
I'm fib
>>> import spam
I'm spam
>>> import grok.blah
I'm grok.__init__
I'm grok.blah
>>> grok.blah.__file__
'http://localhost:15000/grok/blah.py'
>>>
```

è£ŹäÿłçL’źæőŁçŽĎæŰźæąĹäijŽåőL’èčĔäÿĂäÿłçL’źåĹńçŽĎæ§ěæL’¿åŹÍ `UrlMetaFinder` åőđä¿ŃïijŇ ä¡IJäÿž `sys.meta_path` äÿ■æIJĂåŘŐçŽĎåőđä¡ŞãĂĆ å¡ŞæĺąåİŮèćńåŕijåĚěæŮűïijŇäijŽä¿İæ■ő `sys.meta_path` äÿ■çŽĎæ§ěæL’¿åŹĺåőŽä¡■æĺąåİŮãĂĆ åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇ`UrlMetaFinder` åőđä¿ŃæŸŕæIJĂåŘŐäÿĂäÿłæ§ěæL’¿åŹÍæŰźæąĹïijŇ å¡ŞæĺąåİŮåIJÍäżżä¡ŢäÿĂäÿłæŹőéĂŽåIJřæŰźéČ¡æL’¿äÿ

ä¡IJäÿžåÿÿèğĄçŽĎåőđçŐřæŰźæąĹïijŇ`UrlMetaFinder` çśżåŇĚèčĚåIJÍäÿĂäÿłçŤÍæĹůæŇĞåőŽçŽĎURLäÿŁãĂĆ åIJÍåĘĚéČÍïijŇæ§ěæL’¿åŹléĂŽè£ĞæŁŞåŔŰæŇĞåčåŕijåĚěçŽĎæŮűåĂŹïijŇæĺąåİŮåŘ■äijŽèů§åůšæIJL’çŽĎéŞ¿æŐěä¡IJåŕźæŕŤãĂĆåęĆæđIJæL’¿åĹřäžĘäÿĂäÿłåÑäÿĂäÿłå■ŢçŃńçŽĎ `UrlModuleLoader` çśżèćńçŤÍæİěäżŐè£IJçÍŃæIJžåŹÍäÿŁåŁäè¡¡æžŘäżčçăĄåźűåĹŽåżžè£ŹéĞŇçijŞå■ŸéŞ¿æŐěçŽĎäÿĂäÿłåŐ§åŻăæŸŕéĄ£åĚ■äÿ■å£ĚèęĄçŽĎHTTPèŕůæśĆéĞ■åđ’■åŕijåĚěãĂĆ

èĞłåőŽäźL’åŕijåĚěçŽĎçńňäžŇçğ■æŰźæşŢæŸŕçijŰåĘŹäÿĂäÿłéŠl’å■ŘçŻt’æŐěątŇåĚěåĹř `sys.path` åŔŸéĞŘäÿ■åŐżïijŇ èŕĘåĹńæ§ŘäžŻçŻőå¡ŢåŚ¡åŘ■æĺąåijŔãĂĆ åIJÍ `urlimport.py` äÿ■æůżåŁăåęĆäÿŇçŽĎçśżåŠŇæŤŕæŇĄåĞ¡æŢřïijŽ

```
# urlimport.py
# ... include previous code above ...
# Path finder class for a URL
class UrlPathFinder(importlib.abc.PathEntryFinder):
    def __init__(self, baseurl):
        self._links = None
        self._loader = UrlModuleLoader(baseurl)
        self._baseurl = baseurl

    def find_loader(self, fullname):
        log.debug('find_loader: %r', fullname)
        parts = fullname.split('.')
        basename = parts[-1]
        # Check link cache
```

```
        if self._links is None:
            self._links = [] # See discussion
            self._links = _get_links(self._baseurl)

        # Check if it's a package
        if basename in self._links:
            log.debug('find_loader: trying package %r', fullname)
            fullurl = self._baseurl + '/' + basename
            # Attempt to load the package (which accesses __init__.
↪py)
            loader = UrlPackageLoader(fullurl)
            try:
                loader.load_module(fullname)
                log.debug('find_loader: package %r loaded',␣
↪fullname)
            except ImportError as e:
                log.debug('find_loader: %r is a namespace package',␣
↪fullname)
                loader = None
            return (loader, [fullurl])

        # A normal module
        filename = basename + '.py'
        if filename in self._links:
            log.debug('find_loader: module %r found', fullname)
            return (self._loader, [])
        else:
            log.debug('find_loader: module %r not found', fullname)
            return (None, [])

    def invalidate_caches(self):
        log.debug('invalidating link cache')
        self._links = None

# Check path to see if it looks like a URL
_url_path_cache = {}
def handle_url(path):
    if path.startswith(('http://', 'https://')):
        log.debug('Handle path? %s. [Yes]', path)
        if path in _url_path_cache:
            finder = _url_path_cache[path]
        else:
            finder = UrlPathFinder(path)
            _url_path_cache[path] = finder
        return finder
    else:
        log.debug('Handle path? %s. [No]', path)

def install_path_hook():
    sys.path_hooks.append(handle_url)
```

```
    sys.path_importer_cache.clear()
    log.debug('Installing handle_url')

def remove_path_hook():
    sys.path_hooks.remove(handle_url)
    sys.path_importer_cache.clear()
    log.debug('Removing handle_url')
```

èęĄä¡£çŤĺè£Źäÿłèůŕå¿ĎæşěæL'¿åŹĺïijŇä¡ăåŔłéIJĂèęĄåIJĺ sys.path äÿ■åŁăåĚěURLéŞ¿æŐěãĂĆä¿ŃåęĆïijŽ

```
>>> # Initial import fails
>>> import fib
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'

>>> # Install the path hook
>>> import urlimport
>>> urlimport.install_path_hook()

>>> # Imports still fail (not on path)
>>> import fib
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'

>>> # Add an entry to sys.path and watch it work
>>> import sys
>>> sys.path.append('http://localhost:15000')
>>> import fib
I'm fib
>>> import grok.blah
I'm grok.__init__
I'm grok.blah
>>> grok.blah.__file__
'http://localhost:15000/grok/blah.py'
>>>
```

åĚşéŤőçĆźářśæŸŕ handle_url() åĞ¡æŢřïijŇåőČèćńæůžåŁăåĹřäžĘ sys.path_hooks åŔŸéĞŔäÿ■ãĂĆ å¡Ş sys.path çŽĎåőđä¡Şèćńåď'ĎçŘĘæŮűïijŇäijŽèřČçŤĺ sys.path_hooks äÿ■çŽĎåĞ¡æŢřãĂĆ åęĆæđIJäżżä¡Ţäÿ Ääÿłå Ğ¡æŢřè£ŤåŻđäžĘäÿ Ääÿłæ§ěæL'¿åŹĺářźèśąsys.path åőđä¡ŞåŁäè¡¡æĺąåİŮãĂĆ

è£IJçĺŃæĺąåİŮåŁäè¡¡èů§åĚűäżŰçŽĎåŁäè¡¡ä¡£çŤĺæŰźæşŢåĞääźŐæŸŕäÿĂæăůçŽĎãĂĆä¿ŃåęĆïijŽ

```
>>> fib
<urlmodule 'fib' from 'http://localhost:15000/fib.py'>
>>> fib.__name__
'fib'
>>> fib.__file__
```

```
'http://localhost:15000/fib.py'
>>> import inspect
>>> print(inspect.getsource(fib))
# fib.py
print("I'm fib")

def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n-1) + fib(n-2)
>>>
```

èőÍèőž

åIJÍèŕęçżĘèőÍèőžäźŃåL’■ïijŇæIJL’çĆźèęĄåijżèřČçŽĎæŸŕïijŇPythonçŽĎæÍąåİŮãĂĄåŇĚåŠŇåŕijåĚěæIJ
å■şä¡£çżŔéłŇäÿřåŕŇçŽĎPythonçÍŃåžŔåŚŸäź§å¿ÍåřŚèČ¡çš¿éĂŽåőČäżñãĂĆ
æĹŚåIJÍè£ŹéĞŇæŐÍè■ŘäÿĂäžŽåĂijçŽĎåŐżèŕżçŽĎæŰĞæąčåŠŇäźęçś■ïijŇåŇĚæŃň im-
portlib module åŠŇ PEP 302. æŰĞæąčåĘĚåőźåIJÍè£ŹéĞŇäÿ■äijŽèćńéĞ■åd’■æŔŘåĹŕïijŇäÿ■è£ĞæĹŚåIJÍè£Ž

éęŰåĚĹïijŇåęĆædIJä¡ăæČşåĹŻåžžäÿĂäÿłæŰřçŽĎæÍąåİŮåŕźèśąïijŇä¡£çŤÍ `imp.new_module()` åĞ¡æŢřïijŽ

```
>>> import imp
>>> m = imp.new_module('spam')
>>> m
<module 'spam'>
>>> m.__name__
'spam'
>>>
```

æÍąåİŮåŕźèśąéĂŽåÿÿæIJL’äÿĂäžŽæIJ§æIJŻåśđæĂğïijŇåŇĚæŃň `__file__` ïijĹè£ŘèąŇæÍąåİŮåŁäè¡¡èŕ■åŘęçŽĎæŰĞäžűåŘ■ïijL’ åŠŇ `__package__` (åŇĚåŘ■)ãĂĆ

åĚűæňąïijŇæÍąåİŮäijŽèćńèğčéĞŁåŹÍçijŞå■ŸèţůæİěãĂĆæÍąåİŮçijŞå■ŸåŘŕäžěåIJÍå■ŮåĚÿ `sys.modules` äÿ■èćńæL’¿åĹřãĂĆ åŽääÿžæIJL’äžĘè£ŹäÿłçijŞå■ŸæIJžåĹűïijŇéĂŽåÿÿåŘŕäžěårĘçijŞå■ŸåŠ

```
>>> import sys
>>> import imp
>>> m = sys.modules.setdefault('spam', imp.new_module('spam'))
>>> m
<module 'spam'>
>>>
```

åęĆædIJçżŹåőŽæÍąåİŮåůšçżŘå■ŸåIJÍéĆčäźĹåřśäijŽçŻťæŐěèŐůå¿ŮåůšçżŕèćńåĹŻåžžè£ĞçŽĎæÍąåİŮï

```
>>> import math
>>> m = sys.modules.setdefault('math', imp.new_module('math'))
>>> m
<module 'math' from '/usr/local/lib/python3.3/lib-dynload/math.so'>
```

```
>>> m.sin(2)
0.9092974268256817
>>> m.cos(2)
-0.4161468365471424
>>>
```

çŤśäžŐåĹŻåžžæĺąåİŮå¿ĹçőĂå■ŢïijŇå¿Ĺåőźæ˲ŞçijŰåĘŹçőĂå■ŢåĞ¡æŢřæŕŤåęĆçňňäÿĂéČĺåĹĘçŽĎ `load_module()` åĞ¡æŢřãĂĆ è£ŹäÿłæŰźæąĹçŽĎäÿĂäÿłçijžçĆźæŸŕå¿ĹéŽ¿åď ĎçŘĘåď■æİĆæČĚåĘţæŕŤ äÿžäžĘåď ĎçŘĘäÿĂäÿłåŇĚïijŇä¡ãèęĄéĞ■æŰřåőđçŐřæŹőéĂŽimportèŕ■åŔěçŽĎåžŢåśĆéĂżè¿ŚïijĹæŕŤåęĆa æĽğèąŇéĆčäžŻæŰĞäžűïijŇèő¿ç¡őèůŕå¿Ďç■Ľ˲ijĽ˜ãĂĆè£Źäÿłåď ■æİĆæĂğåřśæŸŕäÿžäžĂäźĹæIJĂåě¡çŻť æŐ

æĽľåśŢimportèŕ■åŔěå¿ĹçőĂå■ŢïijŇä¡ĘæŸŕäijŽæIJĽå¿Ĺåď ŽçğżåŁĺæŞ■ä¡IJãĂĆ æIJĂéńŸåśĆäÿŁïijŇåŕijåĚěæŞ■ä¡IJèćńäÿĂäÿłä¡■äžŐsys.meta_pathåĹŮèąĺäÿ■çŽĎâĂIJåĚČèůŕå¿ĎâĂİæ§ěæ åęĆæđIJä¡äè¿ŞåĞžåőČçŽĎåĂijˆijŇäijŽçIJŃåĹřäÿŃéİćè£ŹæăůïijŽ

```
>>> from pprint import pprint
>>> pprint(sys.meta_path)
[<class '_frozen_importlib.BuiltinImporter'>,
<class '_frozen_importlib.FrozenImporter'>,
<class '_frozen_importlib.PathFinder'>]
>>>
```

å¡ŞæĽğèąŇäÿĂäÿłèŕ■åŔěæŕŤåęĆ `import fib` æŮűïijŇèğčéĞŁåŹĺäijŽéĄ■åŐĘsys.mata_pathäÿ■çŽĎ èřČçŤĺåőČäžňçŽĎ `find_module()` æŰźæşŢåőŽä¡■æ■čçąőçŽĎæĺąåİŮåŁäè¡¡åŹĺãĂĆ åŔŕäžěéĂŽè£ĞåőđéłŇæİěçIJŃçIJŃïijŽ

```
>>> class Finder:
...     def find_module(self, fullname, path):
...         print('Looking for', fullname, path)
...         return None
...
>>> import sys
>>> sys.meta_path.insert(0, Finder()) # Insert as first entry
>>> import math
Looking for math None
>>> import types
Looking for types None
>>> import threading
Looking for threading None
Looking for time None
Looking for traceback None
Looking for linecache None
Looking for tokenize None
Looking for token None
>>>
```

æşlæĎŔçIJŃ `find_module()` æŰźæşŢæŸŕæĂŐæăůåIJĺæŕŔäÿĂäÿłåŕijåĚěåřśèćńèğęåŔŠçŽĎãĂĆ è£ŹäÿłæŰźæşŢäÿ■çŽĎpathåŔĆæŢřçŽĎä¡IJçŤĺæŸŕåď ĎçŘĘåŇĚãĂĆ åď ŽäÿłåŇĚèćńåŕijåĚěïijŇåřśæŸŕäÿĂäÿłåŔŕåIJĺåŇĚçŽĎ `__path__` åśđæĂğäÿ■æĽ¿åĹřçŽĎèůŕå¿ĎåĹŮèąĺãĂĆ èęĄæĽ¿åĹřåŇĚçŽĎå■ŘçżĎäžűåřśèęĄæčĂæ§ěè£ŹäžŻèůŕå¿ĎãĂ

æŕŤåęĆæşĺæĎŘåŕźäžŐ xml.etree åŠŇ xml.etree.ElementTree çŽĎèůŕå¿ĎéĚ■ç¡őïijŽ

```
>>> import xml.etree.ElementTree
Looking for xml None
Looking for xml.etree ['/usr/local/lib/python3.3/xml']
Looking for xml.etree.ElementTree ['/usr/local/lib/python3.3/xml/
 ↪etree']
Looking for warnings None
Looking for contextlib None
Looking for xml.etree.ElementPath ['/usr/local/lib/python3.3/xml/
 ↪etree']
Looking for _elementtree None
Looking for copy None
Looking for org None
Looking for pyexpat None
Looking for ElementC14N None
>>>
```

åIJĺ sys.meta_path äÿŁæ§ěæL'¿åŹĺçŽĎä¡■ç¡őå¿ĹéĞ■èęĄïijŇářĘåőČäžŐéŸ§åď't'çğžåĹřéŸ§ář¿ïijŇ

```
>>> del sys.meta_path[0]
>>> sys.meta_path.append(Finder())
>>> import urllib.request
>>> import datetime
```

çŐřåIJĺä¡ăçIJŃäÿ■åĹřäżża¡Ţè¿ŞåĞžäžĘïijŇåŻääÿžářijåĚěèćńsys.meta_pathäÿ■çŽĎåĚűäżŮåődä¡Şåď'Ďç
è£ŹæŮűåĂŹïijŇä¡ääŘłæIJL'åIJĺářijåĚěäÿ■å■ŸåIJĺæĺąåİŮçŽĎæŮűåĂŹæL'■èČ¡çIJŃåĹřåőČèćńèğęåŔŠïijŽ

```
>>> import fib
Looking for fib None
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>> import xml.superfast
Looking for xml.superfast ['/usr/local/lib/python3.3/xml']
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'xml.superfast'
>>>
```

ä¡ääźŃåL'■åőL'èčĚè£ĞäÿĂäÿłæ■ŢèŐůæIJłç§ěæĺąåİŮçŽĎæ§ěæL'¿åŹĺïijŇè£ŹäÿłæŸŕ
UrlMetaFinder çśżçŽĎåĚşéŤőãĂĆ äÿĂäÿł UrlMetaFinder åődä¿ŃèćńæůżåŁääĹř
sys.meta_path çŽĎæIJńář¿ïijŇä¡IJäÿžæIJĂåŘŐäÿĂäÿłæ§ěæL'¿åŹĺæŰźæąĹãĂĆ
åęĆædIJèćńèřůæśĆçŽĎæĺąåİŮåŘ■äÿ■èČ¡åőŽä¡■ïijŇářśäijŽèćńè£Źäÿłæ§ěæL'¿åŹĺåď'ĎçŘĘæŐL'ãĂĆ
åď'ĎçŘĘåŇĚçŽĎæŮűåĂŹéIJĂèęĄæşĺæĎŘïijŇåIJĺpathåŔĆæŢřäÿ■æŇĞåőŽçŽĎåĂijéIJĂèęĄèćńæčĂæ§ëïijŇ
åęĆædIJäÿ■æŸŕïijŇèřěå■ŘæĺąåİŮå£Ěéąžå¡ŠåśđäžŐåĚűäżŰæ§ěæL'¿åŹĺåźűèćńå£¡çŢěæŐL'ãĂĆ

åŕźäžŐåŇĚçŽĎåĚűäżŰåď'ĎçŘĘåŔŕåIJĺ UrlPackageLoader
çśżäÿ■èćńæL'¿åĹřãĂĆ è£Źäÿłçśżäÿ■äijŽåŕijåĚěåŇĚåŘ■ïijŇèĂŇæŸŕåŐžåŁäè¡¡åŕźåžŢçŽĎ
__init__.py æŰĞäžűãĂĆ åőČäź§äijŽèő¿ç¡őæĺąåİŮçŽĎ __path__

åśđæĂğïijŇè£ŹäÿĂæ■ěå¿ĹéĞ■èęĄïijŇ åŻăäÿžåIJĺåŁăè¡¡åŇĚçŽĎå■ŘæĺąåİŮæŮűè£ŹäÿłåĂijäijŽèćńäijăçżŹå
`find_module()` èřČçŤĺãĂĆ å§žäžŐèůřå¿ĎçŽĎåŕijåĚěéŠľå■ŘæŸŕè£ŹäžŻæĂİæČşçŽĎäÿĂäÿłæĽľåśŢïijŇ
æĹŚäžňéČ¡ç§ěéĄŞïijŇ`sys.path` æŸŕäÿĂäÿłPythonæ§ěæĽ¿æĺąåİŮçŽĎçŻőå¡ŢåĹŮèąĺïijŇä¿ŃåęĆïijŽ

```
>>> from pprint import pprint
>>> import sys
>>> pprint(sys.path)
['',
'/usr/local/lib/python33.zip',
'/usr/local/lib/python3.3',
'/usr/local/lib/python3.3/plat-darwin',
'/usr/local/lib/python3.3/lib-dynload',
'/usr/local/lib/...3.3/site-packages']
>>>
```

åIJĺ `sys.path` äÿ■çŽĎæŕŔäÿĂäÿłåőđä¡ŞéČ¡äijŽèćńéćÌåd'ŰçŽĎçżŚåőŽåĹřäÿĂäÿłæ§ěæĽ¿åŹĺåŕźèśąäÿ
ä¡ääŔŕäžěéĂŽè£Ğæ§ěçIJŃ `sys.path_importer_cache`
åŐżçIJŃäÿŃè£ŹäžŻæ§ěæĽ¿åŹĺïijŽ

```
>>> pprint(sys.path_importer_cache)
{'.': FileFinder('.'),
'/usr/local/lib/python3.3': FileFinder('/usr/local/lib/python3.3'),
'/usr/local/lib/python3.3/': FileFinder('/usr/local/lib/python3.3/
 ↪'),
'/usr/local/lib/python3.3/collections': FileFinder('...python3.3/
 ↪collections'),
'/usr/local/lib/python3.3/encodings': FileFinder('...python3.3/
 ↪encodings'),
'/usr/local/lib/python3.3/lib-dynload': FileFinder('...python3.3/
 ↪lib-dynload'),
'/usr/local/lib/python3.3/plat-darwin': FileFinder('...python3.3/
 ↪plat-darwin'),
'/usr/local/lib/python3.3/site-packages': FileFinder('...python3.3/
 ↪site-packages'),
'/usr/local/lib/python33.zip': None}
>>>
```

`sys.path_importer_cache` æŕŤ `sys.path` äijŽæŻt'åd'ğçĆźïijŇ
åŻăäÿžåőČäijŽäÿžæĽĂæIJL'èćńåŁăè¡¡äžčçăĄçŽĎçŻőå¡Ţèőřå¡ŢåőČäžňçŽĎæ§ěæĽ¿åŹĺãĂĆ
è£ŹåŇĚæŃňåŇĚçŽĎå■ŘçŻőå¡ŢïijŇè£ŹäžŻéĂŽåÿÿåIJĺ `sys.path`
äÿ■æŸŕäÿ■å■ŸåIJĺçŽĎãĂĆ

èęĄæĽğèąŇ `import fib` ïijŇäijŽéąžåžŔæčĂæ§ě `sys.path` äÿ■çŽĎçŻőå¡ŢãĂĆ
åŕźäžŐæŕŔäÿłçŻőå¡ŢïijŇåŘ■çğřâĂIJfibâĂİäijŽèćńäijăçżŹçŻÿåžŤçŽĎ `sys.`
`path_importer_cache` äÿ■çŽĎæ§ěæĽ¿åŹĺãĂĆ è£ŹäÿłåŔŕäżěèőľä¡ääĹŽåžžèĞłåůśçŽĎæ§ěæĽ¿åŹĺåźűå

```
>>> class Finder:
... def find_loader(self, name):
...     print('Looking for', name)
...     return (None, [])
...
```

```
>>> import sys
>>> # Add a "debug" entry to the importer cache
>>> sys.path_importer_cache['debug'] = Finder()
>>> # Add a "debug" directory to sys.path
>>> sys.path.insert(0, 'debug')
>>> import threading
Looking for threading
Looking for time
Looking for traceback
Looking for linecache
Looking for tokenize
Looking for token
>>>
```

åIJÍè£ŹéĞŇïijŇä¡ääŔŕäżěäÿžåŘ■å■ŮâĂIJdebugâĂİåĹŻåżžäÿĂäÿłæŰřçŽĎçijŞå■Ÿåőđä¡Şåźűår̨åőČèő¿
`sys.path` äÿŁçŽĎçňňäÿĂäÿłãĂĆ åIJÍæĽĂæIJĽæŐěäÿŃæİěçŽĎårijåĚěäÿ■ïijŇä¡ääijŽçIJŃåĹŕä¡äçŽĎæ§ěa
äÿ■è£ĞïijŇçŤśäžŐåőČè£ŤåŻđ (None, [])ïijŇéĆčäźĹåď ĎçŘĘè£ŻçÍŃäijŽçżğçż■åď ĎçŘĘäÿŃäÿĂäÿłåőđä¡ŞãA

`sys.path_importer_cache` çŽĎä¡£çŤĺèćńäÿĂäÿłå■ŸåĆĺåIJĺ `sys.path_hooks` äÿ■çŽĎåĞ¡æŢřåĹŮèąÍæŐğåĹűãĂĆ èŕŢèŕŢäÿŃéİćçŽĎä¿Ńå■ŘïijŇåőČäijŽæÿĚéŹď çijŞå■Ÿåźűçżź `sys.path_hooks` æůżåŁääÿĂäÿłæŰřçŽĎèůŕå¿ĎæčĂæ§ěåĞ¡æŢř

```
>>> sys.path_importer_cache.clear()
>>> def check_path(path):
...     print('Checking', path)
...     raise ImportError()
...
>>> sys.path_hooks.insert(0, check_path)
>>> import fib
Checked debug
Checking .
Checking /usr/local/lib/python33.zip
Checking /usr/local/lib/python3.3
Checking /usr/local/lib/python3.3/plat-darwin
Checking /usr/local/lib/python3.3/lib-dynload
Checking /Users/beazley/.local/lib/python3.3/site-packages
Checking /usr/local/lib/python3.3/site-packages
Looking for fib
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>>
```

æ■čåęĆä¡äæĽĂèğĄïijŇcheck_path() åĞ¡æŢřèćńæŕŔäÿł `sys.path` äÿ■çŽĎåőđä¡ŞèřČçŤĺãĂĆ äÿ■éą¿ïijŇçŤśäžŐæŁŻåĞžäžĘ `ImportError` åijĆåÿÿïijŇ åŢěéČ¡äÿ■äijŽåŔŞçŤ§äžĘïijĹäžĚäżĚåřĘæčĂæ§ěè¡ňçğżåĹřsys.path_hooksçŽĎäÿŃäÿĂäÿłåĞ¡æŢřïijĽãĂĆ

ç§ěéĄŞäžĘæĂŐæăůsys.pathæŸŕæĂŐæăůèćńåď ĎçŘĘçŽĎïijŇä¡ääřśèČ¡æđĎåżžäÿĂäÿłèĞłåőŽäźĽèůŕå¿

```
>>> def check_url(path):
...     if path.startswith('http://'):
```

```
...         return Finder()
...     else:
...         raise ImportError()
...
>>> sys.path.append('http://localhost:15000')
>>> sys.path_hooks[0] = check_url
>>> import fib
Looking for fib # Finder output!
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'

>>> # Notice installation of Finder in sys.path_importer_cache
>>> sys.path_importer_cache['http://localhost:15000']
<__main__.Finder object at 0x10064c850>
>>>
```

è£Źåřśæ Ÿŕæ IJňèŁĆæ IJĂåŘŐéČĺåŁĘçŽĎåĚşéŤőçĆźãĂĆäžŇåőđäÿŁïijŇäÿĂäÿłçŤĺæİěåIJĺsys.pathäÿ■æ
å¡Şåő Čäžňèćńçćřå Łřç ŽĎæŮűåĂŹïijŇäÿĂäÿłæŰřçŽĎ `UrlPathFinder`
åőđä¿Ńèćńå ŁŽåžžåźűèćńæŤ¿åĚě `sys.path_importer_cache`.
äźŇåŘŐïijŇæĽĂæIJĽéIJĂèęĄæčÄæ§ě `sys.path` çŽĎåŕijåĚěèŕ■åŔěéČ¡äijŽä¡£çŤĺä¡ăçŽĎèĞłåőŽäźĽæ§ě

å§žäžŐèůŕå¿ĎåŕijåĚěçŽĎåŇĚåď ĎçŘEçĺ■å¿őæIJĽçĆźåď■æİĆïijŇåźűäÿŤèů§
`find_loader()` æŰźæşŢè£ŤåŻđåĂijæIJĽåĚşãĂĆ åŕźäžŐçőĂå■ŢæĺąåİŮïijŇ`find_loader()`
è£ŤåŻđäÿĂäÿłåĚČçżĎ(loader, None)ïijŇ åĚűäÿ■çŽĎload-
eræŸŕäÿĂäÿłçŤĺäžŐåŕijåĚěæĺąåİŮçŽĎåŁăè¡¡åŹĺåőđä¿ŃãĂĆ

åŕźäžŐäÿĂäÿłæŹőéĂŽçŽĎåŇĚïijŇ`find_loader()` è£ŤåŻđäÿĂäÿłåĚČçżĎ(loader,
path)ïijŇ åĚűäÿ■çŽĎloaderæŸŕäÿĂäÿłçŤĺäžŐåŕijåĚěåŇĚïijĹåźűæĽğèąŇ__init__.pyïijĽçŽĎåŁăè¡¡åŹĺåőđä¿
pathæŸŕäÿĂäÿłäijŽåĹİåğŃåŇŰåŇĚçŽĎ `__path__` åśđæĂğçŽĎçŽőå¡ŢåĹŮèąĺãĂĆ
ä¿ŃåęĆïijŇåęĆæđIJå§žçąĂURLæŸŕ http://localhost:15000 åźűäÿŤäÿĂäÿłçŤĺæĹůæĽğèąŇ
`import grok` , éĆčäźĹ `find_loader()` è£ŤåŻđçŽĎpathåřśäijŽæŸŕ ['http://localhost:
15000/grok']

`find_loader()` è£ŸèęĄèČ¡åď ĎçŘEäÿĂäÿłåŚ¡åŘ■çľžéŮť åŇĚãĂĆ
äÿĂäÿłåŚ¡åŘ■çľžéŮť åŇĚäÿ■æIJĽäÿĂäÿłåŘĹæşŢçŽĎåŇĚçŽőå¡ŢåŘ■ïijŇä¡ĘæŸŕäÿ■å■ŸåIJĺ__init__.pyæŰ
è£ŹæăůçŽĎèŕİïijŇ`find_loader()` å£ĚéąžèŁŤåŻđäÿĂäÿłåĚČçżĎ(None, path)ïijŇ
pathæŸŕäÿĂäÿłçŽőå¡ŢåĹŮèąĺïijŇçŤśåőČæİěæđĎåžžåŇĚçŽĎåőŽäźĽæIJĽ__init__.pyæŰĞäžűçŽĎ__path__
åŕźäžŐè£ŹçğæČĚåĘţïijŇåŕijåĚěæIJžåĹűäijŽçżğçż■åĽ■èąŇåŐžæčÄæ§ěsys.pathäÿ■çŽĎçŽőå¡ŢãĂĆ
åęĆæđIJæĽ¿åĹřäžĘåŚ¡åŘ■çľžéŮť åŇĚïijŇæĽĂæIJĽçŽĎçżŞæđIJèůŕå¿Ďèćńå ŁååĹřäÿĂèţůæİěæđĎåžžæIJĂ
åĚşäžŐåŚ¡åŘ■çľžéŮť åŇĚçŽĎæŽť åď Žä£qæĄŕèŕůåŔĆèĂČ10.5åřŔèŁĆãĂĆ

æĽĂæIJĽçŽĎåŇĚéČ¡åŇĚåŘńäžĘäÿĂäÿłåĘĚéČĺèůŕå¿Ďèő¿ç¡őïijŇåŔŕäžěåIJĺ__path__åśđæĂğäÿ■çIJŃ

```
>>> import xml.etree.ElementTree
>>> xml.__path__
['/usr/local/lib/python3.3/xml']
>>> xml.etree.__path__
['/usr/local/lib/python3.3/xml/etree']
>>>
```

äźŃåL’■æŔŘåĹřïijŇ__path__çŽĎèő¿ç¡őæŸŕéĂŽè£Ğ find_loader() æŰźæşŢè£ŤåŻđåĂijæŐğåĹűçŽĎãĂĆ äÿ■è£ĞïijŇ__path__æŐëäÿŃæİëäź§èćńsys.path_hooksäÿ■çŽĎåĞ¡æŢåŻăæ■d’ïijŇä¡ĘåŇĚçŽĎå■ŘçżĎäžűèćńåŁăè¡¡åŘŐïijŇä¡■äžŐ__path__äÿ■çŽĎåőđä¡ŞäijŽèćń handle_url() åĞ¡æŢřæčĂæ§ěãĂĆ è£ŹäijŽářijèĞt’æŰřçŽĎ UrlPathFinder åőđä¿Ńèćńå Ĺ ŻåžžåźűäÿŤèćńåŁăåĚěåĹř sys.path_importer_cache äÿ■ãĂĆ

è£ŸæIJL’äÿłéŽ¿çĆźářśæŸŕ handle_url() åĞ¡æŢřäžěåŘŁåőČèů§åĘĚéČĺä¡£çŤĺçŽĎ _get_links() åĞ¡æŢřäźŃéŮt’çŽĎäžd’äžŠãĂĆ åęĆæđIJä¡ăçŽĎæ§ěæL’¿åŹĺåőđçŐřéIJĂèęĄä¡£çŤĺåĹřåĚűæIJL’åŘřèČ¡è£ŹäžŻæĺąåİŮäijŽåIJĺæ§ěæL’¿åŹĺæŞ■ä¡IJæIJ§éŮt’è£ŻèąŇæŻt’åd’ŽçŽĎářijåĚěãĂĆ åőČåŘřäžěářijèĞt’ handle_url() åŠŇåĚűäžŰæ§ěæL’¿åŹĺéČĺåĹĘéŹůåĚěäÿĂçğ■éĂŠå¡Šå¿łçŐŕçŁűæĂĄãäÿžäžEèğčéĞŁè£Źçğ■åŘřèČ¡æĂğïijŇåődçŐřäÿ■æIJL’äÿĂäÿłèćńåĹŻåžžçŽĎæ§ěæL’¿åŹĺçijŞå■ŸïijĹærŔäÿĂ åőČåŘřäžěéĄ£åĚ■åĹŻåžžéĞ■åd’■æ§ěæL’¿åŹĺçŽĎéŮőécŸãĂĆ åŘęåd’ŰïijŇäÿŃéİćçŽĎäžčçăĄçL’Gæőţåŕřäžěçąőä£İæ§ěæL’¿åŹĺäÿ■äijŽåIJĺåĹİåğŃåŇŰéŞ¿æŐěéŻĘåŘĹçŽĎ

```
# Check link cache
if self._links is None:
    self._links = [] # See discussion
    self._links = _get_links(self._baseurl)
```

æIJĂåŘŐïijŇæ§ěæL’¿åŹĺçŽĎ invalidate_caches() æŰźæşŢæŸŕäÿĂäÿłåůěåĚůæŰźæşŢïijŇçŤĺæİěæÿĚçŘĘåĘĚéČĺçijŞå■ŸãĂĆ è£ŹäÿłæŰźæşŢåĘ■çŤĺæĹůèřČçŤĺ importlib.invalidate_caches() çŽĎæŮűåĂŹèćńèğęåŔŠãĂĆ åęĆæđIJä¡ăæČşèől’URLåŕijåĚěèĂĚéĞ■æŰřèŕžåŘŰéŞ¿æŐěåĹŮèąĺçŽĎèŕĺåŘŕä

åŕźæŕŤäÿŃäÿd’çğ■æŰźæąĹïijĹä£őæŤźsys.meta_pathæĹŰä¡£çŤĺäÿĂäÿłèůŕå¿ĎéŠl’å■ŘïijL’ãĂĆ ä¡£çŤĺsys.meta_pathçŽĎåŕijåĚěèĂĚåŘŕäžěæŇL’çĚğèĞłåůśçŽĎéIJĂèęĄèĞłçŤšåd’ĎçŘĘæĺąåİŮãĂĆ ä¿ŃåęĆïijŇåőČäžňåŘŕäžěäžŐæŢřæ■őåžŞäÿ■åŕijåĚěæĹŰäžěäÿ■åŘŇäžŐäÿĂèĹňæĺąåİŮ/åŇĚåd’ĎçŘĘæŰźåi è£Źçğ■èĞłçŤšåŘŇæăůæĎŘåŚşçİĂåŕijåĚěèĂĚéIJĂèęĄèĞłåůśè£ŻèąŇåĘĚéČĺçŽĎäÿĂäžŽçőąçŘĘãĂĆ åŘęåd’ŰïijŇå§žäžŐèůŕå¿ĎçŽĎéŠl’å■ŘåŘłæŸŕéĂĆçŤĺäžŐåŕźsys.pathçŽĎåd’ĎçŘĘãĂĆ éĂŽè£Ğè£Źçğ■æL’l’åśŢåŁăè¡¡çŽĎæĺąåİŮèů§æŹőéĂŽæŰźåijŘåŁăè¡¡çŽĎçL’źæĂğæŸŕäÿĂæăůçŽĎãĂĆ

åęĆæđIJåĹřçŐřåIJĺäÿžæ■ćä¡ăè£ŸæŸŕäÿ■æŸŕå¿ĹæŸŐçŹ¡ïijŇéĆčäźĹåŘŕäžěéĂŽè£ĞåćđåŁăäÿĂäžŽæŮ

```
>>> import logging
>>> logging.basicConfig(level=logging.DEBUG)
>>> import urlimport
>>> urlimport.install_path_hook()
DEBUG:urlimport:Installing handle_url
>>> import fib
DEBUG:urlimport:Handle path? /usr/local/lib/python33.zip. [No]
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>> import sys
>>> sys.path.append('http://localhost:15000')
>>> import fib
DEBUG:urlimport:Handle path? http://localhost:15000. [Yes]
DEBUG:urlimport:Getting links from http://localhost:15000
DEBUG:urlimport:links: {'spam.py', 'fib.py', 'grok'}
DEBUG:urlimport:find_loader: 'fib'
DEBUG:urlimport:find_loader: module 'fib' found
```

```
DEBUG:urlimport:loader: reading 'http://localhost:15000/fib.py'
DEBUG:urlimport:loader: 'http://localhost:15000/fib.py' loaded
I'm fib
>>>
```

æIJĂåŘŐïijŇåżžèőőä¡ăèŁśçĆźæŮűéŮt'çIJŃçIJŃ PEP 302 äżěåŔŁimportlibçŽĎæŰĞæąčãĂĆ

10.12 åŕijåĚěæĺąåİŮçŽĎåŘŇæŮűä£őæŤźæĺąåİŮ

éŮőécŸ

ä¡ăæČşçżŹæ§Řäÿłåůšå■ŸåIJĺæĺąåİŮäÿ■çŽĎåĞ¡æŢřæůżåŁăèčĚéěřåŹĺãĂĆ äÿ■è£ĞïijŇåL'■æŔŘæŸŕè£ŹäÿłæĺąåİŮåůšçżŔèćńåŕijåĚěåźűäÿŤèćńä¡£çŤĺè£ĞãĂĆ

èğčåĘşæŰźæąĹ

è£ŹéĞŇéŮőécŸçŽĎæIJňèť'ĺåřśæŸŕä¡ăæČşåIJĺæĺąåİŮèćńåŁăè¡¡æŮűæL'ğèąŇæ§Řäÿłåŀĺä¡IJãĂĆ åŔŕèČ¡æŸŕä¡ăæČşåIJĺäÿĂäÿłæĺąåİŮèćńåŁăè¡¡æŮűèğęåŔŚæ§ŘäÿłåŻđèřČåĞ¡æŢřæİěéĂŽç§ëä¡ããĂĆ

è£ŹäÿłéŮőécŸåŔŕäżěä¡£çŤĺ10.11åřŔèŁĆäÿ■åŘŇæăůçŽĎåŕijåĚěéŠl'å■ŘæIJžåĹűæİěåőđçŐřãĂĆäÿŃéİć

```
# postimport.py
import importlib
import sys
from collections import defaultdict

_post_import_hooks = defaultdict(list)

class PostImportFinder:
    def __init__(self):
        self._skip = set()

    def find_module(self, fullname, path=None):
        if fullname in self._skip:
            return None
        self._skip.add(fullname)
        return PostImportLoader(self)

class PostImportLoader:
    def __init__(self, finder):
        self._finder = finder

    def load_module(self, fullname):
        importlib.import_module(fullname)
        module = sys.modules[fullname]
        for func in _post_import_hooks[fullname]:
            func(module)
```

```
        self._finder._skip.remove(fullname)
        return module

def when_imported(fullname):
    def decorate(func):
        if fullname in sys.modules:
            func(sys.modules[fullname])
        else:
            _post_import_hooks[fullname].append(func)
        return func
    return decorate

sys.meta_path.insert(0, PostImportFinder())
```

è£Źæăůïijŉä¡ăářšåŔŕäžěä¡£çŤÍ `when_imported()` èčĚéěřåŹĺäžĘïijŉä¿ŃåęĆïijŽ

```
>>> from postimport import when_imported
>>> @when_imported('threading')
... def warn_threads(mod):
...     print('Threads? Are you crazy?')
...
>>>
>>> import threading
Threads? Are you crazy?
>>>
```

ä¡IJäÿžäÿĂäÿłæŻť'åőđéŹĚçŽĎä¿Ńå■Řïijŉä¡ääŔŕèČ¡æČşåIJĺåůšå■ŸåIJĺçŽĎåőŽäźĽ'äÿŁéİćæůżåŁăèčĚé

```
from functools import wraps
from postimport import when_imported

def logged(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        print('Calling', func.__name__, args, kwargs)
        return func(*args, **kwargs)
    return wrapper

# Example
@when_imported('math')
def add_logging(mod):
    mod.cos = logged(mod.cos)
    mod.sin = logged(mod.sin)
```

èőĺèőž

æIJňèŁĆæŁĂæIJŕä¿ÌètŰäžŐ10.11åřŔèŁĆäÿ■èőšè£řè£ĞçŽĎåŕijåĚéěŠľå■ŘïijŉŇåźűçĺ■ä¡IJä£őæŤźãĂĆ

`@when_imported` èčĚéěřåŹĺçŽĎä¡IJçŤĺæŸŕæşĺåĘŇåIJĺåŕijåĚěæŮűèćńæ£Äæť'żçŽĎåď'ĎçŘĘåŹĺåĞ¡à èŕěèčĚéěřåŹĺæčĂæ§ěsys.modulesæİěæ§ěçIJŃæĺąåİŮæŸŕåŘęçIJ§çŽĎåůšçżŔèćńåŁäè¡¡äžĘãĂĆ

åęĆæđIJæŸŕçŽĎèŕÏïijŇèŕěåď'ĎçŘĘåŹÍèćńçńŇå■şèřČçŤĺãĂĆäÿ■çĎűïijŇåď'ĎçŘĘåŹÍèćńæůżåŁäåĹř
`_post_import_hooks` å■ŮåĚÿäÿ■çŽĎäÿĂäÿłåĹŮèąĺäÿ■åŐžãĂĆ
`_post_import_hooks` çŽĎä¡IJçŤĺåřśæŸŕæŤűéŻĘæĽĂæIJĽçŽĎäÿžæŕŔäÿłæĺąåİŮæşĺåĘŇçŽĎåď'ĎçŘĘ
äÿĂäÿłæĺąåİŮåŘŕäžěæşĺåĘŇåď'Žäÿłåď'ĎçŘĘåŹĺãĂĆ

èęĄèőľ'æĺąåİŮåŕijåĚěåŘŐèğęåŘŚæůżåŁăçŽĎåŁĺä¡IJïijŇ`PostImportFinder`
çśżèćńèő¿ç¡őäÿžsys.meta_pathçňňäÿĂäÿłåĚČçť'ããĂĆ åőČäijŽæ■ŢèŐůæĽĂæIJĽ'æĺąåİŮåŕijåĚěæŞ■ä¡IJãĂĆ

æIJňèŁĆäÿ■çŽĎ `PostImportFinder` çŽĎä¡IJçŤĺåźűäÿ■æŸŕåŁäè¡¡æĺąåİŮïijŇèĂŇæŸŕèĞłåÿęåŕijåĚ
åőđéŹĚçŽĎåŕijåĚèèćńåğŤæť¿çżŹä¡■äžŐsys.meta_pathäÿ■çŽĎåĚűäżŰæ§ěæĽ'¿åŹĺãĂĆ
`PostImportLoader` çśżäÿ■çŽĎ `imp.import_module()`
åĞ¡æŢřèćńéĂŠå¡ŠçŽĎèřČçŤĺãĂĆ äÿžäžĘéĄ£åĚ■éŹůåĚěæŮăçž£å¿łçŐŕijŇ`PostImportFinder`
ä£ÍæŇĄäžĘäÿĂäÿłæĽĂæIJĽèćńåŁäè¡¡è£ĞçŽĎæĺąåİŮéŻĘåŘĹãĂĆ
åęĆæđIJäÿĂäÿłæĺąåİŮåŘ■å■ŸåIJĺåřśäijŽçŻť'æŐěèćńå£¡çŢěæŐĽãĂĆ

å¡ŞäÿĂäÿłæĺąåİŮèćń `imp.import_module()` åŁăè¡¡åŘŐïijŇ
æĽĂæIJĽ'åIJĺ_post_import_hooksèćńæşĺåĘŇçŽĎåď'ĎçŘĘåŹÍèćńěřČçŤĺïijŇä¡£çŤĺæŰřåŁäè¡¡æĺąåİŮä¡IJäÿžă

æIJĽäÿĂçĆźéIJĂèęĄæşĺæĎŔçŽĎæŸŕæIJňæIJžäÿ■éĂĆçŤĺäžŐéĆčäžŻéĂŽè£Ğ `imp.reload()` èćńæŸ¿åijŕåŁäè¡¡çŽĎæĺąåİŮãĂĆ äź§åřśæŸŕèŕť'ïijŇåęĆæđIJä¡ääŁäè¡¡äÿĂäÿłäźŇåĽ'■åůšèćńåŁă
åŘęåď'ŰïijŇèęĄæŸŕä¡ääžŐsys.modulesäÿ■åĽäéŹď'æĺąåİŮçĎűåŘŐåĘ■éĞ■æŰřåŕijåĚěïijŇåď'ĎçŘĘåŹĺåŘĹä

æŻť'åď'ŽåĚşäžŐåŕijåĚěåŘŐéŠl'å■Ră£ąæĄŕèŕůåŘĆèĂČ PEP 369.

10.13 åőĽ'èčĚçğĄæIJĽçŽĎåŇĚ

éŮőécŸ

ä¡ääČşèęĄåőL'èčĚäÿĂäÿłçňňäÿL'æŰźåŇĚïijŇä¡ĘæŸŕæšąæIJĽ'æİČéŹŘåřĘåőČåőL'èčĚåĹřçşżçż§Python
æĹŰèĂĚïijŇä¡ääŘŕèČ¡æČşèęĄåőL'èčĚäÿĂäÿłä¿ŻèĞłåůśä¡£çŤĺçŽĎåŇĚïijŇèĂŇäÿ■æŸŕçşżçż§äÿŁéİćæĽ'Ăa

èğčåĘşæŰźæąĹ

PythonæIJĽäÿĂäÿłçŤĺæĹůåőL'èčĚçŽőå¡ŢïijŇé ĂŽåÿÿçśżäijij”~/.local/lib/python3.3/site-packages”ãĂĆ èęĄåijžåĹűåIJĺè£ŹäÿłçŻőå¡Ţäÿ■åőL'èčĚåŇĚïijŇåŘŕä¡£çŤĺåőL'èčĚéĂL'éąźâĂIJ–userâĂİãĂĆä¿ŃåęĆïijŽ

```
python3 setup.py install --user
```

æĹŰèĂĚ

```
pip install --user packagename
```

åIJĺsys.pathäÿ■çŤĺæĹůçŽĎâĂIJsite-packagesâĂİçŻőå¡Ţä¡■äžŐçşżçż§çŽĎâĂIJsite-packagesâĂİçŻőå¡ŢäźŃåĽ■ãĂĆ åŻäæ■ď'ïijŇä¡ääőL'èčĚåIJĺéĞŇéİćçŽĎåŇĚářśæŕŤçşżçż§åůšäőL'èčĚçŽĎåŇïijĹář¡çőąåźűäÿ■æĂżæŸŕè£ŹæăüïijŇèęĄåŘŰåĘşäžŐçňňäÿL'æŰźåŇĚçőąçŘĘåŹĺïijŇæŕŤåęĆdistributeæĹŰpi

èőÍèőž

éĂŽåÿÿåŇĚäijŽèćńåőL'èčĚåĹřçşžçż§çŽĎsite-packagesçŽőå¡Ţäÿ■åŐżïijŇèůŕå¿ĎçśżäijijâĂIJ/usr/local/l
packagesâĂİãĂĆ äÿ■è£ĞïijŇè£ŹæăůåĄŽéIJĂèęĄæIJL'çőąçŘĘåŠŸæİČéŹŘåźűäÿŤä¡£çŤĺsudoåŠ¡äžd'ãĂĆ
åřśçőŮä¡ăæIJL'è£ŹæăůçŽĎæİČéŹŘåŐżæL'ğèąŇåŠ¡äžd'ïijŇä¡£çŤĺsudoåŐżåőL'èčĚäÿĂäÿłæŰřçŽĎïijŇåŔŕèČ

åőL'èčĚåŇĚåĹřçŤĺæĹůçŽőå¡Ţäÿ■éĂŽåÿÿæŸŕäÿĂäÿłæIJL'æŢĹçŽĎæŰźæąĹïijŇåőČåĚĄèőÿä¡ăåĹŽåžžă

åŘęåd'ŰïijŇä¡ăè£ŸåŔŕäžěåĹŽåžžäÿĂäÿłèŹŽæŃ§çŐŕåćČïijŇè£ŹäÿłæĹŚäžňåIJĺäÿŃäÿĂèŁĆäijŽèőšåĹřă

10.14 åĹŽåžžæŰřçŽĎPythonçŐŕåćČ

éŮőécŸ

ä¡ăæČşåĹŽåžžäÿĂäÿłæŰřçŽĎPythonçŐŕåćČïijŇçŤĺæİěåőL'èčĚæĺąåİŮåŠŇåŇĚãĂĆ
äÿ■è£ĞïijŇä¡ăäÿ■æČşåőL'èčĚäÿĂäÿłæŰřçŽĎPythonåĚŃéŽĘijŇäź§äÿ■æČşåŕźçşżçż§PythonçŐŕåćČäžğçŤ§

èğčåĘşæŰźæąĹ

ä¡ăåŘŕäžěä¡£çŤĺ `pyvenv` åŠ¡äžd'åĹŽåžžäÿĂäÿłæŰřçŽĎâĂIJèŹŽæŃ§âĂİçŐŕåćČãĂĆ
è£ŹäÿłåŠ¡äžd'èćńåőL'èčĚåIJĺPythonèğčéĞŁåŹĺåŘŇäÿĂçŽőå¡ŢïijŇæĹŰWindowsäÿŁéİćçŽĎScriptsçŽőå¡Ţäÿ

```
bash % pyvenv Spam
bash %
```

äijăçżŹ `pyvenv` åŠ¡äžd'çŽĎåŘ■å■ŰæŸŕåřĘèęĄèćńåĹŽåžžçŽĎçŽőå¡ŢåŘ■ãĂĆå¡Şèćńå
ĹŽåžžåŘŐïijŇS

```
bash % cd Spam
bash % ls
bin include lib pyvenv.cfg
bash %
```

åIJĺbinçŽőå¡Ţäÿ■ïijŇä¡ăäijŽæL'¿åĹřäÿĂäÿłåŔŕäžěä¡£çŤĺçŽĎPythonèğčéĞŁåŹĺïijŽ

```
bash % Spam/bin/python3
Python 3.3.0 (default, Oct 6 2012, 15:45:22)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more␣
↪information.
>>> from pprint import pprint
>>> import sys
>>> pprint(sys.path)
['',
'/usr/local/lib/python33.zip',
'/usr/local/lib/python3.3',
'/usr/local/lib/python3.3/plat-darwin',
'/usr/local/lib/python3.3/lib-dynload',
'/Users/beazley/Spam/lib/python3.3/site-packages']
>>>
```

è£Źäÿłèğčé ĞŁåŹÍçŽĎçĽźçĆźåřśæŸŕäżŰçŽĎsite-packagesçŻőå¡Ţèćńèő¿ç¡őäÿžæŰřåĹŻåżžçŽĎçŐŕåćČ
åęĆæđIJä¡ăèęĄåőĽèčĚçňňäÿĽæŰźåŇĚïijŇåőČäžňäijŽèćńåőĽèčĚåIJÍéĆčéĞŇïijŇèĂŇäÿ■æŸŕéĂŽåÿÿçşżçż
packagesçŻőå¡ŢãĂĆ

èőÍèőž

åĹŻåżžèŹŽæŃ§çŐŕåćČéĂŽåÿÿæŸŕäÿžäžĘåőĽèčĚåŠŇçőąçŘĘçňňäÿĽæŰźåŇĚãĂĆ
æ■čåęĆä¡ăåIJÍä¿Ńå■Řäÿ■çIJŃåĹřçŽĎéĆčæăůïijŇsys.path
åŔŸéĞŔåŇĚåŘńæÌèèĞłäžŐçşżż§PythonçŽĎçŻőå¡ŢïijŇ èĂŇ site-
packagesçŻőå¡ŢåůšçżŕèćńéĞ■åőŽä¡■åĹřäÿĂäÿłæŰřçŽĎçŻőå¡ŢãĂĆ

æIJL’äžĘäÿĂäÿłæŰřçŽĎèŹŽæŃ§çŐŕåćČïijŇäÿŃäÿĂæ■ěårśæŸŕåőĽèčĚäÿĂäÿłåŇĚçőąçŘĘåŹĺïijŇærŤå
ä¡ĘåőĽèčĚè£Źæăůç ŽĎåůěåĚůåŠŇåŇĚçŽĎæŮűåĂŹïijŇä¡ăéIJĂèęĄçąőä£Ìä¡ăä¡£çŤĺçŽĎæŸŕèŹŽæŃ§çŐŕåćČ
åőČäijŽårĘåŇĚåőĽèčĚåĹřæŰřåĹŻåżžçŽĎsite-packagesçŻőå¡Ţäÿ■åŐžãĂĆ

år¡çőąäÿĂäÿłèŹŽæŃ§çŐŕåćČçIJŃäÿŁåŐżæŸŕPythonåőĽèčĚçŽĎäÿĂäÿłåd’■åĹűïijŇ
äÿ■è£ĞåőČåőđéŹĚäÿŁåŘłåŇĚåŘńäžĘåřŠéĞŔåĞääÿłæŰĞäžűåŠŇäÿĂäžŽçňęåŘůéŞ¿æŐěãĂĆ
æĽĂæIJL’æäĞåĞĘåžŞåĞ¡æŰĞäžűåŠŇåŘŕæL’ğèąŇèğčéĞŁåŹĺéČ¡æÌěèĞłåŐ§æÌěçŽĎPythonåőĽèčĚãĂĆ
åŽăæ■d’ïijŇåĹŻåżžè£Źæăůç ŽĎçŐŕåćČæŸŕå¿Ĺåőźæ ŸŞçŽĎïijŇåźűäÿŤåĞääźŐäÿ■äijŽæűĹèĂŮæIJžåŹĺèţĎæ

éżŸèőd’æČĚåĘţäÿŃïijŇèŹŽæŃ§çŐŕåćČæŸŕçl’žçŽĎïijŇäÿ■åŇĚåŘńäżžä¡ŢéćÍåd’ŰçŽĎçňňäÿL’æŰźåžŞ
åŘŕäžěä¡£çŤĺâĂIJ–system-site-packagesâĂÌéĂL’éąźæÌěåĹŻåżžèŹŽæŃ§çŐŕåćČïijŇä¿ŃåęĆïijŽ

```
bash % pyvenv --system-site-packages Spam
bash %
```

èů§åd’ŽåĚşäžŐ pyvenv åŠŇèŹŽæŃ§çŐŕåćČçŽĎä£ąæĄŕåŘŕäžěåŔĆèĂČ PEP 405.

10.15 åĹĘåŘŠåŇĚ

éŮőécŸ

ä¡ăåůšçżŘçijŰåĘŹäžĘäÿĂäÿłæIJL’çŤĺçŽĎåžŞïijŇæČşårĘåőČåĹĘäžńçżŹåĚűäżŰäžžãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăæČşåĹĘåŘŠä¡ăçŽĎäżčçăĄïijŇçňňäÿĂäžűäžŃårśæŸŕçżŹåőČäÿĂäÿłåŤŕäÿĂçŽĎåŘ■å■Ůïij
ä¿ŃåęĆïijŇäÿĂäÿłåĚÿåđŃçŽĎåĞ¡æŢŕåžŞåŇĚäijŽçśżäijijäÿŃéÍćè£Źæăůïij Ž

```
projectname/
    README.txt
    Doc/
        documentation.txt
    projectname/
        __init__.py
        foo.py
```

```
        bar.py
        utils/
            __init__.py
            spam.py
            grok.py
    examples/
        helloworld.py
        ...
```

èęĄèőľä¡ăçŽĎåŇĚåŘŕäžěåŔŚåÿČåĞžåŐżïijŇéęŰåĚĹä¡ăèęĄçijŰåĘŹäÿĂäÿł `setup.py` ïijŇçśżäijijäÿŃéİćè£ŹæăůïijŽ

```
# setup.py
from distutils.core import setup

setup(name='projectname',
    version='1.0',
    author='Your Name',
    author_email='you@youraddress.com',
    url='http://www.you.com/projectname',
    packages=['projectname', 'projectname.utils'],
)
```

äÿŃäÿĂæ■ěïijŇářśæŸŕåĹŻåžžäÿĂäÿł `MANIFEST.in` æŰĞäžűïijŇåĹŮåĞžæĽĂæIJL'åIJĺä¡äçŽĎåŇĚäÿ

```
# MANIFEST.in
include *.txt
recursive-include examples *
recursive-include Doc *
```

çąőä£İ `setup.py` åŠŇ `MANIFEST.in` æŰĞäžűæŤ¿åIJĺä¡äçŽĎåŇĚçŽĎæIJĂéąűçžğçŻőå¡Ţäÿ■ãĂĆäÿĂæŮęä¡ăåůšçżŔåĄŽäžĘè£ŹäžŻïijŇä¡ăářśåŔŕäžěåČŔäÿŃéİćè£ŹæăůæĽğèąŇåŚ¡äžď'æİěåĹŻåžžäÿĂäÿłæžŘ

```
% bash python3 setup.py sdist
```

åőČäijŽåĹŻåžžäÿĂäÿłæŰĞäžűæŕŤåęĆ"projectname-1.0.zip" æĹŰ âĂIJprojectname-1.0.tar.gzâĂİ, åĚůä¡Şä¿İètŰäžŐä¡äçŽĎçşżçż§åźşåŔřãĂĆåęĆæđIJäÿĂåĹĞæ■čåÿÿïijŇè£ŹäÿłæŰĞäžűåřśåŔŕäžěåŔŚéĂĄçżŹåĹńäžžä¡£çŤĺæĹŰèĂĚäÿŁäijäèĞş Python Package Index.

èőĺèőž

åŕźäžŐçžŕPythonäžčçăĄïijŇçijŰåĘŹäÿĂäÿłæŹőéĂŽçŽĎ `setup.py` æŰĞäžűéĂŽåÿÿå¿ĹçőĂå■ŢãĂĆ äÿĂäÿłåŔŕèČ¡çŽĎéŮőécŸæŸŕä¡ăå£ĚéążæĽŃåŁĺåĹŮåĞžæĽĂæIJL'æđĎæäÿĂäÿłåÿÿèğĄéŤŹèŕŕåřśæŸŕäžĚäžĚåŔłåĹŮåĞžäÿĂäÿłåŇĚçŽĎæIJĂéąűçžğçŻőå¡ŢïijŇå£ŸèőřäžĘåŇĚåŘńåŇè£Źäź§æŸŕäÿžäžĂäźĹåIJĺ `setup.py` äÿ■åŕźäžŐåŇĚçŽĎèŕť'æŸŐåŇĚåŘńäžĘåĹŮèąĺ `packages=['projectname', 'projectname.utils']`

åď'ğéČĺåĹĘPythonçĺŃåžŔåŚŸéČ¡ç§ěéĄŞïijŇæIJL'å¿Ĺåď'ŽçňňäÿL'æŰźåŇĚçőąçŘĘåŹĺä¿ŻéĂL'æŃľïijŇæIJL'äžŻæŸŕäÿžäžĘæŻ£äžčæăĞåĞĘåžŞäÿ■çŽĎdistutilsãĂĆæşĺæĎŔåęĆæđIJä¡ää¿ÌètŰè£ŹäžŻåŇĚïijŇ

çŤÍæĹůåŔŕèČ¡äÿ■èČ¡åőL'èčĚä¡ăçŽĎè¡ŕäżűïijŇéŹď'éİđäžŰäžňåůšçżŔäžŃåĚĹåőL'èčĚè£ĞæL'ĂéIJĂèęĄçŽĎ
æ■čåŽăåęĆæ■ď'ïijŇä¡ăæŻť'åžŤèŕěæŮűåĹżèőřä¡Ŕèű£çőĂå■Ţèű£åě¡çŽĎéĄŞçŘĘãĂĆ
æIJĂåě¡èől'ä¡ăçŽĎäžčçăĄä¡£çŤÍæăĞåĞĘçŽĎPython 3åőL'èčĚãĂĆ
åęĆæđIJåĚűäžŰåŇĚäź§éIJĂèęĄçŽĎèŕİïijŇåŔŕäžěéĂŽè£ĞäÿĂäÿłåŔŕéĂL'éąźæİěæŤŕæŇĄãĂĆ

åŕźäžŐæűL'åŔŁåĹřCæL'l'åśŢçŽĎäžčçăĄæL'ŞåŇĚäÿŐåĹĘåŔŚåŕśæŻť'åď'■æİĆçĆźäžĘãĂĆ
çňň15çńăåŕźåĚşäžŐCæL'l'åśŢçŽĎè£ŹæŰźéİćç§ěèŕĘæIJL'äÿĂäžŻèŕęçżĘèőšèğčïijŇçL'źåĹńæŸŕåIJĺ15.2åřŔèł

çňňå■ĄäÿĂçńäïijŽç¡ŚçżIJäÿŐWebçijŰçÍŃ

æIJňçńăæŸŕåĚşäžŐåIJĺç¡ŚçżIJåžŢçŤĺåŠŇåĹĘåÿČåijŔåžŢçŤĺäÿ■ä¡£çŤĺçŽĎåŘĎçğ■äÿžécŸãĂĆäÿžécŸã

Contents:

11.1 ä¡lJäÿžåőćæĹůçńŕäÿŐHTTPælJ■åŁąäžď'äžŠ

éŮőécŸ

ä¡ăéIJĂèęĄéĂŽè£ĞHTTPå■ŔèőőäžěåőćæĹůçńŕçŽĎæŰźåijŔèő£éŮőåď'Žçğ■æIJ■åŁąãĂĆä¿ŃåęĆïijŇäÿ

èğčåĘşæŰźæąĹ

åŕźäžŐçőĂå■ŢçŽĎäžŃæČĚæİěèŕť'ïijŇéĂŽåÿÿä¡£çŤÍ `urllib.request` æĺąåİŮåřšåď'§äžĘãĂĆä¿ŃåęĆïijŇåŔŚéĂĄäÿĂäÿłçőĂå■ŢçŽĎHTTP GETèŕůæśĆåĹřè£IJçÍŃçŽĎæIJ■åŁąäÿŁïijŇåŔŕäžěè£ŹæăůåĄŽïijŽ

```
from urllib import request, parse

# Base URL being accessed
url = 'http://httpbin.org/get'

# Dictionary of query parameters (if any)
parms = {
   'name1' : 'value1',
   'name2' : 'value2'
}

# Encode the query string
querystring = parse.urlencode(parms)

# Make a GET request and read the response
u = request.urlopen(url+'?' + querystring)
resp = u.read()
```

åęĆæđIJä¡ăéIJĂèęĄä¡£çŤÍPOSTæŰźæşŢåIJÍèŕůæśĆäÿžä¡Şäÿ■åŔŚéĂĄæ§ěèŕćåŔĆæŢřïijŇåŔŕäžěåřĘåŔ
`urlopen()` åĞ¡æŢřïijŇåřšåČŔè£ŹæăůïijŽ

```
from urllib import request, parse

# Base URL being accessed
url = 'http://httpbin.org/post'

# Dictionary of query parameters (if any)
parms = {
   'name1' : 'value1',
   'name2' : 'value2'
}

# Encode the query string
querystring = parse.urlencode(parms)

# Make a POST request and read the response
u = request.urlopen(url, querystring.encode('ascii'))
resp = u.read()
```

åęĆædIJä¡ăéIJĂèęĄåIJĺåŔŚåĞžçŽĎèřůæśĆäÿ■æŔŘä¿ŻäÿĂäžŻèĞłåőŽäźĽçŽĎHTTPåď't'ïijŇä¿ŃåęĆä£
`user-agent` å■Ůæőţ,åŔŕäżěåĹŻåžžäÿĂäÿłåŇĚåŘńå■Ůæőţå̆ijçŽĎå■ŮåĚÿïijŇåźűåĹŻåžžäÿĂäÿłRequestå
`urlopen()` ïijŇåęĆäÿŃïijŽ

```
from urllib import request, parse
...

# Extra headers
headers = {
    'User-agent' : 'none/ofyourbusiness',
    'Spam' : 'Eggs'
}

req = request.Request(url, querystring.encode('ascii'),␣
↪headers=headers)

# Make a request and read the response
u = request.urlopen(req)
resp = u.read()
```

åęĆædIJéIJĂèęĄäžď'äžŠçŽĎæIJ■åŁąæŕŤäÿŁéİćçŽĎä¿Ńå■ŘéČ¡èęĄåď'■æİĆïijŇäź§èőÿåžŤèŕěåŐżçIJŃç
requests åžŞïijĹhttps://pypi.python.org/pypi/requestsïijL'ãĂĆä¿ŃåęĆïijŇäÿŃéİćè£Źäÿłçđ'žä¿ŃéĞĞçŤĺreques

```
import requests

# Base URL being accessed
url = 'http://httpbin.org/post'

# Dictionary of query parameters (if any)
parms = {
   'name1' : 'value1',
   'name2' : 'value2'
```

```
}

# Extra headers
headers = {
    'User-agent' : 'none/ofyourbusiness',
    'Spam' : 'Eggs'
}

resp = requests.post(url, data=parms, headers=headers)

# Decoded text returned by the request
text = resp.text
```

åĚşäžŐrequestsåžŞïijŇäÿĂäÿłåĂijå¿ŮäÿĂæŔŘçŽĎçĽźæĂğářśæŸŕåőČèČ¡äžěåď Žçğ■æŰźåijŔäžŐèŕůæ
`resp.text` åÿęçżŹæĹŚäžňçŽĎæŸŕäžěUnicodeèğčçăĄçŽĎåŞ■åžŤæŰĞæIJñãĂĆä¡ĘæŸŕïijŇåęĆæđIJåŐżèč
`resp.content` ïijŇářśäijŽå¿ŮåĹřåŐ§ăğŃçŽĎäžŇè£ŻåĹűæŢřæ■őãĂĆåŔęäÿĂæŰźéĺćïijŇåęĆæđIJèő£éŮć
`resp.json` ïijŇéČčäźĹářśäijŽå¿ŮåĹřJSONæăijåijŔçŽĎåŞ■åžŤåĘĚåőźãĂĆ

äÿŃéİćè£Źäÿłçdˇžä¿ŃåĹlˇçŤÍ `requests` åžŞåŔŚèţůäÿĂäÿłHEAD-èŕůæśĆïijŇåźűäžŐåŞ■åžŤäÿ■æŔŘåŔŰåĞžäÿĂäžŻHTTPådˇtˇæŢřæ■őçŽĎå■Ůæőţïij Ž

```
import requests

resp = requests.head('http://www.python.org/index.html')

status = resp.status_code
last_modified = resp.headers['last-modified']
content_type = resp.headers['content-type']
content_length = resp.headers['content-length']
```

äÿŃéİćæŸŕäÿĂäÿłåĹlˇçŤÍrequestséĂŽè£Ğå§žæIJňèődˇèŕĄçŹżå¡ŢPypiçŽĎä¿Ńå■ŘïijŽ

```
import requests

resp = requests.get('http://pypi.python.org/pypi?:action=login',
                    auth=('user','password'))
```

äÿŃéİćæŸŕäÿĂäÿłåĹlˇçŤÍrequestsåřĘHTTP cookiesäžŐäÿĂäÿłèŕůæśĆäijåéĂŠåĹřåŔęäÿĂäÿłçŽĎä¿Ńå■

```
import requests

# First request
resp1 = requests.get(url)
...

# Second requests with cookies received on first requests
resp2 = requests.get(url, cookies=resp1.cookies)
```

æIJĂåŘŐä¡Ęåźűéİđæ IJĂäÿ■éĞ■èęĄçŽĎäÿĂäÿłä¿Ńå■ŘæŸŕçŤÍrequestsäÿŁäijååĘĚåőźïijŽ

```
import requests
url = 'http://httpbin.org/post'
files = { 'file': ('data.csv', open('data.csv', 'rb')) }

r = requests.post(url, files=files)
```

èőÍèőž

åŕźäžŐçIJ§çŽĎå¿ĹçőĂå■ŢHTTPåőćæĹůçńŕäžčçăĄïijŇçŤĺåĘĚç¡őçŽĎ `urllib` æĺąåİŮéĂŽåÿÿåřśèűşåď§äžĘãĂĆä¡ĘæŸŕïijŇåęĆædIJä¡ăèęĄåĄŽçŽĎäÿ■äžĚäžĚåŔłæŸŕçőĂå■ŢçŽĎGETæĹ `requests` åď ğæŸ¿èžńæĽŃçŽĎæŮűåĂŹäžĘãĂĆ

ä¿ŃåęĆïijŇåęĆædIJä¡ăåĘşåőŽåİŽæŇĄä¡£çŤĺæăĞåĞĚçŽĎçĺŃžŕåžŞèĂŇäÿ■èĂČèŹŚåČŔ `requests` è£ŹæăůçŽĎçňňäÿĽæŰźåžŞïijŇéĆčäźĹäź§èőÿåřśäÿ■å¿Ůäÿ■ä¡£çŤĺåžŢåśĆçŽĎ `http.client` æĺąåİŮæİěåőđçŐřèĞłåůśçŽĎäžčçăĄãĂĆærŤæŰźèŕť'ïijŇäÿŃéİćçŽĎäžčçăĄåśŢçď'žäžĘåęĆä

```
from http.client import HTTPConnection
from urllib import parse

c = HTTPConnection('www.python.org', 80)
c.request('HEAD', '/index.html')
resp = c.getresponse()

print('Status', resp.status)
for name, value in resp.getheaders():
    print(name, value)
```

åŘŇæăůåIJřïijŇåęĆædIJå£Ěéąžçij ŰåĘŹæűĽåŔŁäžčçŘĘãĂĄèőď'èŕĄãĂĄcookiesäžěåŔŁåĚűäžŰäÿĂäžŽ `urllib` åřśæŸ¿å¿ŮçĽźåĹńåĹńæĽ■åŠŇåŢřåŮęãĂĆærŤæŰźèŕť'ïijŇäÿŃéİćè£Źäÿłçď'žä¿Ńåődç ŐřåIJĺPython

```
import urllib.request

auth = urllib.request.HTTPBasicAuthHandler()
auth.add_password('pypi','http://pypi.python.org','username',
↪'password')
opener = urllib.request.build_opener(auth)

r = urllib.request.Request('http://pypi.python.org/pypi?
↪:action=login')
u = opener.open(r)
resp = u.read()

# From here. You can access more pages using opener
...
```

åİęçŹ¡èŕť'ïijŇæĽĂæIJĽçŽĎè£ŹäžŽæŞ■ä¡IJåIJĺ `requests` åžŞäÿ■éČ¡åŔŸå¿ŮçőĂå■ŢçŽĎåď'ŽãĂĆ

åIJĺåijĂåŔŚè£ĞçĺŃäÿ■æţŃèŕŢHTTPåőćæĹůçńŕäžčçăĄåÿÿåÿÿæŸŕå¿Ĺäžď'äžžæšőäÿğçŽĎïijŇåŽääÿžæĽ //httpbin.orgïijĽãĂĆè£ŹäÿłçńŹçĆźäijŽæŐěæŤűåŔŚåĞžçŽĎèŕůæśĆïijŇçĎűåŘŐäžěJSONçŽĎå¡ćåijŔåřĘçŽÿ

```
>>> import requests
>>> r = requests.get('http://httpbin.org/get?name=Dave&n=37',
...     headers = { 'User-agent': 'goaway/1.0' })
>>> resp = r.json
>>> resp['headers']
{'User-Agent': 'goaway/1.0', 'Content-Length': '', 'Content-Type': '
 ↪',
'Accept-Encoding': 'gzip, deflate, compress', 'Connection':
'keep-alive', 'Host': 'httpbin.org', 'Accept': '*/*'}
>>> resp['args']
{'name': 'Dave', 'n': '37'}
>>>
```

åIJÍèęĄåŘŇäÿĂäÿłçIJ§æ■čçŽĎçńŹçĆźè£ŻèąŇäžď'äžŠåĽ■ïijŇåĚĹåIJÍ httpbin.org è£Źæăůç ŽĎç¡ŚçńŹäÿ ŁåĄŽåőđéłŇåÿÿåÿÿæŸŕåŔŕåŔŰçŽĎåŁđæşŢãĂĆåřď'åĚűæŸŕå¡ŞæĹŚäžňéÍćåŕź3æňąçŹ

år¡çőąæIJňèŁĆæšąæIJĽ'æűĽ'åŔŁïijŇ `request` åžŞè£Ÿåŕźèőÿåď'ŽéńŸçžğçŽĎHTTPåőćæĹůçńŕå■Ŕèőő `requests` æĺąåİŮçŽĎæŰĞæąčïijĹhttp://docs.python-requests.org)èť'ÍéĞŔå¿ĹéńŸïijĹåİęçŹ¡èŕť'æŕŤåIJÍè£Źç§■ç§■çŽĎäÿĂèŁĆçŽĎçŕĞåźĚäÿ■æĽ'ĂæŔŘä¿ŻçŽĎäżżä¡Ţä£ą

11.2 åĹŻåżžTCPæIJ■åŁąåŹÍ

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłæIJ■åŁąåŹÍïijŇéĂŽè£ĞTCPå■Ŕèőőåš ŇåőćæĹůçńŕéĂŽä£ąãĂĆ

èğčåĘşæŰźæąĹ

åĹŻåżžäÿĂäÿłTCPæIJ■åŁąåŹÍçŽĎäÿĂäÿłçőĂå■ŢæŰźæşŢæŸŕä¡£çŤÍ `socketserver` åžŞãĂĆä¿ŃåęĆïijŇäÿŃéÍćæŸŕäÿĂäÿłçőĂå■ŢçŽĎåžŤç■ŤæIJ■åŁąåŹÍïijŽ

```
from socketserver import BaseRequestHandler, TCPServer

class EchoHandler(BaseRequestHandler):
    def handle(self):
        print('Got connection from', self.client_address)
        while True:

            msg = self.request.recv(8192)
            if not msg:
                break
            self.request.send(msg)

if __name__ == '__main__':
    serv = TCPServer(('', 20000), EchoHandler)
    serv.serve_forever()
```

åIJÍè£Źæőţäžčçă Ąäÿ■ïijÑä¡ăåőŽäźL'äžĘäÿĂäÿłçL'źæőŁçŽĎåď'ĎçŘĘçśżïijÑåőđçŐřäžĘäÿĂäÿł
`handle()` æŰźæşŢïijÑçŤÍæÌěäÿžåőćæĹůçńřè£đæŐěæIJ■åŁąãĂĆ `request` åśđæĂğæŸŕåőćæĹůçńřsocketïijÑ`client_address` æIJL'åőćæĹůçńřåIJřåÌĂãĂĆ äÿžäžĘæţÑèŕŢè£ŹäÿłæIJ■åŁąåŹÍïijÑè£ŘèąÑåőČåźűæL'ŞåijĂåŔęåď'ŰäÿĂäÿłPythonè£ŽçÍÑè£đæŐěè£Źäÿłæ

```
>>> from socket import socket, AF_INET, SOCK_STREAM
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s.connect(('localhost', 20000))
>>> s.send(b'Hello')
5
>>> s.recv(8192)
b'Hello'
>>>
```

å¿Ĺåď'ŽæŮűåĂŹïijÑåŔŕäžěå¿Ĺåőźæ ŸŞçŽĎåőŽäźL'äÿĂäÿłäÿ■åŘÑçŽĎåď'ĎçŘĘåŹÍãĂĆäÿÑéÍćæŸŕäÿĂ
`StreamRequestHandler` å§žçśžåřĘäÿĂäÿłçśżæŰĞäžűæŐěåŘčæŤ¿ç¡őåIJĺäžŢåśĆsocketäÿŁçŽĎä¿Ńå■Ř

```
from socketserver import StreamRequestHandler, TCPServer

class EchoHandler(StreamRequestHandler):
    def handle(self):
        print('Got connection from', self.client_address)
        # self.rfile is a file-like object for reading
        for line in self.rfile:
            # self.wfile is a file-like object for writing
            self.wfile.write(line)

if __name__ == '__main__':
    serv = TCPServer(('', 20000), EchoHandler)
    serv.serve_forever()
```

èőÍèőž

`socketserver` åŔŕäžěèőľ'æĹŚäžňå¿Ĺåőźæ ŸŞçŽĎåĹŻåžžçőĂå■ŢçŽĎTCPæIJ■åŁąåŹÍãĂĆ ä¡ĘæŸřïijÑä¡ăéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijÑéżŸèőď'æČĚåĘţäÿÑè£ŹçğæIJ■åŁąåŹÍæŸŕå■Ţçž£çÍÑçŽĎïijÑäÿ åęĆæđIJä¡ăæČşåď'ĎçŘĘåď'ŽäÿłåőćæĹůçńřïijÑåŔŕäžěåĹİåğÑåÑŰäÿĂäÿł `ForkingTCPServer` æĹŰèĂĚæŸŕ `ThreadingTCPServer` åŕźèśąãĂĆä¿ŃåęĆïijŽ

```
from socketserver import ThreadingTCPServer


if __name__ == '__main__':
    serv = ThreadingTCPServer(('', 20000), EchoHandler)
    serv.serve_forever()
```

ä¡£çŤÍforkæĹŰçž£çÍÑæIJ■åŁąåŹÍæIJL'äÿłæ¡IJåIJÍéŮőécŸåŕśæŸŕåőČäžñäijŽäÿžæŕŔäÿłåőćæĹůçńřè£đæ çŤśäžŐåőćæĹůçńřè£đæŐěæŢřæŸŕæšąæIJL'éŹŘåĹűçŽĎïijÑåŻäæ■ď'äÿĂäÿłæĄűæĎŔçŽĎéŹŠåőćåŔŕäžěåŘÑ

åęĆæđIJä¡ăæÑĚå£Čè£ŹäÿłéŮőécŸïijÑä¡ăåŔŕäžěåĹŻåžžäÿĂäÿłécĎåĚĹåĹĘéĚ■åď'ğåřŔçŽĎåůěä¡IJçž£ç ä¡ăåĚĹåĹŻåžžäÿĂäÿłæŹőéĂŽçŽĎéIđçž£çÍÑæIJ■åŁąåŹÍïijÑçĎűåŘŐåIJÍäÿĂäÿłçž£çÍÑæśăäÿ■ä¡£çŤÍ

serve_forever() æŰźæşŢæİěåŘŕåŁĺåőČäžňãĂĆ

```
if __name__ == '__main__':
    from threading import Thread
    NWORKERS = 16
    serv = TCPServer(('', 20000), EchoHandler)
    for n in range(NWORKERS):
        t = Thread(target=serv.serve_forever)
        t.daemon = True
        t.start()
    serv.serve_forever()
```

äÿĂèĹñæİěèőšïijÑäÿĂäÿł TCPServer åIJĺåőđä¿ŃåŇŰçŽĎæŮűåĂŹäijŽçżŚåőŽåźűæ£Äæť żçŻÿåžŢçŽ
socket ãĂĆ äÿ■è£ĞïijÑæIJĽæŮűåĂŹä¡ăæČşéĂŽè£Ğèő¿ç¡őæ§ŘäžŻéĂĽéąźåŐżèřČæŢť åžŢäÿŃçŽĎ
socket‘ ïijÑåŘŕäžěèő¿ç¡őåŔĆæŢř bind_and_activate=False ãĂĆåęĆäÿŇïijŽ

```
if __name__ == '__main__':
    serv = TCPServer(('', 20000), EchoHandler, bind_and_
↪activate=False)
    # Set up various socket options
    serv.socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR,␣
↪True)
    # Bind and activate
    serv.server_bind()
    serv.server_activate()
    serv.serve_forever()
```

äÿŁéİćçŽĎ socket éĂĽéąźæŸŕäÿĂäÿłéİđåÿÿæŹőéĄ■çŽĎéĚ■ç¡őéąźïijÑåőČåĚĄèõÿæIJ■åŁąåŹĺéĞ■æ
çŤśäžŐèęĄèćńçżŔåÿÿä¡£çŤĺåĹřïijÑåőČèćńæŤ¿ç¡őåĹřçśżåŔŸéĞŔäÿ■ïijÑåŔŕäžěçŽťˇæŐěåIJĺ
TCPServer äÿŁéİćèő¿ç¡õãĂĆ åIJĺåőđä¿ŃåŇŰæIJ■åŁąåŹĺçŽĎæŮűåĂŹåŐżèő¿ç¡őåőČçŽĎåĂijïijÑåęĆäÿŇ

```
if __name__ == '__main__':
    TCPServer.allow_reuse_address = True
    serv = TCPServer(('', 20000), EchoHandler)
    serv.serve_forever()
```

åIJĺäÿŁéİćçđ’žä¿Ńäÿ■ïijÑæĹŚäžňæijŤçđ’žäžĘäÿđ’çğ■äÿ■åŘŇçŽĎåđ’ĎçŘĘåŹĺå§žçśżïijĹ
BaseRequestHandler åŠŇ StreamRequestHandler ïijL’ãĂĆ
StreamRequestHandler æŻť’åŁăçĄţæť’żçĆźïijÑèČ¡éĂŽè£Ğèő¿ç¡őåĚűäžŰçŽĎçśżåŔŸéĞŔæİěæŤŕæŇ

```
import socket

class EchoHandler(StreamRequestHandler):
    # Optional settings (defaults shown)
    timeout = 5                    # Timeout on all socket␣
↪operations
    rbufsize = -1                  # Read buffer size
    wbufsize = 0                   # Write buffer size
    disable_nagle_algorithm = False  # Sets TCP_NODELAY socket␣
↪option
    def handle(self):
```

```
        print('Got connection from', self.client_address)
        try:
            for line in self.rfile:
                # self.wfile is a file-like object for writing
                self.wfile.write(line)
        except socket.timeout:
            print('Timed out!')
```

æIJĂåŘŐïijŇè£ŸéIJĂèęĄæşĺæĎŔçŽĎæŸŕåůĺåď˘ğéČĺåĹĘPythonçŽĎéńŸåśÇç¡ŚçżIJæĺąåİŮïijĹæŕŤåęĆI
RPCç■L’ïijL’éČ¡æŸŕåżžçńŇåIJÍ socketserver åŁ§èČ¡äźŇäÿŁãĂĆ
äź§åřśæŸŕèřť’ïijŇçŻť’æŐëä¡£çŤÍ socket åžŞæİěåőđçŐřæIJ■åŁąåŹĺäź§åźűäÿ■æŸŕå¿ĹéŽ¿ãĂĆ
äÿŇéİćæŸŕäÿĂäÿłä¡£çŤÍ socket çŻť’æŐëçijŰçÍŇåőđçŐřçŽĎäÿĂäÿłæIJ■åŁąåŹĺçőĂå■Ţä¿Ňå■ŘïijŽ

```
from socket import socket, AF_INET, SOCK_STREAM

def echo_handler(address, client_sock):
    print('Got connection from {}'.format(address))
    while True:
        msg = client_sock.recv(8192)
        if not msg:
            break
        client_sock.sendall(msg)
    client_sock.close()

def echo_server(address, backlog=5):
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(address)
    sock.listen(backlog)
    while True:
        client_sock, client_addr = sock.accept()
        echo_handler(client_addr, client_sock)

if __name__ == '__main__':
    echo_server(('', 20000))
```

11.3 åĹŻåżžUDPæIJ■åŁąåŹÍ

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłå§žäžŐUDPå■Řèőőç ŽĎæIJ■åŁąåŹĺæİěäÿŐåőćæĹůçńŕéĂŽă£ąãĂĆ

èğčåĘşæŰźæąĹ

èů§TCPäÿĂæăůïijŇUDPæIJ■åŁąåŹĺäź§åŘŕäžěéĂŽè£ĞäĄ£çŤÍ socketserver
åžŞå¿Ĺåőźæ ŸŞçŽĎèćńåĹŻåżžãĂĆ ä¿ŇåęĆïijŇäÿŇéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎæŮűéŮť’æIJ■åŁąåŹĺïijŽ

```
from socketserver import BaseRequestHandler, UDPServer
import time

class TimeHandler(BaseRequestHandler):
    def handle(self):
        print('Got connection from', self.client_address)
        # Get message and client socket
        msg, sock = self.request
        resp = time.ctime()
        sock.sendto(resp.encode('ascii'), self.client_address)

if __name__ == '__main__':
    serv = UDPServer(('', 20000), TimeHandler)
    serv.serve_forever()
```

èů§äźŃåL’■äÿĂæăůïijŇä¡ăåĚĹåőŽäźL’äÿĂäÿłåőđçŐř handle() çL’źæőŁæŰźæşŢçŽĎçśżïijŇäÿžåőćæĹůçńŕè£đæŐěæIJ■åŁąãĂĆ è£ŹäÿłçśżçŽĎ request åśđæĂğæŸŕäÿĂäÿłåŇĚåŘńäžĘæŢřæ■őæŁěåŠŇåžŢåśĆsock-etåŕźèśąçŽĎåĚČçżĎãĂĆclient_address åŇĚåŘńäžĘåőćæĹůçńŕåIJŕåİĂãĂĆ

æĹŚäżňæİěæţŃèŕŢäÿŃè£ŹäÿłæIJ■åŁąåŹĺïijŇéęŰåĚĹè£ŘèąŇåőČïijŇçĎűåŘŐæL’Şåij Ăå Ŕęåd’ŰäÿĂäÿł

```
>>> from socket import socket, AF_INET, SOCK_DGRAM
>>> s = socket(AF_INET, SOCK_DGRAM)
>>> s.sendto(b'', ('localhost', 20000))
0
>>> s.recvfrom(8192)
(b'Wed Aug 15 20:35:08 2012', ('127.0.0.1', 20000))
>>>
```

èőĺèőž

äÿĂäÿłåĚÿåđŃçŽĎUDPæIJ■åŁąåŹĺæŐěæŤűåĹřè¿¿çŽĎæŢřæ■őæŁě(æűĹæĄŕ)åŠŇåőćæĹůçńŕåIJŕåİĂãĂ åőČèęĄçżŹåőćæĹůçńŕåŽđåŘŠäÿĂäÿłæŢřæ■őæŁěãĂĆåŕźäžŐæŢřæ■őæŁěçŽĎäijåéĂĄïijŇ ä¡ăåžŤèŕěä¡£çŤĺsocketçŽĎ sendto() åŠŇ recvfrom() æŰźæşŢãĂĆ åř¡çőąäijåçż§çŽĎ send() åŠŇ recv() äźś§åŘŕäžěè¿¿åĹřåŘŇæăůçŽĎæŢĹæđIJïijŇ ä¡ĘæŸŕåL’■éİççŽĎäÿd’äÿłæŰźæşŢåŕźäžŐUDPè£đæŐěèĂŇèİĂæŻt’æŹőéĄ■ãĂĆ

çŤśäžŐæšąæIJL’åžŢåśĆçŽĎè£đæŐěïijŇUPDæIJ■åŁąåŹĺçŻÿåŕźäžŐTCPæIJ■åŁąåŹĺæİèèőšåőđçŐřèţůæİ äÿ■è£ĞïijŇUDPåd’l’çŤ§æŸŕäÿ■åŘŕéİăçŽĎïijĹåŻääÿžéĂŽä£ąæšąæIJL’åżžçńŇè£đæŐěïijŇæűĹæĄŕåŘŕèČ¡äÿ åŽăæ■d’éIJĂèęĄçŤśä¡ăèĞłåůśæİěåĘşåőŽèŕěæĂŐæăůåd’ĎçŘĘäÿćåd’śæűĹæĄŕçŽĎæČĚåĘţãĂĆè£Źäÿłåůšçżİ äÿ■è£ĞéĂŽåÿÿæİèèŕť’ïijŇåęĆæđIJåŘŕéİăæĂğåŕźäžŐä¡ăçÍŃåžŘå¿ĹéĞ■èęĄïijŇä¡ăéIJĂèęĄåĂ§åŁl’äžŐåžŘåÍ UDPéĂŽåÿÿèćńçŤĺåIJĺéĆčäžŽåŕźäžŐåŘŕéİăäijăè¿ŞèęĄæśĆäÿ■æŸŕå¿ĹéńŸçŽĎåIJžåŘĹãĂĆä¿ŃåęĆïijŇåIJĺå æŮăéIJĂè£ŤåŽđæĄćåd’■äÿćåd’śçŽĎæŢřæ■őåŇĚïijĹçÍŃåžŘåŘłéIJĂçőĂå■ŢçŽĎå£¡çŢěåőČåźűçżğçż■åŘŠåL

UDPServer çśżæŸŕå■Ţçž£çÍŃçŽĎïijŇäź§åřśæŸŕèŕť’äÿĂæńąåŘłèČ¡äÿžäÿĂäÿłåőćæĹůçńŕè£đæŐěæIJ■ åőđéŹĚä¡£çŤĺäÿ■ïijŇè£ŹäÿłæŮăèőžæŸŕåŕźäžŐUDPè£ŸæŸŕTCPéČ¡äÿ■æŸŕäžĂäźĹåd’ğéŮőécŸãĂĆ åęĆæđIJä¡ăæČşèęĄåźűåŘŠæŞ■ä¡IJïijŇåŘŕäžěåőđä¿ŃåŇŰäÿĂäÿł ForkingUDPServer æĹŰ ThreadingUDPServer åŕźèśąïijŽ

```
from socketserver import ThreadingUDPServer

   if __name__ == '__main__':
    serv = ThreadingUDPServer(('',20000), TimeHandler)
    serv.serve_forever()
```

çŻťæŐëä¡£çŤÍ socket æÌëåőđçŐřäÿĂäÿłUDPæIJ■åŁąåŹĺäź§äÿ■éŽ¿ïijŇäÿŃéÌćæŸŕäÿĂäÿłä¿Ńå■ŘïijŽ

```
from socket import socket, AF_INET, SOCK_DGRAM
import time

def time_server(address):
    sock = socket(AF_INET, SOCK_DGRAM)
    sock.bind(address)
    while True:
        msg, addr = sock.recvfrom(8192)
        print('Got message from', addr)
        resp = time.ctime()
        sock.sendto(resp.encode('ascii'), addr)

if __name__ == '__main__':
    time_server(('', 20000))
```

11.4 éĂŽè£ĞCIDRåIJřåÌĂçŤ§æĹŘåŕźåžŤçŽĎIPåIJřåÌĂéŻĘ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłCIDRç¡ŚçżIJåIJřåÌĂæŕŤåęĆâĂIJ123.45.67.89/27âĂÌïijŇä¡ăæČşåřĘåĚűè¡ňæ■ćæĹŘåőČa
ïijĹæŕŤåęĆïijŇâĂIJ123.45.67.64âĂÌ, âĂIJ123.45.67.65âĂÌ, âĂę, âĂIJ123.45.67.95âĂÌ)ïijL'

èğčåĘşæŰźæąĹ

åŘŕäżěä¡£çŤÍ ipaddress æĺąåÌŮå¿ĹåőźæŸŞçŽĎåőđçŐřè£Źæăůçž Ďèőąçő ŮãĂĆä¿ŃåęĆïijŽ

```
>>> import ipaddress
>>> net = ipaddress.ip_network('123.45.67.64/27')
>>> net
IPv4Network('123.45.67.64/27')
>>> for a in net:
...     print(a)
...
123.45.67.64
123.45.67.65
123.45.67.66
123.45.67.67
123.45.67.68
...
```

```
123.45.67.95
>>>

>>> net6 = ipaddress.ip_network('12:3456:78:90ab:cd:ef01:23:30/125')
>>> net6
IPv6Network('12:3456:78:90ab:cd:ef01:23:30/125')
>>> for a in net6:
...     print(a)
...
12:3456:78:90ab:cd:ef01:23:30
12:3456:78:90ab:cd:ef01:23:31
12:3456:78:90ab:cd:ef01:23:32
12:3456:78:90ab:cd:ef01:23:33
12:3456:78:90ab:cd:ef01:23:34
12:3456:78:90ab:cd:ef01:23:35
12:3456:78:90ab:cd:ef01:23:36
12:3456:78:90ab:cd:ef01:23:37
>>>
```

Network äź§åĚąèőÿåČŕæŢřçżĎäÿĂæăůçŽĎçť´ćåijŢåŔŰåĂijïijŇä¿ŃåęĆïijŽ

```
>>> net.num_addresses
32
>>> net[0]

IPv4Address('123.45.67.64')
>>> net[1]
IPv4Address('123.45.67.65')
>>> net[-1]
IPv4Address('123.45.67.95')
>>> net[-2]
IPv4Address('123.45.67.94')
>>>
```

åŔęåď´ŰïijŇä¡ăè£ŸåŔŕäżěæL'ğèąŇç¡ŚçżIJæĹŘåŚŸæčĂæ§ëäźŃçśżçŽĎæŞ■ä¡IJïijŽ

```
>>> a = ipaddress.ip_address('123.45.67.69')
>>> a in net
True
>>> b = ipaddress.ip_address('123.45.67.123')
>>> b in net
False
>>>
```

äÿĂäÿłIPåIJřåİĂåŠŇç¡ŚçżIJåIJřåİĂèČ¡éĂŽè£ĞäÿĂäÿłIPæŐěåŔčæİěæŇĞåőŽïijŇä¿ŃåęĆïijŽ

```
>>> inet = ipaddress.ip_interface('123.45.67.73/27')
>>> inet.network
IPv4Network('123.45.67.64/27')
>>> inet.ip
IPv4Address('123.45.67.73')
```

```
>>>
```

èőÍèőž

ipaddress æÍąåİŮæIJL'å¿Ĺåď'ŽçśżåŔŕäżěèąĺçď'žIPåIJřåİĂãĂĄç¡ŚçżIJåŠŇæŐěåŔčãĂĆ
å¡Šä¡ăéIJĂèęĄæŞ■ä¡IJç¡ŚçżIJåIJřåİĂïijĹæŕŤåęĆèğčæđŘãĂĄæL'Šå■řãĂĄéłŇèŕĄç■L'ïijL'çŽĎæŮűåĂŹäijŽå

èęĄæşĺæĎŘçŽĎæŸŕïijŇipaddress æÍąåİŮèů§åĚűäžŰäÿĂäžŻåŠŇç¡ŚçżIJçŻÿåĚşçŽĎæÍąåİŮæŕŤåęĆ
socket åžŞäžď'éŻĘå¿ĹåřŚãĂĆ æL'ĂäžěïijŇä¡ääÿ■èČ¡ä¡£çŤĺ IPv4Address
çŽĎåőđä¿ŃæİěäžčæŻ£äÿĂäÿłåIJřåİĂå■ŮçňęäÿšïijŇä¡äéęŰåĚĹå¿ŮæŸ¿åijŔçŽĎä¡£çŤĺ
str() è¡ňæ■ćåőČãĂĆä¿ŃåęĆïijŽ

```
>>> a = ipaddress.ip_address('127.0.0.1')
>>> from socket import socket, AF_INET, SOCK_STREAM
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s.connect((a, 8080))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Can't convert 'IPv4Address' object to str implicitly
>>> s.connect((str(a), 8080))
>>>
```

æŻť'åď'ŽçŻÿåĚşåĘĚåőźïijŇèŕůåŔĆèĂČ An Introduction to the ipaddress Module

11.5 åĹŻåžžäÿĂäÿłçőĂå■ŢçŽĎRESTæŐěåŔč

éŮőécŸ

ä¡ăæČşä¡£çŤĺäÿĂäÿłçőĂå■ŢçŽĎRESTæŐěåŔčéĂŽè£Ğç¡ŚçżIJè£IJçÍŃæŐğåĹűæĹŰèő£éŮőä¡ăçŽĎåžŤ

èğčåĘşæŰźæąĹ

æđĎåžžäÿĂäÿłRESTéčŐæăijçŽĎæŐěåŔčæIJĂçőĂå■ŢçŽĎæŰźæşŢæŸŕåĹŻåžžäÿĂäÿłå§žäžŐWSGIæă
3333ïijL'çŽĎå¿ĹåřŔçŽĎåžŞïijŇäÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijŽ

```
# resty.py

import cgi

def notfound_404(environ, start_response):
    start_response('404 Not Found', [ ('Content-type', 'text/plain
↪') ])
    return [b'Not Found']

class PathDispatcher:
    def __init__(self):
```

```
        self.pathmap = { }

    def __call__(self, environ, start_response):
        path = environ['PATH_INFO']
        params = cgi.FieldStorage(environ['wsgi.input'],
                                  environ=environ)
        method = environ['REQUEST_METHOD'].lower()
        environ['params'] = { key: params.getvalue(key) for key in␣
↪params }
        handler = self.pathmap.get((method,path), notfound_404)
        return handler(environ, start_response)

    def register(self, method, path, function):
        self.pathmap[method.lower(), path] = function
        return function
```

äÿžäžĘä¡£çŤĺè£ŹäÿłèřČåžęåŹĺïijŇä¡ääŔłéIJĂèęĄçijŰåĘŹäÿ■åŘŇçŽĎåď ĎçŘĘåŹĺïijŇåřśåČŔäÿŃéİćè£Ź

```
import time

_hello_resp = '''\
<html>
  <head>
     <title>Hello {name}</title>
   </head>
   <body>
     <h1>Hello {name}!</h1>
   </body>
</html>'''

def hello_world(environ, start_response):
    start_response('200 OK', [ ('Content-type','text/html')])
    params = environ['params']
    resp = _hello_resp.format(name=params.get('name'))
    yield resp.encode('utf-8')

_localtime_resp = '''\
<?xml version="1.0"?>
<time>
  <year>{t.tm_year}</year>
  <month>{t.tm_mon}</month>
  <day>{t.tm_mday}</day>
  <hour>{t.tm_hour}</hour>
  <minute>{t.tm_min}</minute>
  <second>{t.tm_sec}</second>
</time>'''

def localtime(environ, start_response):
    start_response('200 OK', [ ('Content-type', 'application/xml')␣
↪])
```

```
    resp = _localtime_resp.format(t=time.localtime())
    yield resp.encode('utf-8')

if __name__ == '__main__':
    from resty import PathDispatcher
    from wsgiref.simple_server import make_server

    # Create the dispatcher and register functions
    dispatcher = PathDispatcher()
    dispatcher.register('GET', '/hello', hello_world)
    dispatcher.register('GET', '/localtime', localtime)

    # Launch a basic server
    httpd = make_server('', 8080, dispatcher)
    print('Serving on port 8080...')
    httpd.serve_forever()
```

èęĄæţŃèŕŢäÿŃè£ŹäÿłæIJ■åŁąåŹĺïijŇä¡ăåŔŕäžěä¡£çŤĺäÿĂäÿłæţŔèğĹåŹĺæĹŰ `urllib` åŠŇåőČäžďäžŠãĂĆä¿ŃåęĆïijŽ

```
>>> u = urlopen('http://localhost:8080/hello?name=Guido')
>>> print(u.read().decode('utf-8'))
<html>
  <head>
     <title>Hello Guido</title>
   </head>
   <body>
     <h1>Hello Guido!</h1>
   </body>
</html>

>>> u = urlopen('http://localhost:8080/localtime')
>>> print(u.read().decode('utf-8'))
<?xml version="1.0"?>
<time>
  <year>2012</year>
  <month>11</month>
  <day>24</day>
  <hour>14</hour>
  <minute>49</minute>
  <second>17</second>
</time>
>>>
```

èőĺèőž

åIJÍçijŰåĘŹRESTæŐěåŘčæŮűïijŇéĂŽåÿÿéČ¡æŸŕæIJ■åŁąäžŐæŹőéĂŽçŽĎHTTPèŕůæśĆãĂĆä¡ĘæŸŕè
è£ŹäžŻæŢřæ■őäžěåŘĎçğ■æăĞåĞĘæăijåijŔçijŰçăĄïijŇæŕŤåęĆXMLãĂĄJSONæĹŰCSVãĂĆ
år¡çőąçÍŃåžŔçIJŃäÿŁåŐżå¿ĹçőĂå■ŢïijŇä¡ĘæŸŕäžěè£Źçğ■æŰźåijŔæŘŘä¿ŻçŽĎAPIåŕźäžŐå¿Ĺåď ŽåžŤçŤĺç

ä¿ŃåęĆïijŇéŢ£æIJ§è£Řèą ŇçŽĎçÍŃåžŔåŔŕèČ¡äijŽä¡£çŤÍäÿĂäÿłREST APIæÌěåőđçŐřçŻŚæŐğæĹŰèŕŁæŰ■ãĂĆ åď ğæƢřæ■őåžŤçŤÍçÍŃåžŔåŔŕäžěä¡£çŤÍRESTæÌěæđĎåžžäÿĂäÿłæ RESTè£ŸèČ¡çŤÍæÌěæŐğåĹűçąňäžűèő¿åď ĞæŕŤåęĆæIJžåŹÍäžžãĂĄäijäæĎ§åŹÍãĂĄåůěåŐĆæĹŰçĄŕæşąãĂĆ æŻť éĞ■èęĄçŽĎæŸŕïijŇREST APIåůšçżŔèćńåď ğéĞŔåőćæĹůçńŕçijŰçÍŃçŐŕåćČæĽ ĂæŤŕæŇĄïijŇæŕŤåęĆ-Javascript, Android, iOSç■Ľ ãĂĆ åŻăæ■ď ïijŇåĹľ çŤÍè£Źçğ■æŐěåŘčåŔŕäžèèőľ ä¡ăåijĂåŔŠåĞžæŻť åŁăåď ■æ

äÿžäžĘåőđçŐřäÿĂäÿłçőĂå■ŢçŽĎRESTæŐěåŘčïijŇä¡ăåŔłéIJĂèőľ ä¡ăçŽĎçÍŃåžŔäžčçăĄæżąęűşPythonç WSGIèćńæăĞåĞĘåžŞæŤŕæŇĄïijŇåŘŇæŮűäź§èćńçżÌåď ğéČĺåĹEçňňäÿĽ æŰźwebæąĘæđűæŤŕæŇĄãĂĆ åŻăæ■ď ïijŇåęĆæđIJä¡ăçŽĎäžčçăĄéĄţå¿łè£ŹäÿłæăĞåĞĘïijŇåIJĺåŘŐéÍćçŽĎä¡£çŤÍè£ĞçÍŃäÿ■åřśäijŽæŻť åŁ

åIJÍWSGIäÿ■ïijŇä¡ăåŔŕäžěåČŔäÿŃéÍćè£ŹæăůçžęáőŽçŽĎæŰźåijŔäžěäÿĂäÿłåŔŕèřČçŤÍåŕźèśąå¡ćåijŔæÌ

```
import cgi

def wsgi_app(environ, start_response):
    pass
```

environ åśđæĂğæŸŕäÿĂäÿłå■ŮåĚÿïijŇåŇĚåŘńäžĘäżŐwebæIJ■åŁąåŹÍåęĆA-pache[åŘĆèĂČInternet RFC 3875]æŔŘä¿ŽçŽĎCGIæŐěåŔčäÿ■èŐůåŘŰçŽĎåĂijãĂĆ èęĄåřĘè£ŹäžŻäÿ■åŘŇçŽĎåĂijæŔŘåŘŰåĞžæÌěïijŇä¡ăåŔŕäžěåČŔè£ŹäźĹè£ŹæăůåĘŹïijŽ

```
def wsgi_app(environ, start_response):
    method = environ['REQUEST_METHOD']
    path = environ['PATH_INFO']
    # Parse the query parameters
    params = cgi.FieldStorage(environ['wsgi.input'],␣
↪environ=environ)
```

æĹŚäžňåśŢçd'žäžĘäÿĂäžŻåÿÿèğĄçŽĎåĂijãĂĆenviron['REQUEST_METHOD'] äžčèąÍèŕůæśĆçśžåđŇåęĆGETãĂĄPOSTãĂĄHEADç■Ľ ãĂĆ environ['PATH_INFO'] èąÍçď žèćńèŕůæśĆèţĎæžŘçŽĎèůŕå¿ĎãĂĆ èŕČçŤÍ cgi.FieldStorage() åŘŕäžěäžŐèŕůæśĆäÿ■æŔŘåŘŰæ§ěèŕćåŔĆæŢřåźűåřĘåőČäžňæŤ¿åĚěäÿĂäÿłçśżå■ŮåĚÿåŕźèśąäÿ■äžěä¿£åŘŐ

start_response åŘĆæŢřæŸŕäÿĂäÿłäÿžäžĘåĹiåğŇåŇŰäÿĂäÿłèŕůæśĆåŕźèśąèĂŇå£ĚéążèćńèŕČçŤÍ çňňäÿĂäÿłåŘĆæŢřæŸŕè£ŤåŻđçŽĎHTTPçŁűæĂĄåĂijïijŇçňňäžŇäÿłåŘĆæŢřæŸŕäÿĂäÿł(åŘ■,åĂij)åĚČçżĎ

```
def wsgi_app(environ, start_response):
    pass
    start_response('200 OK', [('Content-type', 'text/plain')])
```

äÿžäžĘè£ŤåŻđæŢřæ■őïijŇäÿĂäÿłWSGIçÍŃåžŔå£ĚéążèŤåŻđäÿĂäÿłå■ŮèŁĆå■ŮçňęäÿšåžŔåĹŮãĂĆåŕ

```
def wsgi_app(environ, start_response):
    pass
    start_response('200 OK', [('Content-type', 'text/plain')])
    resp = []
    resp.append(b'Hello World\n')
    resp.append(b'Goodbye!\n')
    return resp
```

æĹŰèĂĚïijŇä¡ăè£Ÿåŕäžěä¡£çŤÍ yield ïijŽ

```
def wsgi_app(environ, start_response):
    pass
    start_response('200 OK', [('Content-type', 'text/plain')])
    yield b'Hello World\n'
    yield b'Goodbye!\n'
```

è£ŹéĞŇèęĄåijžèřČçŽĎäÿĂçĆźæŸŕæIJĂåŘŐè£ŤåŻđçŽĎå£ĚéążæŸŕå■ŮèŁĆå■Ůçňęäÿšă
å¡ŞçĎűïijŇåźűæšąæIJL'èęĄæśĆä¡ăè£ŤåŽđçŽĎäÿĂåőŽæŸŕæŰĞæIJñïijŇä¡ăåŔŕäžěå¿Ĺè¡żæİ¿çŽĎçijŮåĘŹäÿ

åř¡çőąWSGIçĺŃåžŔéĂŽåÿÿèćńåőŽäźL'æĹŘäÿĂäÿłåĞ¡æŢřïijŇäÿ■è£Ğä¡ăäź§åŔŕäžěä¡£çŤĺçśžåőđä¿ŃæĺÇ
`__call__()` æŰźæşŢãĂĆä¿ŃåęĆïijŽ

```
class WSGIApplication:
    def __init__(self):
        ...
    def __call__(self, environ, start_response)
       ...
```

æĹŚäžňåůšçżŔåIJĺäÿŁéİćä¡£çŤĺè£Źçğ■æŁĂæIJŕåĹŻåžž `PathDispatcher` çśžãĂĆ
è£ŹäÿłåĹęåŔŚåŹĺäżĚäżĚåŔłæŸŕçőąçŘĘäÿĂäÿłå■ŮåĚÿïijŇářĘ(æŰźæşŢ,èůŕå¿Ď)åŕźæŸăářĎåĹřåď'ĎçŘĘåŹĺ
å¡ŞäÿĂäÿłèŕůæśĆåĹřæİěæŮűïijŇåőČçŽĎæŰźæşŢåŠŇèůŕå¿Ďèćńæŕ̊ŘåŔŰåĞžæİěïijŇçĎűåŘŐèćńåĹĘåŔŚåĹ
åŔęåď'ŰïijŇäżżä¡Ţæ§ěèŕćåŔŸéĞŔäijŽèćńèğčæđŘåŘŐæŤ¿åĹřäÿĂäÿłå■ŮåĚÿäÿ■ïijŇäžě
`environ['params']` å¡ćåijŔå■ŸåĆĺãĂĆ åŘŐéİćè£Źäÿłæ■ěéłď'åď'łåÿÿèğĄïijŇæL'Ăäżěåżžèőőä¡ăåIJĺåĹĘ
ä¡£çŤĺåĹĘåŔŚåŹĺçŽĎæŮűåĂŹïijŇä¡ăåŔłéIJĂçőÅå■ŢçŽĎåĹŽåżžäÿĂäÿłåőđä¿ŃïijŇçĎűåŘŐéĂŽè£ĞåőČæş
çijŮåĘŹè£ŹäžŽåĞ¡æŢřåžŤèŕěèűĚçžğçőÅå■ŢäžĘïijŇåŔłèęĄä¡äéĄţå¿ł
`start_response()` åĞ¡æŢřçŽĎçijŮåĘŹèğĎåĹŹïijŇåźűäÿŤæIJĂåŘŐè£ŤåŻđå■ŮèŁĆå■Ůçňęäÿšå■şåŔŕã

å¡ŞçijŮåĘŹè£Źçğ■åĞ¡æŢřçŽĎæŮűåĂŹè£ŸéIJĂæşĺæĎŔçŽĎäÿĂçĆźåřśæŸŕåŕźäžŐå■Ůçňęäÿšæĺąæİ£çŽ
æšąäžžæĎ£æĎŔåĘŹéĆčçğ■åĹřåď'ĎæůůåŘĹçİĂ `print()` åĞ¡æŢř ãĂĄXM-
LåŠŇåď'ğéĞŔæăijåijŔåŇŰæŞ■ä¡IJçŽĎäžčçăĄãĂĆ æĹŚäžňäÿŁéİćä¡£çŤĺäžĘäÿL'åijŢåŔůåŇĚåŘńçŽĎécĎåĚ
è£Źçğ■æŰźåijŔçŽĎåŔŕäžěèől'æĹŚäžňå¿Ĺåőźæ ŸŞçŽĎåIJĺäžěåŘŐä£őæŤźè¿ŞåĞžæăijåijŔ(åŔłéIJĂèęĄä£őæ

æIJĂåŘŐïijŇä¡£çŤĺWSGIè£ŸæIJL'äÿĂäÿłå¿ĹéĞ■èęĄçŽĎéČĺåĹĘåřśæŸŕæšąæIJL'äżĂäźĹåIJřæŰźæŸŕé
åŻääÿžæăĞåĞĘåŕźäžŐæIJ■åŁąåŹĺåŠŇæąĘædűæŸŕäÿ■çńNçŽĎïijŇä¡ăåŔŕäžěåřĘä¡ăçŽĎçĺŃåžŔæŤ¿åĚäżžä
æĹŚäžňä¡£çŤĺäÿŃéİćçŽĎäžčçăĄæţŃèŕŢæţŃèŕŢæIJñèŁĆäžčçăĄïijŽ

```
if __name__ == '__main__':
    from wsgiref.simple_server import make_server

    # Create the dispatcher and register functions
    dispatcher = PathDispatcher()
    pass

    # Launch a basic server
    httpd = make_server('', 8080, dispatcher)
    print('Serving on port 8080...')
    httpd.serve_forever()
```

äÿŁéİćäžčçăĄåĹŻåżžäžĘäÿĂäÿłçőÅå■ŢçŽĎæIJ■åŁąåŹĺïijŇçĎűåŘŐä¡ăåřśåŔŕäžěæİěæţŃèŕŢäÿŃä¡ăçŽĎ
æIJĂåŘŐïijŇå¡Şä¡ăåĞĘåď'Ğè£ŻäÿĂæ■ěæL'l'åśŢä¡ăçŽĎçĺŃåžŔçŽĎæŰźåŹĺïijŇä¡ăåŔŕäžěä£őæŤźè£Źäÿłäżĕ

WSGIæIJňèžńæŸŕäÿĂäÿłå¿ĹåřŔçŽĎæăĞåĞĘãĂĆåŽăæ■ď åőČåźűæšąæIJL'æŔŘä¿ŻäÿĂäžŻéńŸçžğçŽĎ
è£ŹäžŻä¡äèĞłåůšåőđçŐřèţůæİěäź§äÿ■éŽ¿ãĂĆäÿ■è£ĞåęĆæđIJä¡äæČşèęĄæŻt'åď'ŽçŽĎæŤŕæŇĄïijŇåŔŕäžěè
`WebOb` æĹŰèĂĚ `Paste`

11.6 éĂŽè£ĞXML-RPCåőđçŐřçőĂå■ŢçŽĎè£IJçÍŃèřČçŤÍ

éŮőécŸ

ä¡äæČşæL'¿åĹřäÿĂäÿłçőĂå■ŢçŽĎæŰźåijŔåŐżæL'ğèąŇè£ŘèąŇåIJÍè£IJçÍŃæIJžåŹÍäÿŁéİççŽĎPythonçÍ

èğčåĘşæŰźæąĹ

åőđçŐřäÿĂäÿłè£IJçÍŃæŰźæşŢèřČçŤÍçŽĎæIJĂçőĂå■ŢæŰźåijŔæŸŕä¡£çŤÍXML-RPCãĂĆäÿŃéİćæĹŚäžňæijŤçď'žäÿĂäÿŃäÿĂäÿłåőđçŐřäžĘéŤő-åĂijå■ŸåĆÍåŁ§èČ¡çŽĎçőĂå■ŢæIJ■åŁąåŹĺijŽ

```
from xmlrpc.server import SimpleXMLRPCServer

class KeyValueServer:
    _rpc_methods_ = ['get', 'set', 'delete', 'exists', 'keys']
    def __init__(self, address):
        self._data = {}
        self._serv = SimpleXMLRPCServer(address, allow_none=True)
        for name in self._rpc_methods_:
            self._serv.register_function(getattr(self, name))

    def get(self, name):
        return self._data[name]

    def set(self, name, value):
        self._data[name] = value

    def delete(self, name):
        del self._data[name]

    def exists(self, name):
        return name in self._data

    def keys(self):
        return list(self._data)

    def serve_forever(self):
        self._serv.serve_forever()

# Example
if __name__ == '__main__':
    kvserv = KeyValueServer(('', 15000))
    kvserv.serve_forever()
```

äÿŃéÌćæĹŚäżňäżŐäÿĂäÿłåőćæĹůçńŕæIJžåŹĺäÿŁéÌćæÌěèő£éŮőæIJ■åŁąåŹĺïijŽ

```
>>> from xmlrpc.client import ServerProxy
>>> s = ServerProxy('http://localhost:15000', allow_none=True)
>>> s.set('foo', 'bar')
>>> s.set('spam', [1, 2, 3])
>>> s.keys()
['spam', 'foo']
>>> s.get('foo')
'bar'
>>> s.get('spam')
[1, 2, 3]
>>> s.delete('spam')
>>> s.exists('spam')
False
>>>
```

èőÍèőž

XML-RPC åŘŕäżěèőľ æĹŚäżňå¿Ĺåőźæ ŸŞçŽĎæđĎéĂääÿĂäÿłçőĂå■ŢçŽĎè£IJçÍŃèřČçŤĺæIJ■åŁąãĂĆä¡éĂŽè£ĞåőČçŽĎæŰźæşŢ `register_function()` æÌěæşĺåĘŇåĞ¡æŢřïijŇçĎűåŘŐä¡£çŤĺæŰźæşŢ `serve_forever()` åŘŕåŁĺåőČãĂĆ åIJÍäÿŁéÌćæĹŚäżňårĘè£ŹäžŽæ■ěéłď æŤ¿åIJÍäÿĂèţůåĘŹåĹřäÿĂäÿłçśż

```
from xmlrpc.server import SimpleXMLRPCServer
def add(x,y):
    return x+y

serv = SimpleXMLRPCServer(('', 15000))
serv.register_function(add)
serv.serve_forever()
```

XML-RPCæŽťéIJšåĞžæÌěçŽĎåĞ¡æŢřåŘłèČ¡éĂĆçŤĺäžŐéČĺåĹĘæŢřæ■őçśżåđŃïijŇærŤåęĆå■Ůçňęäÿš åŕźäžŐåĚűäžŰçśżåđŃåřšå¿ŮéIJĂèęĄåĄŽäžŽécÌåď ŰçŽĎåŁ§èŕ¿äžĘãĂĆ ä¿ŃåęĆïijŇåęĆæđIJä¡äæČşéĂŽè£Ğ XML-RPC äijäéĂŠäÿĂäÿłåŕźèšąåőđä¿ŃïijŇåőđéŹĚäÿŁåŘłæIJL'äžŰçŽĎ

```
>>> class Point:
...     def __init__(self, x, y):
...             self.x = x
...             self.y = y
...
>>> p = Point(2, 3)
>>> s.set('foo', p)
>>> s.get('foo')
{'x': 2, 'y': 3}
>>>
```

çśżäijijçŽĎïijŇåŕźäžŐäžŇè£ŻåĹűæŢřæ■őçŽĎåď ĎçŘĘäź§èů§ä¡äæČşèšąçŽĎäÿ■åď łäÿĂæăůïijŽ

```
>>> s.set('foo', b'Hello World')
>>> s.get('foo')
```

```
<xmlrpc.client.Binary object at 0x10131d410>

>>> _.data
b'Hello World'
>>>
```

äÿĂèĹňæÌěèőšïijŇä¡ăäÿ■åžŤèŕěářĘ XML-RPC æIJ■åŁąäžěåĚňåĚśAPIçŽĎæŰźåijŔæŽť'éIJšåĞžæÌěãĂĆ åŕźäžŐè£Źçğ■æČĚåĘţïijŇéĂŽåÿÿåĹĘåÿČåijŔåžŤçŤÍçÍŇåžŔäijŽæŸŕäÿĂäÿłæŻť'åě¡çŽĎéĂL'æŇľãĂĆ

XML-RPCçŽĎäÿĂäÿłçijžçĆźæŸŕåőČçŽĎæĂğèČ¡ãĂĆSimpleXMLRPCServer çŽĎåőđçŐřæŸŕå■Ţçž£çÍŇçŽĎïijŇ æL'Ăäžěåőčäÿ■éĂĆåŘĹäžŐåď'ğåđŃçÍŇåžŔïijŇår¡çőąæĹŚäžňåIJÍ11.2åřł åŘęåď'ŰïijŇçŤšäžŐ XML-RPC åřĘæL'ĂæIJL'æŢřæ■őéČ¡åžŔåĹŮåŇŰäÿžXMLæäijåijŔïijŇæL'ĂäžěåőčäijŽ ä¡ĘæŸŕåőČäź§æIJL'äijŸçĆźïijŇè£ŹçğæŰźåijŔçŽĎçijŰçĂąåŘŕäžěèćńçżÍåď'ğéČĺåĹĘåĚűäžŰçijŰçÍŇèŕ■èÍĂ éĂŽè£Ğä¡£çŤÍè£Źçğ■æŰźåijŔïijŇåĚűäžŰèŕ■èÍĂçŽĎåőćæĹůçńŕçÍŇåžŔéČ¡èČ¡èő£éŮőä¡ăçŽĎæIJ■åŁąãĂĆ

èŹ¡çĎűXML-RPCæIJL'å¿Ĺåď'ŽçijžçĆźïijŇä¡ĘæŸŕåęĆæđIJä¡ăéIJĂèęĄå£néĂ§æđĎåžžäÿĂäÿłçőĂå■Ţè£ æIJL'æŮűåĂŹïijŇçőĂå■ŢçŽĎæŰźæąĹåřśåůšçżŔèűşåď'§äžĘãĂĆ

11.7 âIJÍäÿ■åŘŇçŽĎPythonèğčéĞŁåŹÍäźŃéŮť'äžď'äžŠ

éŮőécŸ

ä¡ăåIJÍäÿ■åŘŇçŽĎæIJžåŹÍäÿŁéİćè£ŘèąŇçİĂåď'ŽäÿłpythonèğčéĞŁåŹÍåőđä¿ŃïijŇåźűåÿŇæIJŻèČ¡åď'§å

èğčåĘşæŰźæąĹ

éĂŽè£Ğä¡£çŤÍ multiprocessing.connection æÍąåİŮåŔŕäžěå¿ĹåőźæŸŞçŽĎåőđçŐřèğčéĞŁåŹÍ äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎåžŤç■ŤæIJ■åŁąåŹÍä¿Ńå■ŘïijŽ

```
from multiprocessing.connection import Listener
import traceback

def echo_client(conn):
    try:
        while True:
            msg = conn.recv()
            conn.send(msg)
    except EOFError:
        print('Connection closed')

def echo_server(address, authkey):
    serv = Listener(address, authkey=authkey)
    while True:
        try:
            client = serv.accept()

            echo_client(client)
        except Exception:
```

```
            traceback.print_exc()

echo_server(('', 25000), authkey=b'peekaboo')
```

çĎűåŘŐåőćæĹůçńŕè£đæŐěæIJ■åŁąåŹĺåźűåŔŠéĂĄæűĹæĄŕçŽĎçőĂå■Ţçđ'žä¿ŃïijŽ

```
>>> from multiprocessing.connection import Client
>>> c = Client(('localhost', 25000), authkey=b'peekaboo')
>>> c.send('hello')
>>> c.recv()
'hello'
>>> c.send(42)
>>> c.recv()
42
>>> c.send([1, 2, 3, 4, 5])
>>> c.recv()
[1, 2, 3, 4, 5]
>>>
```

èů§åžŢåśĆsocketäÿ■åŘŇçŽĎæŸŕïijŇæŕŔäÿłæűĹæĄŕäijŽåőŇæŢt'ä£Ìå■ŸïijĹæŕŔäÿĂäÿłéĂŽè£ĞsendO
åŔęåđ'ŰïijŇæĽĂæIJL'åŕźèśąäijŽéĂŽè£ĞpickleåžŔåĹŮåŇŰãĂĆåŻăæ■đ'ïijŇäżżä¡ŢåĚijåőźpickleçŽĎåŕźèśą

èőĺèőž

çŻőåL'■æIJL'å¿Ĺåđ'ŽçŤĺæÌěåőđçŐřåŘĎçğ■æűĹæĄŕäijăè¿ŞçŽĎåŇĚåŠŇåĞ¡æŢřåžŞïijŇæŕŤåęĆZeroMQ
ä¡ăè£ŸæIJL'åŔęåđ'ŰäÿĂçğ■éĂĽ'æŃl'åŕśæŸŕèĞłåůśåIJĺåžŢåśĆsocketå§žçąĂäźŇäÿŁæÌěåőđçŐřäÿĂäÿłæűĹæ
ä¡ĘæŸŕä¡ăæČşèęĄçőĂå■ŢäÿĂçĆźçŽĎæŰźæąĹïijŇéĆčäźĹè£ŹæŮűåĂŹ
multiprocessing.connection åŕśæť¿äÿŁçŤĺåIJžäžĘãĂĆ
äżĚäżĚä¡£çŤĺäÿĂäžŽçőĂå■ŢçŽĎèŕ■åŔěå■şåŔŕåőđçŐřåđ'Žäÿłèğčéğ̈ŁåŹĺäźŇéŮť'çŽĎæűĹæĄŕéĂŽä£ąãĂĆ

åęĆæđIJä¡ăçŽĎèğčéğ̈ŁåŹĺè£ŘèąŇåIJĺåŘŇäÿĂåŘŕæIJžåŹĺäÿŁéÌćïijŇéĆčäźĹä¡ăåŔŕäżěä¡£çŤĺåŔęåđ'ŰçŽ
èęĄæČşä¡£çŤĺUNIXå§§åěŮæŐěå■ŮæÌěåĹŻåžžäÿĂäÿłè£đæŐěïijŇåŘłéIJĂçőĂå■ŢçŽĎåřĘåIJřåİĂæŤźåĘźäÿ

```
s = Listener('/tmp/myconn', authkey=b'peekaboo')
```

èęĄæČşä¡£çŤĺWindowsåŚ¡åŘ■çőąéĄŞæÌěåĹŻåžžè£đæŐěïijŇåŘłéIJĂåČŔäÿŇéÍćè£ŹæăůäĄ£çŤĺäÿĂäÿła

```
s = Listener(r'\\.\pipe\myconn', authkey=b'peekaboo')
```

äÿĂäÿłéĂŽçŤĺåĞĘåĹŹæŸŕïijŇä¡ääÿ■èęĄä¡£çŤĺ multiprocessing
æÌěåőđçŐřäÿĂäÿłåŕźåđ'ŰçŽĎåĚňåĚśæIJ■åŁąãĂĆ Client() åŠŇ Listener()
äÿ■çŽĎ authkey åŔĆæŢřçŤĺæÌěèőđ'èŕĄåŔŚètůè£đæŐěçŽĎçżĹçńŕçŤĺæĹůãĂĆ
åęĆæđIJåŕĘéŠěäÿ■åŕźäijŽäžğçŤ§äÿĂäÿłåijĆåÿÿãĂĆæ■đ'åđ'ŰïijŇèŕěæĺąåİŮæIJĂéĂĆåŘĹçŤĺæÌěåžžçńŇéŢ£
ä¿ŇåęĆïijŇäÿđ'äÿłèğčéĞŁåŹĺäźŇéŮť'åŘŕåŁĺåŘŐåŕšäijĂåğŇåžžçńŇè£đæŐěåźűåIJĺåđ'ĎçŘĘæ§ŘäÿłéŭőécŸč

åęĆæđIJä¡ăéIJĂèęĄåŕźåžŢåśĆè£đæŐěåĄŽæŻť'åđ'ŽçŽĎæŐğåĹűïijŇæŕŤåęĆéIJĂèęĄæŤŕæŇĄèŭĚæŮűã
ä¡ăæIJĂåě¡ä¡£çŤĺåŘęåđ'ŰçŽĎåžŞæĹŰèĂĚæŸŕåIJléńŸåśĆsocketäÿŁæÌěåőđçŐřè£ŹäžŽçL'źæĂğãĂĆ

11.8 åőđçŐřè£lJçÍŃæŰźæşŢèřČçŤÍ

éŮőécŸ

ä¡ăæČşåIJÍäÿĂäÿłæúĹæĄŕäijăè¿ŞåśĆåęĆ sockets ãĂĄmultiprocessing connections æĹŰ ZeroMQ çŽĎå§žçąĂäźŃäÿŁåőđçŐřäÿĂäÿłçőĂå■ŢçŽĎè£IJçÍŃè£ĞçÍŃèřČçŤÍïijĹRPC

èğčåĘşæŰźæąĹ

åřĘåĞ¡æŢřèřůæśĆãĂĄåŔĆæŢřåŠŇè£ŤåŻđåĂijä¡£çŤÍpickleçijŰçăĄåŘŐïijŇåIJÍäÿ■åŘŇçŽĎèğčéĞŁåŹäÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎPRCåď ĎçŘĘåŹÍïijŇåŔŕäżěèćńæŢť åŘĹåĹřäÿĂäÿłæIJ■åŁąåŹÍäÿ■åŐžïijŽ

```
# rpcserver.py

import pickle
class RPCHandler:
    def __init__(self):
        self._functions = { }

    def register_function(self, func):
        self._functions[func.__name__] = func

    def handle_connection(self, connection):
        try:
            while True:
                # Receive a message
                func_name, args, kwargs = pickle.loads(connection.
↪recv())
                # Run the RPC and send a response
                try:
                    r = self._functions[func_name](*args,**kwargs)
                    connection.send(pickle.dumps(r))
                except Exception as e:
                    connection.send(pickle.dumps(e))
        except EOFError:
            pass
```

èęĄä¡£çŤÍè£Źäÿłåď ĎçŘĘåŹÍïijŇä¡ăéIJĂèęĄåřĘåőČåŁăăĚěåĹřäÿĂäÿłæűĹæĄŕæIJ■åŁąåŹÍäÿ■ãĆĆä¡ăæä¡ĘæŸŕä¡£çŤÍ multiprocessing åžŞæŸŕæIJĂçőĂå■ŢçŽĎãĆĆäÿŃéİćæŸŕäÿĂäÿłR-PCæIJ■åŁąåŹÍä¿Ńå■ŘïijŽ

```
from multiprocessing.connection import Listener
from threading import Thread

def rpc_server(handler, address, authkey):
    sock = Listener(address, authkey=authkey)
    while True:
        client = sock.accept()
        t = Thread(target=handler.handle_connection, args=(client,))
```

```
        t.daemon = True
        t.start()

# Some remote functions
def add(x, y):
    return x + y

def sub(x, y):
    return x - y

# Register with a handler
handler = RPCHandler()
handler.register_function(add)
handler.register_function(sub)

# Run the server
rpc_server(handler, ('localhost', 17000), authkey=b'peekaboo')
```

äÿžäžĘäżŐäÿĂäÿłè£IJçÍŃåőćæĹůçńŕèő£éŮőæIJ■åŁąåŹĺïijŇä¡äéIJĂèęĄåĹŻåżžäÿĂäÿłåŕźåžŤçŽĎçŤÍæİ

```
import pickle

class RPCProxy:
    def __init__(self, connection):
        self._connection = connection
    def __getattr__(self, name):
        def do_rpc(*args, **kwargs):
            self._connection.send(pickle.dumps((name, args,␣
↪kwargs)))
            result = pickle.loads(self._connection.recv())
            if isinstance(result, Exception):
                raise result
            return result
        return do_rpc
```

èęĄä¡£çŤÍè£ŹäÿłäżčçŘĘçśżïijŇä¡äéIJĂèęĄåřĘåĚűåŇĚèčĚåĹřäÿĂäÿłæIJ■åŁąåŹÍçŽĎè£đæŐěäÿŁéİćïijŇ

```
>>> from multiprocessing.connection import Client
>>> c = Client(('localhost', 17000), authkey=b'peekaboo')
>>> proxy = RPCProxy(c)
>>> proxy.add(2, 3)

5
>>> proxy.sub(2, 3)
-1
>>> proxy.sub([1, 2], 4)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "rpcserver.py", line 37, in do_rpc
    raise result
TypeError: unsupported operand type(s) for -: 'list' and 'int'
```

```
>>>
```

èęĄæşĺæĎŘçŽĎæŸŕå¿Ĺåď'ŽæűĹæĄŕåśĆïijĹæŕŤåęĆ multiprocessing ïijĽåůšçżŔä¡£çŤĺpickleåžŔåĹŮåŇŰäžĘæŢřæ■őãĂĆ åęĆæđIJæŸŕè£Źæăůç˝ĎèŕİïijŇåŕź pickle.dumps() åŠŇ pickle.loads() çŽĎèřČçŤĺèęĄåŐżæŐĽ˜ãĂĆ

èőĺèőž

RPCHandler åŠŇ RPCProxy çŽĎå§žæIJňæĂİèůŕæŸŕå¿ĹæŕŤè¿ČçőĂå■ŢçŽĎãĂĆ åęĆæđIJäÿĂäÿłåőćæĹůçńŕæČşèęĄèřČçŤĺäÿĂäÿłè£IJçĺŇåĞ¡æŢřïijŇæŕŤåęĆ foo(1, 2, z=3) ,äžčçŘĘçśżåĹŻåžžäÿĂäÿłåŇĚåŘńäžĘąåĞ¡æŢřåŘ■åŠŇåŔĆæŢřçŽĎåĚČçżĎ ('foo', (1, 2), {'z': 3}) ãĂĆ è£ŹäÿłåĚČçżĎèćńpickleåžŔåĹŮåŇŰåŘŐéĂŽè£Ğç¡ŚçżIJè£đæŐěåŔŚçŤ§åČ è£ŹäÿĂæ■ěåIJĺ RPCProxy çŽĎ __getattr__() æŰźæşŢè£ŤåŽđçŽĎ do_rpc() éŮ■åŇĚäÿ■åőŇæĹŘãĂĆ æIJ■åŁąåŹĺæŐěæŤűåŘŐéĂŽè£ĞpickleåŔ■åžŔåĹŮåŇŰæűĹæĄŕïijŇæ§ěæL'¿åĞ¡æ æL'ğèąŇçżŞæđIJ(æĹŰåijĆåÿÿ)èćńpickleåžŔåĹŮåŇŰåŘŐè£ŤåŽđåŔŚéĂĄçžŹåőćæĹůçńŕãĂĆæĹŚäžňçŽĎåő multiprocessing è£ŻèąŇéĂŽä£ąãĂĆ äÿ■è£ĞïijŇè£ŹçğæŰźåijŔåŕäžěéĂĆçŤĺäžŐåĚűäžŰäžžä¡ŢæűĹ äžĚäžĚåŔłéIJĂèęĄářĘè£đæŐěåŕźèśąæ■ćæĹŘåŘĹéĂĆçŽĎZeroMQçŽĎsocketåŕźèśąå■şåŔŕãĂĆ

çŤśäžŐåžŢåśĆéIJĂèęĄä¿ÌètŰpickleïijŇéĆčäźĹåőL'åĚĺéŰőécŸåřśéIJĂèęĄèĂČèŹŚäžĘ ïijĹåŻääÿžäÿĂäÿłèĄłæŸŐçŽĎéżŚåőćåŔŕäžěåĹŻåžžçL'źåőŽçŽĎæűĹæĄŕïijŇèČ¡åď§èől'äžžæĎŔåĞ¡æŢřéĂŽ åŽăæ■ď'ä¡ăæřÿè£IJäÿ■èęĄåĚĄèőÿæÌěèĞłäÿ■ä£ąäžžæĹŰæIJłèőd'èŕĄçŽĎåőćæĹůçńŕçŽĎRPCãĂĆçL'źåĹńæ è£Źçğ■åŔłèČ¡åIJĺåĘĚéČĺèćńä¡£çŤĺïijŇä¡■äžŐéŸšçĄńáćŹåŘŐéİćåźűäÿŤäÿ■èęĄåŕźåď'ŰæŽť éIJšãĂĆ

ä¡IJäÿžpickleçŽĎæŻ£äžčïijŇä¡ăäź§èőÿåŔŕäžěèĂČèŹŚä¡£çŤĺJSONãĂĄXMLæĹŰäÿĂäžŻåĚűäžŰçŽĎçi ä¿ŇåęĆïijŇæIJňæIJžåőđä¿ŇåŔŕäžěå¿Ĺåőźæ˜ŞçŽĎæŤźåĘŹæĹŘJSONçijŰçăĄæŰźæąĹãĂĆè£ŸéIJĂèęĄářĘ pickle.loads() åŠŇ pickle.dumps() æŻ£æ■ćæĹŘ json.loads() åŠŇ json.dumps() å■şåŔŕïijŽ

```
# jsonrpcserver.py
import json

class RPCHandler:
    def __init__(self):
        self._functions = { }

    def register_function(self, func):
        self._functions[func.__name__] = func

    def handle_connection(self, connection):
        try:
            while True:
                # Receive a message
                func_name, args, kwargs = json.loads(connection.
↪recv())
                # Run the RPC and send a response
                try:
                    r = self._functions[func_name](*args,**kwargs)
                    connection.send(json.dumps(r))
                except Exception as e:
```

```
                    connection.send(json.dumps(str(e)))
            except EOFError:
                 pass

# jsonrpcclient.py
import json

class RPCProxy:
    def __init__(self, connection):
        self._connection = connection
    def __getattr__(self, name):
        def do_rpc(*args, **kwargs):
            self._connection.send(json.dumps((name, args, kwargs)))
            result = json.loads(self._connection.recv())
            return result
        return do_rpc
```

åőđçŐřRPCçŽĎäÿĂäÿłæŕŤè¿Čåď■æİĆçŽĎéŮőéćŸæŸŕåęĆä¡ŢåŐžåď ĎçŘĘåijĆåÿÿãĂĆèĞşåřŚïijŇå¡Şå
åŽăæ■ď ïijŇè£ŤåŻđçżŹåőćæĹůçńŕçŽĎåijĆåÿÿæĽĂäžčèąĺçŽĎåŘńäźĽåřśèęĄåě¡åě¡èő¿èőąäžĘãĂĆ
åęĆæđIJä¡ää¡£çŤĺpickleïijŇåijĆåÿÿåŕźèśąåőđä¿ŃåIJĺåőćæĹůçńŕèČ¡èćńåŔ■åžŔåĹŮåŇŰåźűæŁŻåĞžãĂĆåęĆ
äÿ■è£ĞèĞşåřŚïijŇä¡ääžŤèŕěåIJĺåŞ■åžŤäÿ■è£ŤåŻđåijĆåÿÿå■Ůçňęäÿšã ĂĆæĹŚäžňåIJĺJSONçŽĎä¿Ńå■Řäÿ■å

åŕźäžŐåĚűäżŰçŽĎRPCåőđçŐřä¿Ńå■ŘïijŇæĹŚæŐĺè■Řä¡ăçIJŃçIJŃåIJĺXML-RPCäÿ■ä¡£çŤĺçŽĎ `SimpleXMLRPCServer` åŠŇ `ServerProxy` çŽĎåőđçŐřïijŇ äźş§åřśæŸŕ11.6åřŔèŁĆäÿ■çŽĎåĘĚåőźãĂĆ

11.9 çőĂå■ŢçŽĎåőćæĹůçńŕèöď èŕĄ

éŮőécŸ

ä¡äæČşåIJĺåĹĘåÿČåijŔçşżçž§äÿ■åőđçŐřäÿĂäÿłçőĂå■ŢçŽĎåőćæĹůçńŕè£đæŐěèöď èŕĄåŁ§èČ¡ïijŇåŔĹä

èğčåĘşæŰźæąĹ

åŔŕäžěåĹl'çŤĺ `hmac` æĺąåİŮåőđçŐřäÿĂäÿłè£đæŐěæŔąæĽ'ŃïijŇäžŐèĂŇåőđçŐřäÿĂäÿłçőĂå■ŢèĂŇéńŸ

```
import hmac
import os

def client_authenticate(connection, secret_key):
    '''
    Authenticate client to a remote service.
    connection represents a network connection.
    secret_key is a key known only to both client/server.
    '''
    message = connection.recv(32)
    hash = hmac.new(secret_key, message)
    digest = hash.digest()
```

```
    connection.send(digest)

def server_authenticate(connection, secret_key):
    '''
    Request client authentication.
    '''
    message = os.urandom(32)
    connection.send(message)
    hash = hmac.new(secret_key, message)
    digest = hash.digest()
    response = connection.recv(len(digest))
    return hmac.compare_digest(digest,response)
```

å§žæIJňåŐ§çŘĘæŸŕå¡Şè£đæŐěåžžçńŃåŘŐïijŇæIJ■åŁąåŹĺçżŹåőćæĹůçńŕåŔŠéĂĄäÿĂäÿłéŽŔæIJžçŽĎ os.urandom() è£ŤåŻđåĂijïijL'ãĂĆ åőćæĹůçńŕåŠŇæIJ■åŁąåŹĺåŘŇæŮűåĹl'çŤĺh-macåŠŇäÿĂäÿłåŔłæIJL'åŔŇæŰźç§ěéĄŞçŽĎåŕĘéŠěæİèèőąçőŮåĞžäÿĂäÿłåŁăåŕĘåŞĹåÿŇåĂijãĂĆçĎűåŘŐåæIJ■åŁąåŹĺéĂŽè£ĞæŕŤè¿Čè£ŹäÿłåĂijåŠŇèĞłåůśèőąçőŮçŽĎæŸŕåŘęäÿĂèĞt'æİěåĘşåőŽæŐěåŔŮæĹŰæŃŠ hmac.compare_digest() åĞ¡æŢřãĂĆ ä¡£çŤĺè£ŹäÿłåĞ¡æŢřåŔŕäžěéĄ£åĚ■éĄ■åĹřæŮűéŮt'åĹĘæđŘæŤäÿžäžĘä¡£çŤĺè£ŹäžŽåĞ¡æŢřïijŇä¡ăéIJĂèęĄåřĘåőČéŻĘæĹŘåĹřåůšæIJL'çŽĎç¡ŚçżIJæĹŰæűĹæĄŕäžčçăĄäÿ■

```
from socket import socket, AF_INET, SOCK_STREAM

secret_key = b'peekaboo'
def echo_handler(client_sock):
    if not server_authenticate(client_sock, secret_key):
        client_sock.close()
        return
    while True:

        msg = client_sock.recv(8192)
        if not msg:
            break
        client_sock.sendall(msg)

def echo_server(address):
    s = socket(AF_INET, SOCK_STREAM)
    s.bind(address)
    s.listen(5)
    while True:
        c,a = s.accept()
        echo_handler(c)

echo_server(('', 18000))

Within a client, you would do this:

from socket import socket, AF_INET, SOCK_STREAM

secret_key = b'peekaboo'
```

```
s = socket(AF_INET, SOCK_STREAM)
s.connect(('localhost', 18000))
client_authenticate(s, secret_key)
s.send(b'Hello World')
resp = s.recv(1024)
```

èőÍèőž

hmac èőď'èŕĄçŽĎäÿĂäÿłåÿÿèğĄä¡£çŤĺåIJžæŹŕæŸŕåĘĚéČĺæűĹæĄŕéĂŽä£ąçşżçż§åŠŇè£ŻçÍŇéŮť'éĂŽ
ä¿ŇåęĆïijŇåęĆæđIJä¡ăçijŰåĘŹçŽĎçşżçż§æűĽ'åŔŁåĹřäÿĂäÿłéŻĘç¿d'äÿ■åď'Žäÿłåď'ĎçŘĘåŹÍäźŇéŮť'çŽĎéĂ
ä¡ăåŔŕäžĕä¡£çŤĺæIJňèŁĆæŰźæąĹæĺĕçąőä£ĺåŔłæIJL'èćńåĚĄèőÿçŽĎè£ŻçÍŇäźŇéŮť'æL'■èČ¡å¡ijæ■d'éĂŽä£ą
äžŇåőđäÿŁïijŇå§žäžŐ hmac çŽĎèőď'èŕĄèćń multiprocessing
æĺąåİŮä¡£çŤĺæİĕåőđçŐřå■Řè£ŻçÍŇçŻť'æŐĕçŽĎéĂŽä£ąãĂĆ

è£ŸæIJL'äÿĂçĆźéIJĂèęĄåijžèřČçŽĎæŸŕè£đæŐĕèőď'èŕĄåŠŇåŁăărĘæŸŕäÿd'çăĄäžŇãĂĆ
èőď'èŕĄæĹŘåŁ§äźŇåŘŐçŽĎéĂŽä£ąæűĹæĄŕæŸŕäżĕæŸŐæŰĞå¡ćåijŔåŔŚéĂĄçŽĎïijŇäżżä¡Ţäžžåŕłèę Ąæč

hmacèőď'èŕĄçőŮæşŢå§žäžŐåŞĹåÿŇåĞ¡æŢřåęĆMD5åŠŇSHA-
1ïijŇåĚşäžŐè£ŹäÿłåIJĺIETF RFC 2104äÿ■æIJL'èŕęçżĘäżŃçż■ãĂĆ

11.10 åIJĺç¡ŚçżIJæIJ■åŁąäÿ■åŁăåĚěSSL

éŮőécŸ

ä¡ăæČşåőđçŐřäÿĂäÿłå§žäžŐsocketsçŽĎç¡ŚçżIJæIJ■åŁąïijŇåőćæĹůçńŕåŠŇæIJ■åŁąåŹĺéĂŽè£ĞSSLå■Ŕ

èğčåĘşæŰźæąĹ

ssl æĺąåİŮèČ¡äÿžåžŢåśĆsocketè£đæŐĕæůżåŁăSSLçŽĎæŤŕæŇĄãĂĆ ssl.
wrap_socket() åĞ¡æŢřæŐĕåŔŮäÿĂäÿłåůšå■ŸåIJĺçŽĎsocketä¡IJäÿžåŔĆæŢřåźűä¡£çŤĺSSLåśĆæİĕåŇĚèč
ä¿ŇåęĆïijŇäÿŇéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎåžŤç■ŤæIJ■åŁąåŹĺïijŇèČ¡åIJĺæIJ■åŁąåŹĺçńŕäÿžæL'ĂæIJL'åőćæĹ

```
from socket import socket, AF_INET, SOCK_STREAM
import ssl

KEYFILE = 'server_key.pem'    # Private key of the server
CERTFILE = 'server_cert.pem' # Server certificate (given to client)

def echo_client(s):
    while True:
        data = s.recv(8192)
        if data == b'':
            break
        s.send(data)
    s.close()
    print('Connection closed')
```

```
def echo_server(address):
    s = socket(AF_INET, SOCK_STREAM)
    s.bind(address)
    s.listen(1)

    # Wrap with an SSL layer requiring client certs
    s_ssl = ssl.wrap_socket(s,
                            keyfile=KEYFILE,
                            certfile=CERTFILE,
                            server_side=True
                            )
    # Wait for connections
    while True:
        try:
            c,a = s_ssl.accept()
            print('Got connection', c, a)
            echo_client(c)
        except Exception as e:
            print('{}: {}'.format(e.__class__.__name__, e))

echo_server(('', 20000))
```

äÿŃéÌćæĹŚäżňæijŤçd'žäÿĂäÿłåőćæĹůçńŕè£đæŐěæIJ■åŁąåŹĺçŽĎäžd'äžŠä¿Ńå■ŘãĂĆåőćæĹůçńŕäijŽèŕü

```
>>> from socket import socket, AF_INET, SOCK_STREAM
>>> import ssl
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s_ssl = ssl.wrap_socket(s,
                cert_reqs=ssl.CERT_REQUIRED,
                ca_certs = 'server_cert.pem')
>>> s_ssl.connect(('localhost', 20000))
>>> s_ssl.send(b'Hello World?')
12
>>> s_ssl.recv(8192)
b'Hello World?'
>>>
```

è£Źçğ■çŻt'æŐěåd'ĎçŘĘåžŢåśĆsocketæŰźåijŔæIJL'äÿłéŮőécŸåřśæŸŕåőČäÿ■èČ¡å¿Ĺåě¡çŽĎèů§æăĞåČ
ä¿ŃåęĆïijŇçżÌåd'ğéČĺåĹĘæIJ■åŁąåŹĺäžčçăĄïijĹHTTPãĂĄXML-
RPCç■L'ïijL'åőđéŹĚäÿŁæŸŕå§žäžŐ `socketserver` åžŞçŽĎãĂĆ
åőćæĹůçńŕäžčçăĄåIJĺäÿĂäÿłè¿ČéńŸåśĆäÿŁåőđçŐřãĂĆæĹŚäżňéIJĂèęĄåŔęåd'ŰäÿĂçğ■çĺ■å¿őäÿ■åŘŇçŽĎ

éęŰåĚĹïijŇårźäžŐæIJ■åŁąåŹĺèĂŇèĺĂïijŇåŔŕäžěéĂŽè£ĞåČŔäÿŃéÌćè£Źæåůä¡£çŤĺäÿĂäÿłmixinçśžæÌě

```
import ssl

class SSLMixin:
'''
Mixin class that adds support for SSL to existing servers based
on the socketserver module.
```

```
'''
def __init__(self, *args,
             keyfile=None, certfile=None, ca_certs=None,
             cert_reqs=ssl.CERT_NONE,
             **kwargs):
    self._keyfile = keyfile
    self._certfile = certfile
    self._ca_certs = ca_certs
    self._cert_reqs = cert_reqs
    super().__init__(*args, **kwargs)

def get_request(self):
    client, addr = super().get_request()
    client_ssl = ssl.wrap_socket(client,
                                 keyfile = self._keyfile,
                                 certfile = self._certfile,
                                 ca_certs = self._ca_certs,
                                 cert_reqs = self._cert_reqs,
                                 server_side = True)
    return client_ssl, addr
```

äÿžäžĘä¡£çŤĺè£ŹäÿłmixinçśżïijŇä¡ăåŔŕäżěåřĘåőČèů§åĚűäżŰæIJ■åŁąåŹĺçśżæůůåŘĹãĂĆä¿ŃåęĆïijŇäÿ RPCæIJ■åŁąåŹĺä¿Ńå■ŘïijŽ

```
# XML-RPC server with SSL

from xmlrpc.server import SimpleXMLRPCServer

class SSLSimpleXMLRPCServer(SSLMixin, SimpleXMLRPCServer):
    pass

Here's the XML-RPC server from Recipe 11.6 modified only slightly␣
↪to use SSL:

import ssl
from xmlrpc.server import SimpleXMLRPCServer
from sslmixin import SSLMixin

class SSLSimpleXMLRPCServer(SSLMixin, SimpleXMLRPCServer):
    pass

class KeyValueServer:
    _rpc_methods_ = ['get', 'set', 'delete', 'exists', 'keys']
    def __init__(self, *args, **kwargs):
        self._data = {}
        self._serv = SSLSimpleXMLRPCServer(*args, allow_none=True,␣
↪**kwargs)
        for name in self._rpc_methods_:
            self._serv.register_function(getattr(self, name))
```

```
    def get(self, name):
        return self._data[name]

    def set(self, name, value):
        self._data[name] = value

    def delete(self, name):
        del self._data[name]

    def exists(self, name):
        return name in self._data

    def keys(self):
        return list(self._data)

    def serve_forever(self):
        self._serv.serve_forever()

if __name__ == '__main__':
    KEYFILE='server_key.pem'     # Private key of the server
    CERTFILE='server_cert.pem'   # Server certificate
    kvserv = KeyValueServer(('', 15000),
                            keyfile=KEYFILE,
                            certfile=CERTFILE)
    kvserv.serve_forever()
```

ä¡£çŤÍè£ŹäÿłæIJ■åŁąåŹÍæÛűïijŇä¡ăåŔŕäžěä¡£çŤÍæŹőéĂŽçŽĎ `xmlrpc.client` æÍąåİŮæİěè£đæŐěåőČãĂĆ åŔŕéIJĂèęĄåIJÍURLäÿ■æŇĞåőŽ `https:` å■şåŔŕïijŇä¿ŃåęĆïijŽ

```
>>> from xmlrpc.client import ServerProxy
>>> s = ServerProxy('https://localhost:15000', allow_none=True)
>>> s.set('foo','bar')
>>> s.set('spam', [1, 2, 3])
>>> s.keys()
['spam', 'foo']
>>> s.get('foo')
'bar'
>>> s.get('spam')
[1, 2, 3]
>>> s.delete('spam')
>>> s.exists('spam')
False
>>>
```

åŕźäžŐSSLåőćæĹůçńŕæİěèőšäÿĂäÿłæŕŤè¿Čåđ’■æİĆçŽĎéŮőécŸæŸŕåęĆä¡Ţçąőèőđ’æIJ■åŁąåŹÍèŕĄäźęäÿ■åźÿçŽĎæŸŕïijŇæŽĆæŮűè£ŸæšąæIJL’äÿĂäÿłæăĞåĞĘæŰźæşŢæİěèğčåĘşè£ŹäÿłéŮőécŸïijŇéIJĂèęĄèĞłäÿ■è£ĞïijŇäÿŃéİćçżŹåĞžäÿĂäÿłä¿Ńå■ŘïijŇçŤÍæİěåžžçńŃäÿĂäÿłåőL’åĚÍçŽĎXML-RPCè£đæŐěæİěçąőèőđ’æIJ■åŁąåŹÍèŕĄäźęïijŽ

```
from xmlrpc.client import SafeTransport, ServerProxy
import ssl

class VerifyCertSafeTransport(SafeTransport):
    def __init__(self, cafile, certfile=None, keyfile=None):
        SafeTransport.__init__(self)
        self._ssl_context = ssl.SSLContext(ssl.PROTOCOL_TLSv1)
        self._ssl_context.load_verify_locations(cafile)
        if certfile:
            self._ssl_context.load_cert_chain(certfile, keyfile)
        self._ssl_context.verify_mode = ssl.CERT_REQUIRED

    def make_connection(self, host):
        # Items in the passed dictionary are passed as keyword
        # arguments to the http.client.HTTPSConnection()␣
↪constructor.
        # The context argument allows an ssl.SSLContext instance to
        # be passed with information about the SSL configuration
        s = super().make_connection((host, {'context': self._ssl_
↪context}))

        return s

# Create the client proxy
s = ServerProxy('https://localhost:15000',
                transport=VerifyCertSafeTransport('server_cert.pem
↪'),
                allow_none=True)
```

æIJ■åŁąåŹĺåřĘèŕĄäźęåŔŠéĂĄçżŹåőćæĹůçńřïijŇåőćæĹůçńřæİěçąőèőď'åőČçŽĎåŘĹæşŢæĂğãĂĆè£Źçğ
åęĆæđIJæIJ■åŁąåŹĺæČşèęĄçąőèőď'åőćæĹůçńřïijŇåŔřäžěåřĘæIJ■åŁąåŹĺåŘřåŁĺäžčçăĄä£őæŤźåęĆäÿŃïijŽ

```
if __name__ == '__main__':
    KEYFILE='server_key.pem'    # Private key of the server
    CERTFILE='server_cert.pem' # Server certificate
    CA_CERTS='client_cert.pem' # Certificates of accepted clients

    kvserv = KeyValueServer(('', 15000),
                            keyfile=KEYFILE,
                            certfile=CERTFILE,
                            ca_certs=CA_CERTS,
                            cert_reqs=ssl.CERT_REQUIRED,
                            )
    kvserv.serve_forever()
```

äÿžäžĘèőľXML-RPCåőćæĹůçńřåŔŠéĂĄèŕĄäźęïijŇä£őæŤź `ServerProxy` çŽĎåĹİåğŃåŇŰäžčçăĄåęĆäÿŃïijŽ

```
# Create the client proxy
s = ServerProxy('https://localhost:15000',
```

```
                transport=VerifyCertSafeTransport('server_cert.pem',
                                                  'client_cert.pem',
                                                  'client_key.pem'),
                allow_none=True)
```

èőÍèőž

èŕŢçİĂåŐżè£ŘèąŇæIJňèŁĆçŽĎäžčçăĄèČ¡æţŃèŕŢä¡ăçŽĎçşżçż§éĚ■ç¡őèČ¡åŁŻåŠŇçŘĘèğčSSLãĂĆ
åŘŕèČ¡æIJĂåď ğçŽĎæŇŠæĹŸæŸŕåęĆä¡Ţäÿ Ăæ■ěæ■ěçŽĎèŐůåŘŰåĹİåğŃéĚ■ç¡őkeyãĂĄèŕĄäźęåŠŇåĚűäżŰ

æĹŚèğčéĞŁäÿŃåĹřåžŢéIJĂèęĄåŢěïijŇæŕŔäÿĂäÿłSSLè£đæŐěçźĹçńŕäÿĂèĹňéČ¡äijŽæIJL’äÿĂäÿłçğĄé
è£ŹäÿłèŕĄäźęåŇĚåŘńäžĘåĚňéŠěåźűåIJÍæŕŔäÿĂæňąè£đæŐěçŽĎæŮűåĂŹéČ¡äijŽåŘŚéĂĄçżŹåŕźæŰźãĂĆ
åŕźäžŐåĚňåĚśæIJ■åŁąåŹĺïijŇåőČäżňçŽĎèŕĄäźęéĂŽåÿÿæŸŕècńæİČåÍĄèŕĄäźęæIJžædĎæŕŤåęĆVerisignãĂA
äÿžäžEçąőèőď æIJ■åŁąåŹĺç■¿åŘ■ïijŇåőćæĹůçńŕåŻđä£Ìå■ŸäÿĂäż¡åŇĚåŘńäžĘä£ąäżżæŐĹæİČæIJžædĎçŽĎ
ä¿ŃåęĆïijŇwebæţŔèğĹåŹÍä£Ìå■ŸäžĘäÿżèęĄçŽĎèőď èŕĄæIJžædĎçŽĎèŕĄäźęïijŇåźűä¡£çŤĺåőČæİěäÿżæŕŔä
åŕźæIJňåřŔèŁĆçď žä¿ŃèĂŇèĺĂïijŇåŔłæŸŕäÿžäžĘæţŃèŕŢïijŇæĹŚäžňåŔŕäžěåĹŻåżżèĞłç■¿åŘ■çŽĎèŕĄäźęïij

bash % openssl req -new -x509 -days 365 -nodes -out server_cert.pem
-keyout server_key.pem

Generating a 1024 bit RSA private key .....++++++ ...++++++

writing new private key to ‘server_key.pem’

You are about to be asked to enter information that will be incorporated into your certificate request. What you are about to enter is what is called a Distinguished Name or a DN. There are quite a few fields but you can leave some blank For some fields there will be a default value, If you enter ‘.’, the field will be left blank.

Country Name (2 letter code) [AU]:US State or Province Name (full name) [Some-State]:Illinois Locality Name (eg, city) []:Chicago Organization Name (eg, company) [Internet Widgits Pty Ltd]:Dabeaz, LLC Organizational Unit Name (eg, section) []: Common Name (eg, YOUR name) []:localhost Email Address []: bash %

åIJĺåĹŻåżżèŕĄäźęçŽĎæŮűåĂŹïijŇåŘĎäÿłåĂijçŽĎèő¿åőŽåŔŕäżěæŸŕäżżæĎŔçŽĎïijŇä¡ĘæŸŕâĂİComm
NameâĂIJçŽĎåĂijéĂŽåÿÿèęĄåŇĚåŘńæIJ■åŁąåŹĺçŽĎDNSäÿżæIJžåŘ■ãĂĆ
åęĆædIJä¡ääŔłæŸŕåIJĺæIJňæIJžæţŃèŕŢïijŇéĆčäźĹåŕšä¡£çŤĺâĂİlocalhostâĂIJïijŇåŘęåĹŹä¡£çŤĺæIJ■åŁąåŹĺç

——BEGIN RSA PRIVATE KEY—— MIICXQIBAAKBgQCZr-CNLoEyAKF+f9UNcFaz5Osa6jf7qkbUl8si5xQrY3ZYC7juu
nL1dZLn/VbEFIITaUOgvBtPv1qUWTJGwga62VSG1oFE0ODIx3g2Nh4sRf+rySsx2
L4442nx0z4O5vJQ7k6eRNHAZUUnCL50+YvjyLyt7ryLSjSuKhCcJsbZgPwIDAQAB
AoGAB5evrr7eyL4160tM5rHTeATlaLY3UBOe5Z8XN8Z6gLiB/ucSX9AysviVD/6F
3oD6z2aL8jbeJc1vHqjt0dC2dwwm32vVl8mRdyoAsQpWmiqXrkvP4Bsl04VpBeHw
Qt8xNSW9SFhceL3LEvw9M8i9MV39viih1ILyH8OuHdvJyFECQQDLEjl2d2ppxND9

PoLqVFAirDfX2JnLTdWbc+M11a9Jdn3hKF8TcxfEnFVs5Gav1MusicY5KB0ylYPb
YbTvqKc7AkEAwbnRBO2VYEZsJZp2X0IZqP9ovWokkpYx+PE4+c6MySDgaMcigL7v
WDIHJG1CHudD09GbqENasDzyb2HAIW4CzQJBAKDdkv+xoW6gJx42Auc2WzTcUHCA
eXR/+BLpPrhKykzbvOQ8YvS5W764SUO1u1LWs3G+wnRMvrRvlMCZKgggBjkCQQCG
Jewto2+a+WkOKQXrNNScCDE5aPTmZQc5waCYq4UmCZQcOjkUOiN3ST1U5iuxRqfb
V/yX6fw0qh+fLWtkOs/JAkA+okMSxZwqRtfgOFGBfwQ8/iKrnizeanTQ3L6scFXI
CHZXdJ3XQ6qUmNxNn7iJ7S/LDawo1QfWkCfD9FYoxBlg ——END RSA PRI-
VATE KEY——

æIJ■åŁąåŹÍèŕĄäźęæŰĞäżűserver_cert.pemåĘĚåőźçśżäijijäÿŃéİćè£ŹæăůïijŽ

——BEGIN CERTIFICATE—— MIIC+DCCAmGgAwIBAgIJAPMd+vi45js3MA0GCSqGSIb3DQEBB
BAYTAlVTMREwDwYDVQQIEwhJbGxpbm9pczEQMA4GA1UEBxMHQ2hpY2FnbzEUMBIG
A1UEChMLRGFiZWF6LCBMTEMxEjAQBgNVBAMTCWxvY2FsaG9zdDAeFw0xMzAxMTEx
ODQyMjdaFw0xNDAxMTExODQyMjdaMFwxCzAJBgNVBAYTAlVTMREwDwYDVQQIEwhJ
bGxpbm9pczEQMA4GA1UEBxMHQ2hpY2FnbzEUMBIGA1UEChMLRGFiZWF6LCBMTEMx
EjAQBgNVBAMTCWxvY2FsaG9zdDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
mawjS6BMgChfn/VDXBWs+TrGuo3+6pG1JfLIucUK2N2WAu47rpy9XWS5/1WxBSCE
2lDoLwbT79alFkyRsIGutlUhtaBRNDgyMd4NjYeLEX/q8krMdi+OONp8dM+DubyU

O5OnkTRwGVFJwi+dPmL48i8re68i0o0rioQnCbG2YD8CAwEAAaOBwTCBvjAdBgNV
HQ4EFgQUrtoLHHgXiDZTr26NMmgKJLJLFtIwgY4GA1UdIwSBhjCBg4AUrtoLHHgX
iDZTr26NMmgKJLJLFtKhYKReMFwxCzAJBgNVBAYTAlVTMREwDwYDVQQIEwhJbGxp
bm9pczEQMA4GA1UEBxMHQ2hpY2FnbzEUMBIGA1UEChMLRGFiZWF6LCBMTEMxEjAQ
BgNVBAMTCWxvY2FsaG9zdIIJAPMd+vi45js3MAwGA1UdEwQFMAMBAf8wDQYJKoZI
hvcNAQEFBQADgYEAFci+dqvMG4xF8UTnbGVvZJPIzJDRee6Nbt6AHQo9pOdAIMAu
WsGCplSOaDNdKKzl+b2UT2Zp3AIW4Qd51bouSNnR4M/gnr9ZD1ZctFd3jS+C5XRp
D3vvcW5lAnCCC80P6rXy7d7hTeFu5EYKtRGXNvVNd/06NALGDflrrOwxF3Y=
——END CERTIFICATE——

åIJÍæIJ■åŁąåŹÍçńŕäžčçăĄäÿ■ïijŇçğĄéŠěåŠŇèŕĄäźęæŰĞäżűäijŽèćńäijăçżŹSSLçŻÿåĚşçŽĎåŇĚèčĚåĞ¡
çğĄéŠěåžŤèŕěåIJÍä£Íå■ŸåIJÍæIJ■åŁąåŹÍäÿ■ïijŇåźűåŁääžěåőĽåĚÍä£ÍæŁď ãĂĆ

åIJÍåőćæĹůçńŕäžčçăĄäÿ■ïijŇéIJĂèęĄä£Íå■ŸäÿĂäÿłåŘĹæşŢèŕĄäźęæŐĹæİČæŰĞäżűæİěçąőèőď æIJ■åŁ
åęĆæđIJä¡ăæšąæIJL’è£ŹäÿłæŰĞäżűïijŇä¡ăåŔŕäžěåIJÍåőćæĹůçńŕåď’■åĹűäÿĂäż¡æIJ■åŁąåŹÍçŽĎèŕĄäźęåźűä¡
è£đæŐěåžžçńŇåŘŐïijŇæIJ■åŁąåŹÍäijŽæŔŘä¿ŻåőČçŽĎèŕĄäźęïijŇçĎűåŘŐä¡ăåŕśèČ¡ä¡£çŤÍåůšçżŔä£Íå■ŸçŽ

æIJ■åŁąåŹÍäź§èČ¡éĂL’æŃl’æŸŕåŘęèęĄçąőèőď’ăőćæĹůçńŕçŽĎèžńäż¡ãĂĆåęĆæđIJèęĄè£Źæăůå ĄŽçŽĎ
æIJ■åŁąåŹÍäź§éIJĂèęĄä£Íå■ŸäÿĂäÿłèćńä£ąäżżèŕĄäźęæŐĹæİČæŰĞäżűæİěçąőèőď’ăőćæĹůçńŕèŕĄäźęãĂĆ

åęĆæđIJä¡ăèęĄåIJÍçIJ§ăőđçŐŕåćČäÿ■äÿžä¡ăçŽĎç¡ŚçżIJæIJ■åŁąåŁääÿŁSSLçŽĎæŤŕæŇĄïijŇè£ŹåŕŔèŁ
ä¡ăè£ŸåžŤèŕěåŔĆèĂČåĚűäżŰçŽĎæŰĞæąčïijŇåĄŽåě¡èŁśèť źäÿ■åŕŚæŮűéŮť’æİěæţŃèŕŢåőČæ■čåÿÿåůěä¡IJ
^_^

11.11 è£ŽçÍŃéŮť’äijăéĂŠSocketæŰĞäżűæŔŔè£řçňę

éŮőécŸ

ä¡ăæIJL’åd’ŽäÿłpythonèğčéĞŁåŹÍè£ŽçÍŃåIJÍåŘŇæŮűè£ŘèąŇïijŇä¡ăæČşåřĘæ§Řäÿłæ L’ŞåijĂçŽĎæŰĞä
ærŤåęĆïijŇåĄĞèő¿æIJL’äÿłæIJ■åŁąåŹÍè£ŽçÍŃçŻÿåžŤè£đæŐěèřůæśĆïijŇä¡ĘæŸŕåőđéŹĚçŽĎçŽÿåžŤéĂżè¿Ś

èğčåĘşæŰźæąĹ

äÿžäžĘåIJĺåď'Žäÿłè£ŻçÍŃäÿ■äijăéĂŠæŰĞäžűæŔŕè£řçňęïijŇä¡ăéęŰåĚĹéIJĂèęĄåřĘåőČäžňè£đæŐěåĹřä
èĂŇåIJĺwindowsäÿŁéİćä¡ăéIJĂèęĄä¡£çŤĺåŚ¡åŘ■çőąéĄŞãĂĆäÿ■è£Ğä¡ăæŮăéIJĂçIJ§çŽĎéIJĂèęĄåŐżæŞ■ä¡
éĂŽåÿÿä¡£çŤĺ `multiprocessing` æĺąåİŮæİěåĹŻåžžè£Źæăůç ŽĎè£đæŐěäijŽæŻť'åőźæŸŞäÿĂäžŽãĂĆ

äÿĂæŮęäÿĂäÿłè£đæŐěèćńåĹŻåžžïijŇä¡ăåŔŕäžěä¡£çŤĺ `multiprocessing.reduction` äÿ■çŽĎ `send_handle()` åŠŇ `recv_handle()` åĞ¡æŢřåIJĺäÿ■åŘŇçŽĎåď'ĎçŘĘåŹĺçŻť'æŐěäijăéĂŠæŰĞäžűæŔŕè£řçňęãĂĆ äÿŃéİćçŽĎä¿Ńå■ŘæijŤçď'žäžĘæIJĂå§žæIJňçŽĎçŤĺæşŢïijŽ

```
import multiprocessing
from multiprocessing.reduction import recv_handle, send_handle
import socket

def worker(in_p, out_p):
    out_p.close()
    while True:
        fd = recv_handle(in_p)
        print('CHILD: GOT FD', fd)
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM,␣
↪fileno=fd) as s:
            while True:
                msg = s.recv(1024)
                if not msg:
                    break
                print('CHILD: RECV {!r}'.format(msg))
                s.send(msg)

def server(address, in_p, out_p, worker_pid):
    in_p.close()
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, True)
    s.bind(address)
    s.listen(1)
    while True:
        client, addr = s.accept()
        print('SERVER: Got connection from', addr)
        send_handle(out_p, client.fileno(), worker_pid)
        client.close()

if __name__ == '__main__':
    c1, c2 = multiprocessing.Pipe()
    worker_p = multiprocessing.Process(target=worker, args=(c1,c2))
    worker_p.start()

    server_p = multiprocessing.Process(target=server,
                   args=(('', 15000), c1, c2, worker_p.pid))
    server_p.start()

    c1.close()
```

```
    c2.close()
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijÑäÿď äÿłè£ŻçÍŃèćńåĹŻåżžåźűéĂŽè£ĞäÿĂäÿł
multiprocessing çőąéĄŞè£đæŐěęţůæÍěãĂĆ æIJ■åŁąåŹÍè£ŻçÍŃæL'Şåij Ääÿ Ääÿłsocketåźűç■L'å¿Ěåőćæ
åůěä¡IJè£ŻçÍŃäżĚäżĚä¡£çŤÍ recv_handle() åIJÍçőąéĄŞäÿŁéÍćç■L'å¿ĚæŐěæŤűäÿĂäÿłæŰĞäżűæŔŘè£řç
å¡ŞæIJ■åŁąåŹÍæŐěæŤűåĹřäÿĂäÿłè£đæŐëïijŇåőČåřĘäžğçŤ§çŽĎsocketæŰĞäżűæŔŘè£řçňęéĂŽè£Ğ
send_handle() äijăéĂŠçżŹåůěä¡IJè£ŻçÍŃãĂĆ åůěä¡IJè£ŻçÍŃæŐěæŤűåĹřsocketåŘŐåŘŠåőćæĹůçńřåŻđå

åęĆæđIJä¡ăä¡£çŤÍTelnetæĹŰçśżäijijåůěåĚůè£đæŐěåĹřæIJ■åŁąåŹÍïijÑäÿŃéÍćæŸŕäÿĂäÿłæijŤçď žä¿Ńå■

bash % python3 passfd.py SERVER: Got connection from ('127.0.0.1', 55543)
CHILD: GOT FD 7 CHILD: RECV b'Hellorn' CHILD: RECV b'Worldrn'

æ■ď ä¿ŃæIJĂéĞ■èęĄçŽĎéČÍåĹĘæŸŕæIJ■åŁąåŹÍæŐěæŤűåĹřçŽĎåőćæĹůçńřsocketåőđéŹĚäÿŁèćńåŘęå
æIJ■åŁąåŹÍäżĚäżĚåŘłæŸŕåřĘåĚűè¡ňæL'ŃáźűåĕşéŮ■æ■ď è£đæŐěïijÑçĎűåŘŐç■L'å¿ĚäÿŃäÿĂäÿłè£đæŐěãĂ

èőÍèőž

åŕźäžŐåď ğéČÍåĹĘçÍŃåžŘåŚŸæÍěèőšåIJÍäÿ■åŘŃè£ŻçÍŃäźŃéŮť äijăéĂŠæŰĞäżűæŔŘè£řçňęåě¡åČŔæša
ä¡ĘæŸŕïijÑæIJL'æŮűåĂŹåőČæŸŕædĎåžžäÿĂäÿłåŘŕæL'ľåśŢçşżçż§çŽĎå¿ĹæIJL'çŤÍçŽĎåůěåĚůãĂĆä¿ŃåęĆ
ä¡ăåŘŕäžěæIJL'åď ŽäÿłpythonèğčéĞŁåŹÍåőđä¿ŃïijÑářĘæŰĞäżűæŔŘè£řçňęäijăéĂŠçżŹåĚűåőČèğčéĞŁåŹÍæ

send_handle() åŠŇ recv_handle() åĞ¡æŢřåŘłèČ¡åď §çŤÍäžŐ
multiprocessing è£đæŐěãĂĆ ä¡£çŤÍåőČäžňæÍěäžčæŻ£çőąéĄŞçŽĎä¡£çŤÍïijĹåŔĆèĂČ11.7èŁĆïijL'ïijÑ
ä¿ŃåęĆïijÑä¡ăåŘŕäžěèőľ'æIJ■åŁąåŹÍåŠŇåůěä¡IJèĂĚåŘĎèĞłäžěå■ŢçŃńçŽĎçÍŃåžŔæÍěåŘŕåŁÍãĂĆäÿŃéÍćæŸ

```
# servermp.py
from multiprocessing.connection import Listener
from multiprocessing.reduction import send_handle
import socket

def server(work_address, port):
    # Wait for the worker to connect
    work_serv = Listener(work_address, authkey=b'peekaboo')
    worker = work_serv.accept()
    worker_pid = worker.recv()

    # Now run a TCP/IP server and send clients to worker
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, True)
    s.bind(('', port))
    s.listen(1)
    while True:
        client, addr = s.accept()
        print('SERVER: Got connection from', addr)

        send_handle(worker, client.fileno(), worker_pid)
        client.close()

if __name__ == '__main__':
```

```
    import sys
    if len(sys.argv) != 3:
        print('Usage: server.py server_address port', file=sys.
↪stderr)
        raise SystemExit(1)

    server(sys.argv[1], int(sys.argv[2]))
```

è£ŘèąŇè£ŹäýłæIJ■åŁąåŹĺïijŇåŔłéIJĂèęĄæĽ'ğèąŇ *python3 servermp.py /tmp/servconn 15000* ïijŇäýŇéĺćæŸŕçŽÿåžŤçŽĎåůěä¡IJèĂĚäžčçăĄïijŽ

```
# workermp.py

from multiprocessing.connection import Client
from multiprocessing.reduction import recv_handle
import os
from socket import socket, AF_INET, SOCK_STREAM

def worker(server_address):
    serv = Client(server_address, authkey=b'peekaboo')
    serv.send(os.getpid())
    while True:
        fd = recv_handle(serv)
        print('WORKER: GOT FD', fd)
        with socket(AF_INET, SOCK_STREAM, fileno=fd) as client:
            while True:
                msg = client.recv(1024)
                if not msg:
                    break
                print('WORKER: RECV {!r}'.format(msg))
                client.send(msg)

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 2:
        print('Usage: worker.py server_address', file=sys.stderr)
        raise SystemExit(1)

    worker(sys.argv[1])
```

èęĄè£ŘèąŇåůěä¡IJèĂĚïijŇæĽ'ğèąŇæĽ'ğèąŇåŚ¡äžď' *python3 workermp.py /tmp/servconn* . æŢĹædIJèů§ä¡£çŤĺPipe()ä¿Ńå■ŘæŸŕåőŇåĚĺäÿĂæăůçŽĎãĂĆ æŰĞäžűæŔŔè£řçňęçŽĎäijăéĂŠäijŽæűĽ'åŔŁåĹřUNIXå§§åěŮæŐěå■ŮçŽĎåĹŻåžžåŠŇåěŮæŐěå■ŮçŽĎ `sendmsg()` æŰźæşŢãĂĆ äÿ■è£Ğè£ŹçğæŁĂæIJŕåźűäÿ■åÿÿèğĄïijŇäýŇéĺćæŸŕä¡£çŤĺåěŮæŐěå■Ůæİěäijăé

```
# server.py
import socket

import struct
```

```
def send_fd(sock, fd):
    '''
    Send a single file descriptor.
    '''
    sock.sendmsg([b'x'],
                 [(socket.SOL_SOCKET, socket.SCM_RIGHTS, struct.
 ↪pack('i', fd))])
    ack = sock.recv(2)
    assert ack == b'OK'

def server(work_address, port):
    # Wait for the worker to connect
    work_serv = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
    work_serv.bind(work_address)
    work_serv.listen(1)
    worker, addr = work_serv.accept()

    # Now run a TCP/IP server and send clients to worker
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, True)
    s.bind(('',port))
    s.listen(1)
    while True:
        client, addr = s.accept()
        print('SERVER: Got connection from', addr)
        send_fd(worker, client.fileno())
        client.close()

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 3:
        print('Usage: server.py server_address port', file=sys.
 ↪stderr)
        raise SystemExit(1)

    server(sys.argv[1], int(sys.argv[2]))
```

äÿŃéÌćæŸŕä¡£çŤĺåěŮæŐěå■ŮçŽĎåůěä¡IJèĂĚåőđçŐřïijŽ

```
# worker.py
import socket
import struct

def recv_fd(sock):
    '''
    Receive a single file descriptor
    '''
    msg, ancdata, flags, addr = sock.recvmsg(1,
                                     socket.CMSG_LEN(struct.
 ↪calcsize('i')))
```

```
    cmsg_level, cmsg_type, cmsg_data = ancdata[0]
    assert cmsg_level == socket.SOL_SOCKET and cmsg_type == socket.
↪SCM_RIGHTS
    sock.sendall(b'OK')

    return struct.unpack('i', cmsg_data)[0]

def worker(server_address):
    serv = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
    serv.connect(server_address)
    while True:
        fd = recv_fd(serv)
        print('WORKER: GOT FD', fd)
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM,␣
↪fileno=fd) as client:
            while True:
                msg = client.recv(1024)
                if not msg:
                    break
                print('WORKER: RECV {!r}'.format(msg))
                client.send(msg)

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 2:
        print('Usage: worker.py server_address', file=sys.stderr)
        raise SystemExit(1)

    worker(sys.argv[1])
```

åęĆædIJä¡ăæČşåIJÍä¡ăçŽĎçÍŃåžŔäÿ■äijăéĂŠæŰĞäżűæŔŔè£řçňęïijŃåżžèőőä¡ăåŔĆéŸĚåĚűäżŰäÿĂäžŽ
æŕŤåęĆ `Unix Network Programming by W. Richard Stevens (Prentice Hall, 1990)`. åIJÍWindowsäÿŁäijăéĂŠæŰĞäżűæŔŔè£řçňęèů§UnixæŸŕäÿ■äÿĂæăůçŽĎïijŃåżžèőőä¡ăçă
`multiprocessing.reduction` äÿ■çŽĎæžŘäžčçăAçIJŃçIJŃåĚűåůëä¡IJåŐ§çŘĘãĂĆ

11.12 çŘĘèğčäžŃäżűéľśåŁÍçŽĎIO

éŮőécŸ

ä¡ăåžŤèŕěåůšçżŔåŘňè£Ğå§žäžŐäžŃäżűéľśåŁÍæĹŰåijĆæ■ëI/OçŽĎåŇĚïijŃä¡ĘæŸŕä¡ăè£Ÿäÿ■èČ¡åőŇåŘ
æĹŰèĂĚæŸŕåęĆædIJä¡£çŤÍåőČçŽĎèŕÏäijŽåŕźä¡ăçŽĎçÍŃåžŔäžğçŤ§äżĂäźĹå¡śåŞ■ãĂĆ

èğčåĘşæŰźæąĹ

äžŃäżűéľśåŁÍI/OæIJňèť´ÍäÿŁæİěèőšåřśæŸŕåřĘå§žæIJňI/OæŞ■ä¡IJïijĹæŕŤåęĆèŕżåŠŇåĘŹïijL'è¡ňåŇŰäÿž
ä¿ŃåęĆïijŃå¡ŞæŢřæ■őåIJÍæ§ŘäÿłsocketäÿŁèćńæŐěåŔŮåŘŐïijŃåőČäijŽè¡ňæ■ćæĹŘäÿĂäÿł

receive äžŃäžűïijŇçĎűåŘŐèćńä¡ăåőŽäźL’çŽĎåŻđèřČæŰźæşŢæĹŰåĞ¡æŢřæİěåď’ĎçŘĘãĂĆ
ä¡IJäÿžäÿĂäÿłåŔŕèČ¡çŽĎèţůåğŇçĆźïijŇäÿĂäÿłäžŇäžűél’śåŁĺçŽĎæąĘæđűåŔŕèČ¡äijŽäžěäÿĂäÿłåőđçŐřäžĘä

```
class EventHandler:
    def fileno(self):
        'Return the associated file descriptor'
        raise NotImplemented('must implement')

    def wants_to_receive(self):
        'Return True if receiving is allowed'
        return False

    def handle_receive(self):
        'Perform the receive operation'
        pass

    def wants_to_send(self):
        'Return True if sending is requested'
        return False

    def handle_send(self):
        'Send outgoing data'
        pass
```

è£Źäÿłçśžç ŽĎåőđä¿Ńä¡IJäÿžæŔŠäžűèćńæŤ¿åĚèçśżäijijäÿŃéİćè£Źæăůç ŽĎäžŃäžűå¿łçŐŕäÿ■ïijŽ

```
import select

def event_loop(handlers):
    while True:
        wants_recv = [h for h in handlers if h.wants_to_receive()]
        wants_send = [h for h in handlers if h.wants_to_send()]
        can_recv, can_send, _ = select.select(wants_recv, wants_
↪send, [])
        for h in can_recv:
            h.handle_receive()
        for h in can_send:
            h.handle_send()
```

äžŃäžűå¿łçŐŕçŽĎåĚşéŤőéČĺåĹĘæŸŕ select() `` èřČçŤĺïijŇåőČäijŽäÿ■æŰ■è¡őèŕćæŰĞ
åIJĺèřČçŤĺ ``select() äźŃåL’■ïijŇæŮűéŮť’å¿łçŐŕäijŽèŕćéŮőæL’ĂæIJL’çŽĎåď’ĎçŘĘåŹĺæİěåĘşåőŽ
çĎűåŘŐåőČářĘçżŞæđIJåĹŮèąĺæŔŘä¿ŻçżŹ select() ãĂĆçĎűåŘŐ select()
è£ŤåŽđåĞĘåď’ĞæŐěåŔŮæĹŰåŔŠéĂĄçŽĎåŕźèśąçżĎæĹŘçŽĎåĹŮèąĺãĂĆ
çĎűåŘŐçŻÿåžŤçŽĎ handle_receive() æĹŰ handle_send()
æŰźæşŢèćńèğęåŔŠãĂĆ

çijŰåĘŹåžŤçŤĺçĺŃåžŘçŽĎæŮűåĂŹïijŇEventHandler
çŽĎåőđä¿ŃäijŽèćńåĹŻåžžãĂĆä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿd’äÿłçőĂå■ŢçŽĎå§žäžŐUDPç¡ŚçżIJæIJ■åŁąçŽĎåd

```
import socket
import time
```

```
class UDPServer(EventHandler):
    def __init__(self, address):
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self.sock.bind(address)

    def fileno(self):
        return self.sock.fileno()

    def wants_to_receive(self):
        return True

class UDPTimeServer(UDPServer):
    def handle_receive(self):
        msg, addr = self.sock.recvfrom(1)
        self.sock.sendto(time.ctime().encode('ascii'), addr)

class UDPEchoServer(UDPServer):
    def handle_receive(self):
        msg, addr = self.sock.recvfrom(8192)
        self.sock.sendto(msg, addr)

if __name__ == '__main__':
    handlers = [ UDPTimeServer(('',14000)), UDPEchoServer(('',
↪15000))  ]
    event_loop(handlers)
```

æţŃèŕŢè£ŹæőţäżčçăĄïijŃèŕŢçİĂäżŐåŔęåď Űäÿ ĂäÿłpythonèğčéĞŁåŹĺè£đæŐěåőČïijŽ

```
>>> from socket import *
>>> s = socket(AF_INET, SOCK_DGRAM)
>>> s.sendto(b'',('localhost',14000))
0
>>> s.recvfrom(128)
(b'Tue Sep 18 14:29:23 2012', ('127.0.0.1', 14000))
>>> s.sendto(b'Hello',('localhost',15000))
5
>>> s.recvfrom(128)
(b'Hello', ('127.0.0.1', 15000))
>>>
```

åőđçŐřäÿ Ăäÿ łTCPæIJ■åŁąåŹĺäijŽæŻť åŁååď■æİĆäÿ ĂçĆźïijŃåŻääÿžæŕŔäÿ Ăäÿ łåőćæĹůçńŕéČ¡èęĄåĹİ äÿŃéİćæŸŕäÿ Ăäÿ łTCPåžŤç■ŤåőćæĹůçńŕä¿Ńå■ŘïijŽ

```
class TCPServer(EventHandler):
    def __init__(self, address, client_handler, handler_list):
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_
↪STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR,
↪ True)
        self.sock.bind(address)
```

```
        self.sock.listen(1)
        self.client_handler = client_handler
        self.handler_list = handler_list

    def fileno(self):
        return self.sock.fileno()

    def wants_to_receive(self):
        return True

    def handle_receive(self):
        client, addr = self.sock.accept()
        # Add the client to the event loop's handler list
        self.handler_list.append(self.client_handler(client, self.
↪handler_list))

class TCPClient(EventHandler):
    def __init__(self, sock, handler_list):
        self.sock = sock
        self.handler_list = handler_list
        self.outgoing = bytearray()

    def fileno(self):
        return self.sock.fileno()

    def close(self):
        self.sock.close()
        # Remove myself from the event loop's handler list
        self.handler_list.remove(self)

    def wants_to_send(self):
        return True if self.outgoing else False

    def handle_send(self):
        nsent = self.sock.send(self.outgoing)
        self.outgoing = self.outgoing[nsent:]

class TCPEchoClient(TCPClient):
    def wants_to_receive(self):
        return True

    def handle_receive(self):
        data = self.sock.recv(8192)
        if not data:
            self.close()
        else:
            self.outgoing.extend(data)

if __name__ == '__main__':
   handlers = []
```

```
    handlers.append(TCPServer(('',16000), TCPEchoClient, handlers))
    event_loop(handlers)
```

TCPä¿Ńå■ŘçŽĎåĚşéŤőçĆźæŸŕäžŐåď'ĎçŘĘåŹÍäÿ■åĹŮèąĺåćđåŁäåŠŇåĹăéŹď'åőćæĹůçńŕçŽĎæŞ■ä¡IJãåŕźæŕŔäÿĂäÿłè£đæŐěïijŇäÿĂäÿłæŰřçŽĎåď'ĎçŘĘåŹÍèćńåĹŻåžžåźűåŁäåĹřåĹŮèąĺäÿ■ãĂĆå¡Şè£đæŐěèćńåĚåęĆæđIJä¡äè£ŘèąŇçÍŇåžŔåźűèŕŢçİĂçŤÍTelnetæĹŰçśžäijijåůěåĚůè£đæŐěïijŇåőČäijŽårĘä¡ääŔŠéĂĄçŽĎæű

èőÍèőž

åőđéŹĚäÿŁæL'ĂæIJL'çŽĎäžŇäžűél'śåŁÍæąĘæđűåŐ§çŘĘèů§äÿŁéİćçŽĎä¿Ńå■ŘçŻÿåůőæŮăăĞãăĂĆåőđä¡ĘæŸŕåIJÍæIJĂæäÿå£ČçŽĎéČÍåĹĘïijŇéČ¡äijŽæIJL'äÿĂäÿłè¡őèŕćçŽĎå¿łçŐŕæİěæčĂæ§ěæt'żåŁÍsocketïijŇåz

äžŇäžűél'śåŁÍI/OçŽĎäÿĂäÿłåŔŕèČ¡åě¡åd'ĎæŸŕåőČèČ¡åd'ĎçŘĘéIđåÿÿåď ğçŽĎåźűåŔŚè£đæŐěïijŇèĂŇäźś§åřśæŸŕèŕt'ïijŇselect() èřČçŤÍïijĹæĹŰåĚűäžŰç■L'æŢĹçŽĎïijL'èČ¡çŻŚåŘňåď ğéĞŔçŽĎsocketåźűåŞ■åIJÍå¿łçŐŕäÿ■äÿĂæňąåď'ĎçŘĘäÿĂäÿłäžŇäžűïijŇåźűäÿ■éIJĂèęĄåĚűäžŰçŽĎåźűåŔŚæIJžåĹűãĂĆ

äžŇäžűél'śåŁÍI/OçŽĎçijžçĆźæŸŕæšąæIJL'çIJ§æ■čçŽĎåŘŇæ■ěæIJžåĹűãĂĆ åęĆæđIJäżżä¡ŢäžŇäžűåď'ĎçŘĘåŹÍæŰźæşŢéŸżåąđæĹŰæL'ğèąŇäÿĂäÿłèĂŮæŮűèőąçőŮïijŇåőČäijŽéŸżåąđæèřČçŤÍéĆčäžŻåźűäÿ■æŸŕäžŇäžűél'śåŁÍéčŐæäijçŽĎåžŞåĞ¡æŢřäź§äijŽæIJL'éŮőécŸïijŇåŘŇæăůèęĄæŸŕæ§ŁåŕźäžŐéŸżåąđæĹŰèĂŮæŮűèőąçőŮçŽĎéŮőécŸåŔŕäżěéĂŽè£ĞåřĘäžŇäžűåŔŚéĂĄäÿłåĚűäžŰå■ŢçŇňçäÿ■è£ĞïijŇåIJÍäžŇäžűå¿łçŐŕäÿ■åijŢåĚěåď'Žçž£çÍŇåŠŇåď'Žè£ŽçÍŇæŸŕæŕŤè¿ČæčŸæL'ŇçŽĎïijŇäÿŇéİćçŽĎä¿Ńå■ŘæijŤçď'žäžĘåęĆä¡Ţä¡£çŤÍ concurrent.futures æĺąåİŮæİěåőđçŐŕïijŽ

```
from concurrent.futures import ThreadPoolExecutor
import os

class ThreadPoolHandler(EventHandler):
    def __init__(self, nworkers):
        if os.name == 'posix':
            self.signal_done_sock, self.done_sock = socket.
↪socketpair()
        else:
            server = socket.socket(socket.AF_INET, socket.SOCK_
↪STREAM)
            server.bind(('127.0.0.1', 0))
            server.listen(1)
            self.signal_done_sock = socket.socket(socket.AF_INET,
                                                  socket.SOCK_
↪STREAM)
            self.signal_done_sock.connect(server.getsockname())
            self.done_sock, _ = server.accept()
            server.close()

        self.pending = []
        self.pool = ThreadPoolExecutor(nworkers)

    def fileno(self):
        return self.done_sock.fileno()
```

```
    # Callback that executes when the thread is done
    def _complete(self, callback, r):

        self.pending.append((callback, r.result()))
        self.signal_done_sock.send(b'x')

    # Run a function in a thread pool
    def run(self, func, args=(), kwargs={},*,callback):
        r = self.pool.submit(func, *args, **kwargs)
        r.add_done_callback(lambda r: self._complete(callback, r))

    def wants_to_receive(self):
        return True

    # Run callback functions of completed work
    def handle_receive(self):
        # Invoke all pending callback functions
        for callback, result in self.pending:
            callback(result)
            self.done_sock.recv(1)
        self.pending = []
```

åIJÍäžčçăĄäÿ■ïijŇrun() æŰźæşŢèćńçŤĺæİěářĘåůěä¡IJæŔŘäžd'çżŹåŻđèřČåĞ¡æŢřæšäïijŇåd'ĎçŘĘåőŇ
åőđéŹĚåůěä¡IJèćńæŔŘäžd'çżŹ ThreadPoolExecutor åőđä¿ŃãĂĆ
äÿ■è£ĞäÿĂäÿłéŽ¿çĆźæŸŕå■ŔèřČèőąçőŮçżŞæđIJåŠŇäžŃäžűå¿łçŐŕïijŇäÿžäžĘèğčåĘşåőČïijŇæĹŚäžňåĹŻåž
å¡Şçž£çÍŃæśăåőŇæĹŘåůěä¡IJåŘŐïijŇåőČäijŽæL'ğèąŇçśžäÿ■çŽĎ _complete()
æŰźæşŢãĂĆ è£ŹäÿłæŰźæşŢåĘ■æ§ŘäÿłsocketäÿŁåĘŹåĚěå■ŮèŁĆäźŃåL'■äijŽèőšæŇĆèţůçŽĎåŻđèřČåĞ¡æŢ
fileno() æŰźæşŢè£ŤåŽđåŔęåd'ŰçŽĎéĆčäÿłsocketãĂĆ åŻăæ■d'ïijŇè£Źäÿłå■ŮèŁĆèćńåĘŹåĚěæŮűïijŇå
çĎűåŘŐ handle_receive() æŰźæşŢèćńæ£Ăæt'žåźűäÿžæL'ĂæIJL'äźŃåL'■æŔŘäžd'çŽĎåůěä¡IJæL'ğèąŇ
åİęçŹ¡èőšïijŇèŕt'äžĘè£ŹäźĹåd'Žè£đæĹŚèĞłåůśéČ¡æŹŢäžĘãĂĆ
äÿŃéİćæŸŕäÿĂäÿłçőÅå■ŢçŽĎæIJ■åŁąåŹĺïijŇæijŤçd'žäžĘåęĆä¡Ţä¡£çŤĺçž£çÍŃæšăæİěåőđçŐřèĂŮæŮűçŽĎè

```
# A really bad Fibonacci implementation
def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n - 1) + fib(n - 2)

class UDPFibServer(UDPServer):
    def handle_receive(self):
        msg, addr = self.sock.recvfrom(128)
        n = int(msg)
        pool.run(fib, (n,), callback=lambda r: self.respond(r,␣
↪addr))

    def respond(self, result, addr):
        self.sock.sendto(str(result).encode('ascii'), addr)
```

```
if __name__ == '__main__':
    pool = ThreadPoolHandler(16)
    handlers = [ pool, UDPFibServer(('',16000))]
    event_loop(handlers)
```

è£ŘèąŇè£ŹäýłæIJ■åŁąåŹĺïijŇçĎűåŘŐèŕŢçİĂçŤĺåĚűåőČPythonçĺŃåžŔæİěæţŃèŕŢåőČïijŽ

```
from socket import *
sock = socket(AF_INET, SOCK_DGRAM)
for x in range(40):
    sock.sendto(str(x).encode('ascii'), ('localhost', 16000))
    resp = sock.recvfrom(8192)
    print(resp[0])
```

ä¡ăåžŤèŕěèČ¡åIJĺäý■åŘŇçłŮåŔčäý■éĞ■åd'■çŽĎæL'ğèąŇè£ŹäýłçĺŃåžŔïijŇåźűäýŤäý■äijŽå¡śåŞ■åŁřåĚ

åůšçżŔéŸĚèŕżåőŇäžĘè£ŹäýĂåřŔèŁĆïijŇéĆčäźĹä¡ăåžŤèŕěä¡£çŤĺè£ŹéĞŇçŽĎäżčçăĄåŘŮïij§äź§èőÿäý
äý■è£ĞïijŇåęĆæđIJä¡äçŘĘèğčäžĘå§žæIJňåŐ§çŘĘïijŇä¡ăåŕśèČ¡çŘĘèğčè£ŹäžŻæąĘæđűæL'Ää¡£çŤĺçŽĎæäý
ä¡IJäýžåŕźåŻđèřČåĞ¡æŢřçijŰçĺŃçŽĎæŻ£äžčïijŇäžŇäżűél'śåŁĺçijŰçăĄæIJL'æŮűåĂŹäijŽä¡£çŤĺåĹřå■ŘçĺŃï

11.13 åŔŚéĂĄäÿŐæŐěæŤűåď'ğåđŃæŢřçżĎ

éŮőécŸ

ä¡äèęĄéĂŽè£Ğç¡ŚçżIJè£đæŐěåŔŚéĂĄåŠŇæŐěåŔŮè£đçż■æŢřæ■őçŽĎåď'ğåđŃæŢřçżĎïijŇåźűår¡éĞŔ

èğčåĘşæŰźæąĹ

äÿŃéİćçŽĎåĞ¡æŢřåĹl'çŤĺ `memoryviews` æİěåŔŚéĂĄåŠŇæŐěåŔŮåď'ğæŢřçżĎïijŽ

```
# zerocopy.py

def send_from(arr, dest):
    view = memoryview(arr).cast('B')
    while len(view):
        nsent = dest.send(view)
        view = view[nsent:]

def recv_into(arr, source):
    view = memoryview(arr).cast('B')
    while len(view):
        nrecv = source.recv_into(view)
        view = view[nrecv:]
```

äÿžäžĘæţŃèŕŢçĺŃåžŔïijŇéęŰåĚĹåĹŻåžžäÿĂäÿłéĂŽè£Ğsocketè£đæŐěçŽĎæIJ■åŁąåŹĺåŠŇåőćæĹůçńŕç

```
>>> from socket import *
>>> s = socket(AF_INET, SOCK_STREAM)
```

```
>>> s.bind(('', 25000))
>>> s.listen(1)
>>> c,a = s.accept()
>>>
```

åIJÍåőćæÉlůçńřïijÍåŔęåď Űäÿ Ääÿłèğčé ĞŁåŹÍäÿ■ïijL'ïijŽ

```
>>> from socket import *
>>> c = socket(AF_INET, SOCK_STREAM)
>>> c.connect(('localhost', 25000))
>>>
```

æIJňèŁĆçŽĎçŻőæăĞæŸŕä¡äèČ¡éĂŽè£Ğè£đæŐěäijäè¿ŞäÿÄäÿłèűĚåď ğæŢřçżĎãĂĆè£Źçğ■æČĚåĘţçŽĎ
array æÍąåİŮæÉLŰ numpy æÍąåİŮæİěåŁŻåžžæŢřçżĎïijŽ

```
# Server
>>> import numpy
>>> a = numpy.arange(0.0, 50000000.0)
>>> send_from(a, c)
>>>

# Client
>>> import numpy
>>> a = numpy.zeros(shape=50000000, dtype=float)
>>> a[0:10]
array([ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.])
>>> recv_into(a, c)
>>> a[0:10]
array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.])
>>>
```

èőÍèőž

åIJÍæŢřæ■őåŕĘéŻĘåđŃåĹĘåÿČåijŔèőąçőŮåŠŇåźşèąŇèőąçőŮçÍŃåžŔäÿ■ïijŇèĞłåůśåĘŹçÍŃåžŔæİěåőđç
äÿ■è£ĞïijŇèęĄæŸŕä¡ăçąőåőđæČşè£Źæăůå ĄŽïijŇä¡ăåŔŕèČ¡éIJĂèęĄåŕĘä¡ăçŽĎæŢřæ■őè¡ňæ■ćæĹŘåŐ§ågŃ
ä¡ăåŔŕèČ¡è£ŸéIJĂèęĄåŕĘæŢřæ■őåĽĞåĽšæĹŘåď ŽäÿłåİŮïijŇåŻääÿžåď ğéČĺåĹĘåŠŇç¡ŚçżIJçŻÿåĚşçŽĎåĞ¡

äÿĂçğ■æŰźæşŢæŸŕä¡£çŤĺæ§Řçğ■æIJžåĹűåžŔåĹŮåŇŰæŢřæ■őâĂŤâĂŤåŔŕèČ¡ářĘåĚűè¡ňæ■ćæĹŘäÿĂ
äÿ■è£ĞïijŇè£ŹæăůæIJĂçżĹäijŽåĹŻåžžæŢřæ■őçŽĎäÿĂäÿłåď ■åĹűãĂĆ
ářśçőŮä¡ăåŔłæŸŕéŽűçćÓçŽĎåĄŽè£ŹäžŻïijŇä¡ăçŽĎäżčçăĄæIJĂçżĹè£ŸæŸŕäijŽæIJL'åď ğéĞŘçŽĎåřŔåđŃå

æIJňèŁĆéĂŽè£Ğä¡£çŤĺåĘĚå■ŸèğĘåŻ¿åśŢçď žäžĘäÿĂäžŹé■ŤæşŢæŞ■ä¡IJãĂĆ
æIJňèť ĺäÿŁïijŇäÿĂäÿłåĘĚå■ŸèğĘåŻ¿åřśæŸŕäÿĂäÿłåůšå■ŸåIJĺæŢřçżĎçŽĎèęĘçŽŰåśĆãĂĆäÿ■äžĚäžĚæŸŕé
åĘĚå■ŸèğĘåŻ¿è£ŸèČ¡äžěäÿ■åŘŇçŽĎæŰźåijŔè¡ňæ■ćæĹŘäÿ■åŘŇçśžåđŃæİěèąĺçŐřæŢřæ■őãĂĆ
è£ŹäÿłåřśæŸŕäÿŇéÍćè£Źäÿłèŕ■åŔěçŽĎçŽőçŽĎïijŽ

```
view = memoryview(arr).cast('B')
```

åőČæŐěåŔŮäÿĂäÿłæŢřçżĎ arråźűåřĘåĚűè¡ňæ■ćäÿžäÿĂäÿłæŮäçňęåŔůå■ŮèŁĆçŽĎåĘĚå■ŸèğĘåŻ¿ãĂĆ
æŕŤåęĆ socket.send() æĹŰ send.recv_into() ãĂĆ

åIJÍåĘĚéČÍïijŇè£ŹäžŻæŰźæşŢèČ¡åď§çŻť'æŐěæŞ■ä¡IJè£ŹäÿłåĘĚå■ŸåŇžå§§ãĂĆä¿ŃåęĆïijŇsock.
send() çŻť'æŐěäžŐåĘĚå■ŸäÿåŔŠçŢ§æŢřæ■őèĂŇäÿ■éIJĂèęĄåď■åĹűãĂĆ send.
recv_into() ä¡£çŢÍè£ŹäÿłåĘĚå■ŸåŇžå§§ä¡IJäÿžæŐěåŔŮæŞ■ä¡IJçŽĎè¿ŞåĚěçijŞåĘšåŇžãĂĆ

åL'l'äÿŃçŽĎäÿĂäÿłéŽ¿çĆźåřśæŸřsocketåĞ¡æŢřåŔřèČ¡åŔłæŞ■ä¡IJéČÍåĹĘæŢřæ■őãĂĆ
éĂŽåÿÿæİěèőšïijŇæĹŚäžňå¿Ůä¡£çŢĺå¿Ĺåď'Žäÿ■åŘŃçŽĎ send() åŠŇ recv_into()
æİěäijäè¿ŞæŢť'äÿłæŢřçżĎãĂĆ äÿ■çŢÍæŃĚå£ČïijŇæŕŔæňąæŞ■ä¡IJåŘŐïijŇèğĘåŻ¿äijŽéĂŽè£ĞåŔŚéĂĄæĹŰ
æŰřçŽĎèğĘåŻ¿åŘŇæăůäź§æŸŕåĘĚå■ŸèęEçŻŰåśĆãĂĆåŽăæ■d'ïijŇè£ŸæŸŕæšąæIJL'äżżä¡ŢçŽĎåď■åĹűæŞ

è£ŹéĞŇæIJL'äÿłéŮőécŸåřśæŸŕæŐěåŔŮèĂĚå£Ěéążäž ŃåĚĹç§ěéĄŞæIJL'åď'ŽåřŚæŢřæ■őèęĄèćńåŔŚéĂ
äżěä¿£åőČèČ¡écĎåĹĘéĚ■äÿĂäÿłæŢřçżĎæĹŰèĂĚçąőä£İåőČèČ¡åřĘæŐěåŔŮçŽĎæŢřæ■őæŤ¿åĚěäÿĂäÿłåůš
åęĆædIJæšąåŁđæşŢç§ěéĄŞçŽĎèŕÍïijŇåŔŚéĂĄèĂĚåřšå¿ŮåĚĹåřĘæŢřæ■őåď'ğåřŔåŔŚéĂĄè£ĞæİěïijŇçĎűå

çňňå■ĄäžŇçńäïijŽåźűåŔŚçijŰçĺŃ

åŕźäžŐåźűåŔŚçijŰçĺŃ, PythonæIJL'åď'Žçğ■éŢ£æIJ§æŤŕæŇĄçŽĎæŰźæşŢ,
åŇĚæŃňåď'Žçž£çĺŃ, èřČçŤĺå■Řè£ŻçĺŃ, äžěåŔŁåŘĎçğ■åŘĎæăůçŽĎåĚşäžŐçŤ§æĹŘåŹĺåĞ¡æŢřçŽĎæŁĂåů
è£ŹäÿĂçńäářĘäijŽçżŹåĞžåźűåŔŚçijŰçĺŃåŘĎçğ■æŰźéĺćçŽĎæŁĂåůğ,
åŇĚæŃňéĂŽçŤĺçŽĎåď'Žçž£çĺŃæŁĂæIJŕäžěåŔŁåźűèąŇèőąçőŮçŽĎåőđçŐřæŰźæşŢ.

åČŔçżŔéłŇäÿřářŃçŽĎçĺŃåžŔåŚŸæL'Ăç§ěéĄŞçŽĎéĆčæăů,
åď'ğåőűæŃĚå£Čåźűåŕ̊ŚçŽĎçĺŃåžŔæIJL'æ¡IJåIJĺçŽĎå■śéŹl'. åŻăæ■d',
æIJňçńăçŽĎäÿżèęĄçŽőæăĞäźŃäÿĂæŸŕçżŹåĞžæŻť'åŁăåŔŕä£ąèţŰåŠŇæŸŞèřČèŕŢçŽĎäžčçăĄ.

Contents:

12.1 åŘŕåŁĺäÿŐåĄIJæ■ćçž£çĺŃ

éŮőécŸ

ä¡ăèęĄäÿžéIJĂèęĄåźűåŔŚæL'ğèąŇçŽĎäžčçăĄåĹŻåžž/éŤĂæŕAçž£çĺŃ

èğčåĘşæŰźæąĹ

threading åžŞåŔŕäžěåIJĺå■ŢçŃňçŽĎçž£çĺŃäÿ■æL'ğèąŇäżżä¡ŢçŽĎåIJĺ
Python äÿ■åŔŕäžěèřČçŤĺçŽĎåŕźèšąăĂĆä¡ååŔŕäžěåĹŻåžžäÿĂäÿł Thread
åŕźèšąåźűåřĘä¡ăèęĄæL'ğèąŇçŽĎåŕźèšąäžě target åŔĆæŢřçŽĎå¡ćåijŔæŔŘä¿ŻçżŹèŕěåŕźèšąăĂĆ
äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎä¿Ńå■ŘïijŽ

```
# Code to execute in an independent thread
import time
def countdown(n):
    while n > 0:
        print('T-minus', n)
        n -= 1
        time.sleep(5)
```

```
# Create and launch a thread
from threading import Thread
t = Thread(target=countdown, args=(10,))
t.start()
```

å¡Şä¡ääĹŻåžžåě¡äÿĂäÿłçž£çÍŃářźèśąåŘŐïijŇèřěářźèśąåźűäÿ■äijŽçńŇå■şæL’ğèąŇïijŇéŹd’éİđä¡ăèřČçŤĺ
start() æŰźæşŢïijĹå¡Şä¡ăèřČçŤĺ start() æŰźæşŢæŮűïijŇåőČäijŽèřČçŤĺä¡ääijăéĂŠè£ŻæİěçŽĎåĞ¡æ
POSIX çž£çÍŃæĹŰèĂĚäÿĂäÿł Windows çž£çÍŃïijL’ïijŇè£ŹäžŻçž£çÍŃářĘçŤśæŞ■ä¡IJçşżçż§æİěåĚĺæİČçőąçĽ

```
if t.is_alive():
    print('Still running')
else:
    print('Completed')
```

ä¡ääź§åŘŕäžěářĘäÿĂäÿłçž£çÍŃåŁääĚěåĹřå¡ŞåL’■çž£çÍŃïijŇåźűç■L’å¿ĚåőČçżĺæ■ćïijŽ

```
t.join()
```

PythonèğčéĞŁåŹÍçŻt’åĹřæL’ĂæIJL’çž£çÍŃéČ¡çżĹæ■ćåL’■äż■ä‡İæŇą£ŘèąŇãĂĆåŕźäžŐéIJĂèęĄéŢ£æ
ä¿ŃåęĆïijŽ

```
t = Thread(target=countdown, args=(10,), daemon=True)
t.start()
```

åŘŐåŘřçž£çÍŃæŮăæşŢç■L’å¿ĚïijŇäÿ■è£ĞïijŇè£ŹäžŻçž£çÍŃäijŽåIJÍäÿżçž£çÍŃçżĹæ■ćæŮűèĞłåŁÍéŤĂa
éŹd’äžĘåęĆäÿŁæL’Ăçd’žçŽĎäÿd’äÿłæŞ■ä¡IJïijŇåźűæšąæIJL’åd’łåd’ŽåŘŕäžěářźçž£çÍŃåĄŽçŽĎäžŃæČĚãĂĆ

```
class CountdownTask:
    def __init__(self):
        self._running = True

    def terminate(self):
        self._running = False

    def run(self, n):
        while self._running and n > 0:
            print('T-minus', n)
            n -= 1
            time.sleep(5)

c = CountdownTask()
t = Thread(target=c.run, args=(10,))
t.start()
c.terminate() # Signal termination
t.join()      # Wait for actual termination (if needed)
```

åęĆædIJçž£çÍŃæL’ğèąŇäÿĂäžŻåČŔI/Oè£ŹæăůçŽĎéŸżåąđæŞ■ä¡IJïijŇéĆčäźĹéĂŽè£Ğè¡őèŕćæİěçžĹæ■
ä¿Ńå■ŘåęĆäÿŃïijŽ

```
class IOTask:
    def terminate(self):
        self._running = False

    def run(self, sock):
        # sock is a socket
        sock.settimeout(5)         # Set timeout period
        while self._running:
            # Perform a blocking I/O operation w/ timeout
            try:
                data = sock.recv(8192)
                break
            except socket.timeout:
                continue
            # Continued processing
            ...
        # Terminated
        return
```

èőÍèőž

çŤśäžŐåĚĺåśĂèğčéĞŁéŤĄŗijĹGILïijĽçŽĎåŐ§åŻăïijŇPython çŽĎçž£çÍŃèćńéŹŘåĹűåĹřåŘŇäÿĂæŮűåĹżåŘłåĚĄèőÿäÿĂäÿłçž£çÍŃæĽğèąŇè£ŹæăůäÿĂäÿłæĽğèąŇæĺąåđŃ çŽĎçž£çÍŃæŻť´éĂĆçŤĺäžŐåď´ĎçŘĘI/OåŠŇåĚűäżŮéIJĂèęĄåźűåŘŚæĽğèąŇçŽĎéŸżåąđæŞ■ä¡IJïijĹæŕŤåęĆ

æIJL'æŮűä¡ääijŽçIJŃåĹřäÿŃè¿źè£ŹçğéĂŽè£ĞçżğæĽ£ `Thread` çśżæİěåőđçŐřçŽĎçž£çÍŃïijŽ

```
from threading import Thread

class CountdownThread(Thread):
    def __init__(self, n):
        super().__init__()
        self.n = 0
    def run(self):
        while self.n > 0:

            print('T-minus', self.n)
            self.n -= 1
            time.sleep(5)

c = CountdownThread(5)
c.start()
```

åř¡çőąè£ŹæăůäźŞåŘŕäžěåůěä¡IJïijŇä¡Ęè£Źä¡£å¿Ůä¡ăçŽĎäžčçăĄä¿İèţŰäžŐ `threading` åžŞïijŇæĽĂäžěä¡ăçŽĎè£ŹäžŻäžčçăĄåŘłèČ¡åIJÍçž£çÍŃäÿŁäÿŃæŰĞäÿ■ä¡£çŤĺãĆĆäÿŁæŰĞæ `threading` åžŞæŮăåĚşçŽĎïijŇè£ŹæăůåřśäŁ£å¿Ůè£ŹäžŻäžčçăĄåŘŕäžěèćńçŤĺåIJÍåĚűäżŰçŽĎäÿŁäÿŃæŰČ `multiprocessing` æĺąåİŮåIJĺäÿĂäÿłå■ŢçŃňçŽĎè£ŽçÍŃäÿ■æĽğèąŇä¡ăçŽĎäžčçăĄïijŽ

```
import multiprocessing
c = CountdownTask(5)
p = multiprocessing.Process(target=c.run)
p.start()
```

åE■æñąéĞ■çŤşïijŇè£Źæőţäžčçă Ąäż ĚéĂĆçŤĺäžŐ CountdownTask çśżæŸŕäżěçŃńçńŇäžŐåőđéŹĚçŽĎåźűåŔŚæL’Ńæőţïij Ĺåd’Žçž£çĺŃãĂ Ąåd’Žè£ŻçĺŃç■L’ç■L’ïijL’åőđçŐřçŽĎæ

12.2 åĹd’æŰ■çž£çĺŃæŸŕåŘęåůšçżŔåŘŕåŁĺ

éŮőécŸ

ä¡ăåůšçżŔåŘŕåŁĺäžĘäÿĂäÿłçž£çĺŃïijŇä¡ĘæŸŕä¡ăæČşç§ěéĄŞåőČæŸŕäÿ■æŸŕçIJ§çŽĎåůšçżŔåijĂåğŃè£

èğčåĘşæŰźæąĹ

çž£çĺŃçŽĎäÿĂäÿłåĚşéŤőçL’źæĂğæŸŕæŕŔäÿłçž£çĺŃéČ¡æŸŕçŃňçńŃè£ŘèąŇäÿŤçŁűæĂĄäÿ■åŕŕécĎæţ `threading` åžŞäÿ■çŽĎ `Event` åŕźèśąãĂĆ `Event` åŕźèśąåŇĚåŘńäÿĂäÿłåŔŕçŤśçž£çĺŃèő¿ç¡őçŽĎä£ąåŔůæ åŕźèśąäÿ■çŽĎä£ąåŔůæăĞå£Ůèćńèő¿ç¡őäÿžåĄĞãĂĆåęĆæđIJæIJL’çž£çĺŃç■L’å¿ĚäÿĂäÿł event åŕźèśąïijŇèĂŇè£Źäÿł event åŕźèśąçŽĎæăĞå£ŮäÿžåĄĞïijŇéĆčäźĹè£Źäÿłçž£çĺŃårĘäijŽèćńäÿĂçŻt’éŸź event åŕźèśąçŽĎä£ąåŔůæăĞå£Ůèő¿ç¡őäÿžçIJ§ïijŇåőČårĘåŤd’éĘŠæL’ĂæIJL’ç■L’å¿Ěè£Źäÿł event åŕźèśąçŽĎçž£çĺŃãĆĆåęĆæđIJäÿĂäÿłçž£çĺŃç■L’å¿ĚäÿĂäÿłåůšçżŔèćńèő¿ç¡őäÿžçIJ§çŽĎ event åŕźèśąïijŇéĆčäźĹåőČårĘå£¡çŢěè£ŹäÿłäžŃäżűïijŇçżğçż■æL’ğèąŇãĂĆ äÿŃè¿źçŽĎäżčçĂąåśŢçd’žäžĘåęĆä¡Ţä¡£çŤĺ `Event` æİèå■ŔèřČçž£çĺŃçŽĎåŘŕåŁĺïijŽ

```
from threading import Thread, Event
import time

# Code to execute in an independent thread
def countdown(n, started_evt):
    print('countdown starting')
    started_evt.set()
    while n > 0:
        print('T-minus', n)
        n -= 1
        time.sleep(5)

# Create the event object that will be used to signal startup
started_evt = Event()

# Launch the thread and pass the startup event
print('Launching countdown')
t = Thread(target=countdown, args=(10,started_evt))
t.start()

# Wait for the thread to start
```

```
started_evt.wait()
print('countdown is running')
```

å¡Şä¡äæL'ğèąÑè£Źæőţäżčçă Ąïij ÑâĂIJcountdown is runningâĂİ æĂżæŸŕæŸ¿çd'žåIJÍ âĂIJcountdown startingâĂİ äźÑåŘŐæŸ¿çd'žãĂĆè£ŹæŸŕçŤśäžŐä¡£çŤÍ event æİěå■ŔèřČçž£çÍŃïijÑä¡£å¿Ůäÿżçž£çÍŃèęĄç■L'åĹř `countdown()` åĞ¡æŢřè¿ŞåĞžåŘŕåŁĺä£ąæĄŕåŘŐïijÑæL'■èČ¡çżğçż■æL'ğèąÑãĂĆ

èőĺèőž

event åŕźèśąæIJĂåě¡å■Ţæňąä¡£çŤĺïijÑåřśæŸŕèŕť'ïijÑä¡ăåĹŻåžžäÿĂäÿł event åŕźèśąïijÑèőľæ§Řäÿłçž£çÍŃç■L'å¿Ěè£ŹäÿłåŕźèśąïijÑäÿĂæŮęè£Źäÿłåŕźèśąèćńèő¿ç¡őäÿžçIJ§ïijÑä¡ăåřśåžŤèŕè `clear()` æŰźæşŢæİěéĞ■ç¡ő event åŕźèśąïijÑä¡ĘæŸŕå¿ĹéŽ¿çąőä£ĺåőL'åĚĺåIJřæÿĚçŘĘ event åŕźèśąåźűåŕźåőČéĞ■æŰřèţÑåĂijãĂĆå¿ĹåŔŕèČ¡äijŽåŔŚçŤ§éŤŹè£ĞäžÑäžűãĂĄæ■żéŤĄæĹŰèĂĚåĚűä event åŕźèśąçŽĎäžčçăĄäijŽåIJĺçž£çÍŃåĘ■æňąç■L'å¿Ěè£ŹäÿłåŕźèśąäźÑåL'■æL'ğèąÑïijL'ãĂĆåęĆæđIJäÿĂäÿłçž£çÍŃéIJĂèęĄäÿ■åĄIJåIJřéĞ■åď'■ä¡£çŤÍ event åŕźèśąïijÑä¡ăæIJĂåě¡ä¡£çŤÍ `Condition` åŕźèśąæİěäžčæŽ£ãĂĆäÿŃéİćçŽĎäžčçăĄä¡£çŤÍ `Condition` åŕźèśąåődçŐřäžĘäÿĂäÿłåŚĺæIJ§åőŽæŮűåŹĺïijÑærŔå¡ŞåőŽæŮűåŹĺèűĚæŮűçŽĎæŮűåĂŹïijÑåĺ

```
import threading
import time

class PeriodicTimer:
    def __init__(self, interval):
        self._interval = interval
        self._flag = 0
        self._cv = threading.Condition()

    def start(self):
        t = threading.Thread(target=self.run)
        t.daemon = True

        t.start()

    def run(self):
        '''
        Run the timer and notify waiting threads after each interval
        '''
        while True:
            time.sleep(self._interval)
            with self._cv:
                self._flag ^= 1
                self._cv.notify_all()

    def wait_for_tick(self):
        '''
        Wait for the next tick of the timer
        '''
        with self._cv:
```

```
            last_flag = self._flag
            while last_flag == self._flag:
                self._cv.wait()

# Example use of the timer
ptimer = PeriodicTimer(5)
ptimer.start()

# Two threads that synchronize on the timer
def countdown(nticks):
    while nticks > 0:
        ptimer.wait_for_tick()
        print('T-minus', nticks)
        nticks -= 1

def countup(last):
    n = 0
    while n < last:
        ptimer.wait_for_tick()
        print('Counting', n)
        n += 1

threading.Thread(target=countdown, args=(10,)).start()
threading.Thread(target=countup, args=(5,)).start()
```

eventåŕźèśąçŽĎäÿĂäÿłéĞ■èęAçL’źçĆźæŸŕå¡ŞåőČèćńèő¿ç¡őäÿžçIJ§æŮűäijŽåŤď’éĘŠæL’ĂæIJL’ç■L’å¿Ě
Condition åŕźèśąæİěæŽ£äžčãĂĆèĂČèŹŚäÿĂäÿŃè£Źæőţä¡£çŤĺä£ąåŔůéĞŔåőđçŐřçŽĎäžčçăĄïijŽ

```
# Worker thread
def worker(n, sema):
    # Wait to be signaled
    sema.acquire()

    # Do some work
    print('Working', n)

# Create some threads
sema = threading.Semaphore(0)
nworkers = 10
for n in range(nworkers):
    t = threading.Thread(target=worker, args=(n, sema,))
    t.start()
```

è£ŘèąŇäÿŁè¿źçŽĎäžčçăĄåřĘäijŽåŘŕåŁĺäÿĂäÿłçž£çÍŃæśäïijŇä¡ĘæŸŕåźűæšąæIJL’äżĂäźĹäžŃæČĚåŔŚ

```
>>> sema.release()
Working 0
>>> sema.release()
Working 1
>>>
```

çijŰåĘŹæűĽåŔŁåĹřåď ğéĞŔçŽĎçž£çÍŃéŮť åŘŇæ■ěéŮőécŸçŽĎäžčçăĄäijŽèőľä¡ăçŮŻäÿ■æňšçŤ§ãĂĆ

12.3 çž£çÍŃéŮťéĂŽä£ą

éŮőécŸ

ä¡ăçŽĎçÍŃåžŔäÿ■æIJĽåď Žäÿłçž£çÍŃïijŇä¡ăéIJĂèęĄåIJÍè£ŹäžŻçž£çÍŃäźŃéŮť åőĽåĚÍåIJřäžď æ■ćä£ąa

èğčåĘşæŰźæąĹ

äžŐäÿĂäÿłçž£çÍŃåŘŠåŔęäÿĂäÿłçž£çÍŃåŔŠéĂĄæŢřæ■őæIJĂåőĽåĚÍçŽĎæŰźåijŔåŔřèČ¡åřśæŸřä¡£çŤÍ queue åžŞäÿ■çŽĎéŸ§åĹŮäžĘãĂĆåĹŻåžžäÿĂäÿłèćńåď Žäÿłçž£çÍŃåĚśäžńçŽĎ Queue åŕźèśąïijŇè£ŹäžŻçž£çÍŃéĂŽè£Ğä¡£çŤÍ put() åŠŇ get() æŞ■ä¡IJæİěåŘŠéŸ§åĹŮäÿ■æůžåŁăæĹŰèĂĚåĹăéŹď åĚČçť ăãĂĆ ä¿ŃåęĆïijŽ

```
from queue import Queue
from threading import Thread

# A thread that produces data
def producer(out_q):
    while True:
        # Produce some data
        ...
        out_q.put(data)

# A thread that consumes data
def consumer(in_q):
    while True:
# Get some data
        data = in_q.get()
        # Process the data
        ...

# Create the shared queue and launch both threads
q = Queue()
t1 = Thread(target=consumer, args=(q,))
t2 = Thread(target=producer, args=(q,))
t1.start()
t2.start()
```

Queue åŕźèśąåůšçżŔåŇĚåŘńäžĘå£ĚèęĄçŽĎéŤĄïijŇæĽĂäžěä¡ååŔŕäžěéĂŽè£ĞåőČåIJÍåď Žäÿłçž£çÍŃé å¡Şä¡£çŤĺéŸ§åĹŮæŮűïijŇå■ŔèřČçŤ§äžğèĂĚåŠŇæűĹèť źèĂĚçŽĎåĚşéŮ■éŮőécŸåŔřèČ¡äijŽæIJĽäÿĂäžŹé

```
from queue import Queue
from threading import Thread

# Object that signals shutdown
_sentinel = object()
```

```
# A thread that produces data
def producer(out_q):
    while running:
        # Produce some data
        ...
        out_q.put(data)

    # Put the sentinel on the queue to indicate completion
    out_q.put(_sentinel)

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()

        # Check for termination
        if data is _sentinel:
            in_q.put(_sentinel)
            break

        # Process the data
        ...
```

æIJñä¿Ńäÿ■æIJL’äÿĂäÿłçL’źæőŁçŽĎåIJřæŰźïijŽæűĹèt’źèĂĚåIJĺèŕżåĹřè£ŹäÿłçL’źæőŁåĂijäźŃåŘŐçńŃ
åř¡çőąéŸ§åĹŮæŸŕæIJĂåÿÿèğĄçŽĎçž£çÍŃéŮt’éĂŽä£ąæIJžåĹűïijŇä¡ĘæŸŕäż■çĎůåŔŕäžěèĞłåůśéĂŽè£ĞåĹŽ
Condition åŔŸéĞŔæİěåŇĚèčĚä¡ăçŽĎæŢřæ■őçżŞæđĎãĂĆäÿŃè¿źè£ŹäÿłäżŃå■ŘæijŤçd’žäžĘåęĆä¡ŢåĹŽ

```
import heapq
import threading

class PriorityQueue:
    def __init__(self):
        self._queue = []
        self._count = 0
        self._cv = threading.Condition()
    def put(self, item, priority):
        with self._cv:
            heapq.heappush(self._queue, (-priority, self._count,␣
↪item))
            self._count += 1
            self._cv.notify()

    def get(self):
        with self._cv:
            while len(self._queue) == 0:
                self._cv.wait()
            return heapq.heappop(self._queue)[-1]
```

ä¡£çŤĺéŸ§åĹŮæİěè£ŻèąŇçž£çĺŃéŮťéĂŽä£ąæŸŕäÿĂäÿłå■ŢåŘŚãĂĄäÿ■çąőåőŽçŽĎè£ĞçĺŃãĂĆéĂŽåÿ
`task_done()` åŠŇ `join()` ïijŽ

```
from queue import Queue
from threading import Thread

# A thread that produces data
def producer(out_q):
    while running:
        # Produce some data
        ...
        out_q.put(data)

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()

        # Process the data
        ...
        # Indicate completion
        in_q.task_done()

# Create the shared queue and launch both threads
q = Queue()
t1 = Thread(target=consumer, args=(q,))
t2 = Thread(target=producer, args=(q,))
t1.start()
t2.start()

# Wait for all produced items to be consumed
q.join()
```

åęĆædIJäÿĂäÿłçž£çĺŃéIJĂèęĄåIJĺäÿĂäÿłâĂIJæűĹèťźèĂĚâĂİçž£çĺŃåď ĎçŘĘåőŇçĽźåőŽçŽĎæŢřæ■ő
`Event` æŤ¿åĹřäÿĂèţůä¡£çŤĺïijŇè£ŹæăůâĂIJçŤ§äžğèĂĚâĂİåřśåŕŕäżěéĂŽè£Ğè£ŹäÿłEventåŕźèśąæİěçŽŚæţ

```
from queue import Queue
from threading import Thread, Event

# A thread that produces data
def producer(out_q):
    while running:
        # Produce some data
        ...
        # Make an (data, event) pair and hand it to the consumer
        evt = Event()
        out_q.put((data, evt))
        ...
        # Wait for the consumer to process the item
        evt.wait()
```

```
# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data, evt = in_q.get()
        # Process the data
        ...
        # Indicate completion
        evt.set()
```

èőÍèőž

å§žäžŐçőÄå■ŢéŸ§åĹŮçijŰåĘŹåď'ŽçŽ£çÍŃçÍŃåžŘåIJÍåď'ŽæŢřæČĚåĘţäÿŃæŸŕäÿĂäÿłæŕŤè¿ČæŸŐæŹžä¡£çŤÍçž£çÍŃéŸ§åĹŮæIJL'äÿĂäÿłèęĄæşÍæĎŔçŽĎéŮőécŸæŸŕïijŇåŘŚéŸ§åĹŮäÿ■æůžåŁäæŢřæ■őéąźæŮűå

```
from queue import Queue
from threading import Thread
import copy

# A thread that produces data
def producer(out_q):
    while True:
        # Produce some data
        ...
        out_q.put(copy.deepcopy(data))

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()
        # Process the data
        ...
```

Queue åŕźèśąæŔŘä¿ŻäÿĂäžŻåIJÍå¡ŞåL'■äÿŁäÿŃæŰĞå¿ŁæIJL'çŤÍçŽĎéŹĎåŁăçL'źæĂğãĂĆæŕŤåęĆåIJ
Queue åŕźèśąæŮűæŔŘä¿ŻåŘŕéĂL'çŽĎ `size` åŘĆæŢřæİěéŹŘåĹűåŘŕäžěæůžåŁäåĹřéŸ§åĹŮäÿ■çŽĎåĚČçť'ă
âĂIJæűĹèť'źâĂİçŽĎéĂ§åžęå£ńïijŇéĆčäźŁä¡£çŤÍåŻžåőŽåď'ğåřRçŽĎéŸ§åĹŮåřśåŘŕäžěåIJÍéŸ§åĹŮåůšæżąçŹ
`get()` åŠŇ `put()` æŰźæşŢéČ¡æŤřæŇĄéİđéŸżåąđæŰźåijŘåŠŇèő¿åőŽèűĚæŮűïijŇä¿ŃåęĆïijŽ

```
import queue
q = queue.Queue()

try:
    data = q.get(block=False)
except queue.Empty:
    ...

try:
```

```
    q.put(item, block=False)
except queue.Full:
    ...

try:
    data = q.get(timeout=5.0)
except queue.Empty:
    ...
```

è£ŹäžŻæŞ■ä¡IJéČ¡åŔŕäżěçŤĺæİěéĄ£åĚ■å¡ŞæĽğèąŇæ§ŘäžŻçĽźåőŽéŸ§åĹŮæŞ■ä¡IJæŮűåŔŠçŤ§æŮăé
put() æŰźæşŢåŠŇäÿĂäÿłåŻžåőŽåď ğářŔçŽĎéŸ§åĹŮäÿĂèţůä¡£çŤĺïijŇè£Źæăůå¡ŞéŸ§åĹŮåůšæżąæŮűåřš

```
def producer(q):
    ...
    try:
        q.put(item, block=False)
    except queue.Full:
        log.warning('queued item %r discarded!', item)
```

åęĆæđIJä¡ăèŕŢåŻ¿èőľ æűĹèt´źèĂĚçž£çÍŃåIJĺæĽ ğèąŇåČŔ q.get() è£Źæăůç ŽĎæŞ■ä¡IJæŮűïijŇèűĚæŮűèĞłåŁĺçžĹæ■ćäžěä¿£æčĂæ§ěçžĹæ■ćæăĞå£ŮïijŇä¡ăåžŤèŕěä¡£çŤĺ q.get() çŽĎåŔŕéĂĽåŔĆæŢř timeout ïijŇåęĆäÿŃïijŽ

```
_running = True

def consumer(q):
    while _running:
        try:
            item = q.get(timeout=5.0)
            # Process item
            ...
        except queue.Empty:
            pass
```

æIJĂåŘŐïijŇæIJĽ q.qsize() ïijŇ q.full() ïijŇ q.empty() ç■Ľăőđç ŤĺæŰźæşŢåŔŕäžèèŐůåŔŰäÿĂäÿłéŸ§åĹŮçŽĎå¡ŞåĽ■åď ğářŔåŠŇçŁűæĂĄãĂĆä¡ĘèęĄæşĺæĎŔïijŇè empty() åĹď æŰ■åĞžè£ŹäÿłéŸ§åĹŮäÿžçľ žïijŇä¡ĘåŘŇæŮűåŔęåď ŰäÿĂäÿłçž£çÍŃåŔŕèČ¡åůšçżŔåŘŠè£Źă

12.4 çżŹåĚşéŤőéČĺåĹĘåŁăéŤĄ

éŮőécŸ

ä¡ăéIJĂèęĄåŕźåď Žçž£çÍŃçÍŃåžŔäÿ■çŽĎäÿt´çŢŃåŇžåŁăéŤĄäžěéĄ£åĚ■çńđäžĽæİąäžűãĂĆ

èğčåĘşæŰźæąĹ

èęĄåIJĺåď Žçž£çÍŃçÍŃåžŔäÿ■åőĽåĚĺä¡£çŤĺåŔŕåŔŸåŕźèśąïijŇä¡ăéIJĂèęĄä¡£çŤĺ threading åžŞäÿ■çŽĎ Lock åŕźèśąïijŇåřšåČŔäÿŃè¿źè£ŹäÿłäżŃå■Řè£Źæăůïijž

```
import threading

class SharedCounter:
    '''
    A counter object that can be shared by multiple threads.
    '''
    def __init__(self, initial_value = 0):
        self._value = initial_value
        self._value_lock = threading.Lock()

    def incr(self,delta=1):
        '''
        Increment the counter with locking
        '''
        with self._value_lock:
             self._value += delta

    def decr(self,delta=1):
        '''
        Decrement the counter with locking
        '''
        with self._value_lock:
             self._value -= delta
```

Lock åŕźèśąåŠŇ with èŕ■åŔěåİŮäÿĂèţůä¡£çŤĺåŔŕäżěä£İèŕĄäžŠæŰěæL’ğèąŇïijŇåřśæŸŕæŕŔæňąåŔłæI with èŕ■åŔěåŇĚåŘńçŽĎäżčçăĄåİŮãĂĆwith èŕ■åŔěäijŽåIJĺè£Źäÿłäžčçă Ąå İŮæL’ğèąŇåL’■èĞłåŁĺèŐůåŔŰéŤ

èőĺèőž

çž£çÍŃèřČåžęæIJňèť´ĺäÿŁæŸŕäÿ■çąőåőŽçŽĎïijŇåŻäæ■d’ïijŇåIJĺåď’Žçž£çÍŃçÍŃåžŔäÿ■éŤŹèŕŕåIJřä¡£çŤ åIJĺäÿĂäžŻâĂIJèĂĄçŽĎâĂİ Python äžčçăĄäÿ■ïijŇæŸ¿åijŔèŐůåŔŰåŠŇéĞŁæŤ¿éŤĄæŸŕå¿Ĺåÿÿèğ ĄçŽĎãĂ

```
import threading

class SharedCounter:
    '''
    A counter object that can be shared by multiple threads.
    '''
    def __init__(self, initial_value = 0):
        self._value = initial_value
        self._value_lock = threading.Lock()

    def incr(self,delta=1):
        '''
        Increment the counter with locking
        '''
        self._value_lock.acquire()
        self._value += delta
        self._value_lock.release()
```

```
    def decr(self,delta=1):
        '''
        Decrement the counter with locking
        '''
        self._value_lock.acquire()
        self._value -= delta
        self._value_lock.release()
```

çŻÿæŕŤäžŐè£Źçğ■æŸ¿åijŔèřČçŤĺçŽĎæŰźæşŢïijŇwith èŕ■åŔěæŻť'åŁääijŸéŻĚïijŇäź§æŻť'äÿ■åőźæŸŞ release() æŰźæşŢæĹŰèĂĚçĺŃåžŔåIJĺèŐůå¿ŮéŤĄäźŃåŘŐäžğçŤ§åijĆåÿÿè£Źäÿd'çğ■æČĚåĘţïijĹä¡£çŤĺ with èŕ■åŔěåŔŕäżěä£ÌèŕĄåIJĺè£Źäÿd'çğ■æČĚåĘţäÿŃäż■èČ¡æ■čçąőéĞŁæŤ¿éŤĄïijL'ãĂĆ äÿžäžĘéĄ£åĚ■åĞžçŐřæ■żéŤĄçŽĎæČĚåĘţïijŇä¡£çŤĺéŤĄæIJžåĹűçŽĎçĺŃåžŔåžŤèŕěèő¿åőŽäÿžæŕŔäÿłçž£çĺ åIJĺ `threading` åžŞäÿ■è£ŸæŔŔä¿ŻäžĘåĚűäżŰçŽĎåŘŇæ■ěåŐ§èŕ■ïijŇæŕŤåęĆ `RLock` åŠŇ `Semaphore` åŕźèśąãĂĆä¡ĘæŸŕæěźæ■őäžěå¿ĂçżŔéłŇïijŇè£ŹäžŻåŐ§èŕ■æŸŕçŤĺäžŐäÿĂäžŻçL'źæőŁçŽ `RLock` ïijĹåŔŕéĞ■åĚěéŤĄïijL'åŔŕäžěèćńåŘŇäÿĂäÿłçž£çĺŃåd'Žæňąè ŐůåŔŰïijŇäÿżèęĄçŤĺæİěåőđçŐřå§žäžĆ SharedCounter çśżïijŽ

```
import threading

class SharedCounter:
    '''
    A counter object that can be shared by multiple threads.
    '''
    _lock = threading.RLock()
    def __init__(self, initial_value = 0):
        self._value = initial_value

    def incr(self,delta=1):
        '''
        Increment the counter with locking
        '''
        with SharedCounter._lock:
            self._value += delta

    def decr(self,delta=1):
        '''
        Decrement the counter with locking
        '''
        with SharedCounter._lock:
             self.incr(-delta)
```

åIJĺäÿŁè¿źè£Źäÿłä¿Ńå■Řäÿ■ïijŇæšąæIJL'åŕźæŕŔäÿĂäÿłåőđä¿Ńäÿ■çŽĎåŔŕåŔŸåŕźèśąåŁäéŤĄïijŇåŔŰè decr æŰźæşŢãĂĆ è£Źçğ■åőđçŐřæŰźåijŔçŽĎäÿĂäÿłçL'źçĆźæŸŕïijŇæŮäèőžè£Źäÿłçśżæ IJL'åd'ŽåřŚäÿłåőđä¿ ä£qåŔůéĞŘåŕźèśąæŸŕäÿĂäÿłåžžçńŇåIJĺåĚśäžńèőąæŢřåŹĺå§žçąĂäÿŁçŽĎåŘŇæ■ěåŐ§èŕ■ãĂĆåęĆæđIJèőąæŢ èŕ■åŔěåřĘèőąæŢřåŹĺåĞŔ1ïijŇçž£çĺŃèćńåĚĄèőÿæL'ğèąŇãĂĆwith èŕ■åŔěæL'ğèąŇçżŞæİ§åŘŐïijŇèőąæŢřåŹĺåŁäïijŚãĂĆåęĆæđIJèőąæŢřåŹĺäÿž0ïijŇçž£çĺŃåřĘèćńéŸżåąđïijŇçŽ

```
from threading import Semaphore
import urllib.request
```

```
# At most, five threads allowed to run at once
_fetch_url_sema = Semaphore(5)

def fetch_url(url):
    with _fetch_url_sema:
        return urllib.request.urlopen(url)
```

åęĆæđIJä¡ăåŕźçž£çÍŃåŘŇæ■ěåŐ§èŕ■çŽĎåžŢåśĆçŘĘèőžåŠŇåőđçŐřæĎ§åĚť'èűčïijŇåŔŕäżěåŔĆèĂČæŞ

12.5 éŸšæ■ćæ■żéŤĄçŽĎåŁăéŤĄælJžåĹű

éŮőécŸ

ä¡ăæ■čåIJĺåĘŹäÿĂäÿłåď'Žçž£çÍŃçÍŃåžŔïijŇåĚűäÿ■çž£çÍŃéIJĂèęĄäÿĂæňąèŐůåŔŰåď'ŽäÿłéŤĄïijŇæ■d

èğčåĘşæŰźæąĹ

åIJÍåď'Žçž£çÍŃçÍŃåžŔäÿ■ïijŇæ■żéŤĄéŮőécŸå¿Ĺåď'ğäÿĂéČĺåĹĘæŸŕçŤšäžŐçž£çÍŃåŘŇæŮűèŐůåŔŰå
æŮűåĂŹåŘŚçŤ§éŸżåąđïijŇéĆčäźĹè£Źäÿłçž£çÍŃåřśåŔŕèČ¡éŸżåąđåĚűäżŰçž£çÍŃçŽĎæĽ'ğèąŇïijŇäżŐèĂŇå
èğčåĘşæ■żéŤĄéŮőécŸçŽĎäÿĂçğ■æŰźæąĹæŸŕäÿžçÍŃåžŔäÿ■çŽĎæŕŔäÿĂäÿłéŤĄåĹĘéĚ■äÿĂäÿłåŤŕäÿĂçŽ
æŸŕéİđåÿÿåőźæŸŞåőđçŐřçŽĎïijŇçď'žä¿ŃåęĆäÿŇïijŽ

```
import threading
from contextlib import contextmanager

# Thread-local state to stored information on locks already acquired
_local = threading.local()

@contextmanager
def acquire(*locks):
    # Sort locks by object identifier
    locks = sorted(locks, key=lambda x: id(x))

    # Make sure lock order of previously acquired locks is not␣
↪violated
    acquired = getattr(_local,'acquired',[])
    if acquired and max(id(lock) for lock in acquired) >=␣
↪id(locks[0]):
        raise RuntimeError('Lock Order Violation')

    # Acquire all of the locks
    acquired.extend(locks)
    _local.acquired = acquired

    try:
        for lock in locks:
```

```
            lock.acquire()
        yield
    finally:
        # Release locks in reverse order of acquisition
        for lock in reversed(locks):
            lock.release()
        del acquired[-len(locks):]
```

åęĆä¡Ţä¡£çŤĺè£ŹäÿłäýŁäÿŃæŰĞçőąçŘĘåŹĺåŚćïij§ä¡ääŔŕäžěæŇL’çĚğæ■čåÿÿéĂŤå¿ĎåĹŻåžžäÿĂäÿłéŤ
acquire() åĞ¡æŢřæİěçŤşèřůéŤĄïijŇ çd’žä¿ŃåęĆäÿŃïijŽ

```
import threading
x_lock = threading.Lock()
y_lock = threading.Lock()

def thread_1():
    while True:
        with acquire(x_lock, y_lock):
            print('Thread-1')

def thread_2():
    while True:
        with acquire(y_lock, x_lock):
            print('Thread-2')

t1 = threading.Thread(target=thread_1)
t1.daemon = True
t1.start()

t2 = threading.Thread(target=thread_2)
t2.daemon = True
t2.start()
```

åęĆæđIJä¡ăæL’ğèąŇè£ŹæőţäžčçăĄïijŇä¡ăäijŽåŔŚçŐřåőČå■şä¡£åIJĺäÿ■åŘŇçŽĎåĞ¡æŢřäÿ■äžěäÿ■åŘŇç
åĚűåĚşéŤőåIJĺäžŐïijŇåIJĺçňňäÿĂæőţäžčçăĄäÿ■ïijŇæĹŚäžňåŕźè£ŹäžŻéŤĄè£ŻèąŇäžĘæŐŠåžŔãĂĆéĂŽè£Ğ
åęĆæđIJæIJL’åd’Žäÿł acquire() æŞ■ä¡IJèćńåţŇåěŮèřČçŤĺïijŇåŔŕäžěéĂŽè£ĞçžçĺŃæIJňåIJřå■ŸåĆĺïijĹT
åĄĞèő¿ä¡ăçŽĎäžčçăĄæŸŕè£ŹæăůåĘŹçŽĎïijŽ

```
import threading
x_lock = threading.Lock()
y_lock = threading.Lock()

def thread_1():

    while True:
        with acquire(x_lock):
            with acquire(y_lock):
                print('Thread-1')

def thread_2():
```

```
    while True:
        with acquire(y_lock):
            with acquire(x_lock):
                print('Thread-2')

t1 = threading.Thread(target=thread_1)
t1.daemon = True
t1.start()

t2 = threading.Thread(target=thread_2)
t2.daemon = True
t2.start()
```

åęĆædIJä¡ăè£Řèąňè£ŹäýłçL'ĺæIJňçŽĎäżčçăĄïijŇå£Ěåő Žäij ŽæIJL'äýĂäýłçž£çÍŇåŔŠçŢ§åť l' æžČïijŇåi

```
Exception in thread Thread-1:
Traceback (most recent call last):
  File "/usr/local/lib/python3.3/threading.py", line 639, in _
↪bootstrap_inner
    self.run()
  File "/usr/local/lib/python3.3/threading.py", line 596, in run
    self._target(*self._args, **self._kwargs)
  File "deadlock.py", line 49, in thread_1
    with acquire(y_lock):
  File "/usr/local/lib/python3.3/contextlib.py", line 48, in __
↪enter__
    return next(self.gen)
  File "deadlock.py", line 15, in acquire
    raise RuntimeError("Lock Order Violation")
RuntimeError: Lock Order Violation
>>>
```

åŔŠçŢ§åť l' æžČçŽĎåŐ§åŽăåIJÍäžŐïijŇæŕŔäýłçž£çÍŇéČ¡èőřå¡ŢçİĂèĞłåůśåůšçżŔèŐůåŔŰåĹřçŽĎéŤĄã
acquire() åĞ¡æŢřäijŽæčĂæ§ěäźŇåL'■åůšçżŔèŐůåŔŰçŽĎéŤĄåĹÙèąÍïijŇ
çŤšäžŐéŤĄæŸŕæŇL'çĚğå■ĞåžŔæŐŠåĹÙèŐůåŔŰçŽĎïijŇæL'Ăäžěå Ğ¡æŢřäijŽèőď äýžäźŇåL'■åůšèŐůåŔŰç

èőÍèőž

æ■żéŤĄæŸŕæŕŔäýĂäýłåď Žçž£çÍŇçÍŇåžŔéČ¡äijŽéİćäýť çŽĎäýĂäýłéŮőécŸïijĹåŕšåČŔåőČæŸŕæŕŔäýĂ
çž£çÍŇåŔłèČ¡åŘŇæŮűä£ÍæŇĄäýĂäýłéŤĄïijŇè£ŹæăůçÍŇåžŔåŕšäý■äijŽècńæ■żéŤĄéŮőécŸæL'ĂåŽřæL'řãĂ

æ■żéŤĄçŽĎæčĂæţŇäýŐæĄćåď'■æŸŕäýĂäýłåĞăäźŐæšąæIJL'äijŸéŽĚçŽĎèğčåĘşæŰźæąĹçŽĎæL'l'åśŢ
è£ŘèąŇçŽĎæŮűåĂŹäijŽæŕŔéŽŤäýĂæőţæŮűéŮť éĞ■ç¡őèőąæŢřåŹÍïijŇåIJÍæšąæIJL'åŔŠçŢ§æ■żéŤĄçŽĎæČ
èűĚæŮűïijŇè£ŹæŮűçÍŇåžŔäijŽéĂŽè£ĞéĞ■åŘŕèĞłèžńæĄćåď'■åĹřæ■čåÿÿçŁűæĂĄãĂĆ

éĄ£åĚ■æ■żéŤĄæŸŕåŖęåď ŰäÿĂçğ■èğčåĘşæ■żéŤĄéŮőécŸçŽĎæŰźåijŔïijŇåIJÍè£ŽçÍŇèŐůåŔŰéŤĄçŽ
æ■żéŤĄçŁűæĂĄãĂĆèŕĄæŸŐåŕšçŢŹçżŹèŕżèĂĚä¡IJäÿžçżČäźăäžĘãĂĆéĄ£åĚ■æ■żéŤĄçŽĎäÿżèęĄæĂİæČşa
æ■żéŤĄçŽĎäÿĂäýłå£ĚèęĄæİąäžűïijŇäžŐèĂŇéĄ£åĚ■çÍŇåžŔè£ŻåĚěæ■żéŤĄçŁűæĂĄãĂĆ

äÿŇéİćäżěäÿĂäýłåĚşäžŐçž£çÍŇæ■żéŤĄçŽĎçżŔåĚÿéŮőécŸïijŽâĂIJåŞšå■ęåőűåŕšéď ŘéŮőécŸâĂİïijŇä

éİćåĽ■æIJL'äÿĂçćŮéě■åŠÑäÿĂåŔłç■ůå■ŘãĂĆåIJÍè£ŹéĞÑæŕŔäÿłåŞšå■ęåőűåŔŕäžěçIJŃåĄŽæŸŕäÿĂäÿłçŃ
æĂİèĂČãĂĄåŘČéě■äÿĽçğ■çŁűæĂĄäÿ■çŽĎäÿĂäÿłãĂĆéIJĂèęĄæşÍæĎŔçŽĎæŸŕïijÑæŕŔäÿłåŞšå■ęåőűåŘČ
éĆčäźĹäżŰäžňäžŤäÿłéČ¡åŔłèČ¡æŃ£çİĂäÿĂåŔłç■ůå■ŘåİŘåIJÍéĆčåĎ£ïijÑçŻť'åĹřéě£æ■żãĂĆæ■ď'æŮűäżŰä
äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎä¡£çŤÍæ■żéŤĄéĄ£åĚ■æIJžåĹűèğčåĘşâĂIJåŞšå■ęåőűåřšéď'ŘéŮőécŸâĂİçŽĎå

```
import threading

# The philosopher thread
def philosopher(left, right):
    while True:
        with acquire(left,right):
             print(threading.currentThread(), 'eating')

# The chopsticks (represented by locks)
NSTICKS = 5
chopsticks = [threading.Lock() for n in range(NSTICKS)]

# Create all of the philosophers
for n in range(NSTICKS):
    t = threading.Thread(target=philosopher,
                         args=(chopsticks[n],chopsticks[(n+1) %␣
↪NSTICKS]))
    t.start()
```

æIJĂåŘŐïijÑèęĄçL'źåĹńæşÍæĎŔåĹřïijÑäÿžäžĘéĄ£åĚ■æ■żéŤĄïijÑæL'ĂæIJL'çŽĎåŁăéŤĄæŞ■ä¡IJå£Ěé
`acquire()` åĞ¡æŢřãĂĆåęĆæđIJäżčçăĄäÿ■çŽĎæ§ŘéČĺåĹĘçżŢè£Ğacquire
åĞ¡æŢřçŻť'æŐěçŤşèřúéŤĄïijÑéĆčäźĹæŢť'äÿłæ■żéŤĄéĄ£åĚ■æIJžåĹűåřšäÿ■ètůä¡IJçŤĺäžĘãĂĆ

12.6 ä£İå■Ÿçž£çÍŃçŽĎçŁűæĂĄä£ąæĄŕ

éŮőécŸ

ä¡ăéIJĂèęĄä£İå■Ÿæ■čåIJÍè£ŘèąÑçž£çÍŃçŽĎçŁűæĂĄïijÑè£ŹäÿłçŁűæĂĄåŕźäžŐåĚűäżŰçŽĎçž£çÍÑæŸ

èğčåĘşæŰźæąĹ

æIJL'æŮűåIJÍåď'Žçž£çÍŃçijŰçÍÑäÿ■ïijÑä¡ăéIJĂèęĄåŘłä£İå■Ÿå¡ŞåL'■è£ŘèąÑçž£çÍŃçŽĎçŁűæĂĄãĂĆ
èęĄè£ŹäźĹåĄŽïijÑåŘŕä¡£çŤÍ `thread.local()` åĹŽåžžäÿĂäÿłæIJňåIJřçž£çÍŃå■ŸåĆĺåŕźèśąãĂĆ
åŕźè£ŹäÿłåŕźèśąçŽĎåśđæĂğçŽĎä£İå■ŸåŠÑèŕžåŘŰæŞ■ä¡IJéČ¡åŘłäijŽåŕźæL'ğèąÑçž£çÍŃåŘŕèğĄïijÑèĂÑåĚ

ä¡IJäÿžä¡£çŤÍæIJňåIJřå■ŸåĆÍçŽĎäÿĂäÿłæIJL'èűčçŽĎåőđéŹĚä¿Ńå■ŘïijÑ
èĂČèŹŚåIJÍ8.3åŕŘèŁĆåőŽäźL'è£ĞçŽĎ `LazyConnection` äÿŁäÿŃæŰĞçőąçŘĘåŹÍçśžãĂĆ
äÿŃéİćæĹŚäžňåŕźåőČè£ŻèąÑäÿĂäžŽåŕŘçŽĎä£őæŤźä¡£å¿ŮåőČåŘŕäžěéĂĆçŤÍäžŐåď'Žçž£çÍŃïijŽ

```
from socket import socket, AF_INET, SOCK_STREAM
import threading

class LazyConnection:
```

```
    def __init__(self, address, family=AF_INET, type=SOCK_STREAM):
        self.address = address
        self.family = AF_INET
        self.type = SOCK_STREAM
        self.local = threading.local()

    def __enter__(self):
        if hasattr(self.local, 'sock'):
            raise RuntimeError('Already connected')
        self.local.sock = socket(self.family, self.type)
        self.local.sock.connect(self.address)
        return self.local.sock

    def __exit__(self, exc_ty, exc_val, tb):
        self.local.sock.close()
        del self.local.sock
```

äžčçăĄäÿ■ïijŇèĞłåůśèğĆåŕ§åŕźäžŐ `self.local` åśđæĂğçŽĎä¡£çŤĺãĂĆ åőČèćńåĹİåğŃåŇŰåř¿äÿĂäÿł `threading.local()` åőđä¿ŃãĂĆ åĚűäžŰæŰźæşŢæŞ■ä¡IJèćńå■ŸåĆĺäÿž `self.local.sock` çŽĎåěŮæŐěå■Ůåŕźèśąã ĂĆæIJL'äžĘè£ŹäžŻåŕśåŘŕäžěåIJĺåd'Žçž£çĺŃäÿ■åőL'åĚĺçŽĎä¡£çŤĺ `LazyConnection` åőđä¿ŃäžĘãĂĆä¿ŃåęĆïijŽ

```
from functools import partial
def test(conn):
    with conn as s:
        s.send(b'GET /index.html HTTP/1.0\r\n')
        s.send(b'Host: www.python.org\r\n')

        s.send(b'\r\n')
        resp = b''.join(iter(partial(s.recv, 8192), b''))

    print('Got {} bytes'.format(len(resp)))

if __name__ == '__main__':
    conn = LazyConnection(('www.python.org', 80))

    t1 = threading.Thread(target=test, args=(conn,))
    t2 = threading.Thread(target=test, args=(conn,))
    t1.start()
    t2.start()
    t1.join()
    t2.join()
```

åőČäźŃæL'Ăäžěèąňå¿ŮéĂŽçŽĎåŐ§åŻăæŸŕæŕŘäÿłçž£çĺŃäijŽåĹŻåžžäÿĂäÿłèĞłåůśäÿŞåśđçŽĎåěŮæŐ
åŻăæ■d'ïijŇå¡Şäÿ■åŘŇçŽĎçž£çĺŃæL'ğèąŇåěŮæŐěå■ŮæŞ■ä¡IJæŮűïijŇçŤśäžŐæŞ■ä¡IJçŽĎæŸŕäÿ■åŘŇçŽ

èőÍèőž

åIJÍåď ğéČÍåĹEçÍŃåžŔäÿ■åĹŻåžžåŠŇæŞ■ä¡IJçž£çÍŃçL’źåőŽçŁűæĂĄåźűäÿ■äijŽæIJL’äžĂäźĹéŮőécŸãA
äÿ■è£ĞïijŇå¡ŞåĞžäžĘéŮőécŸçŽĎæŮűåĂŹïijŇéĂŽåÿÿæŸŕåŽääÿžæ§Řäÿłářźèśąèćńåď’Žäÿłçž£çÍŃä¡£çŤÍåĹi
æŕŤåęĆäÿĂäÿłåëŮæŐëå■ŮæĹŰæŰĞäžűãĂĆä¡ääÿ■èČ¡èőľ’æL’ĂæIJL’çž£çÍŃèt’ąçŇőäÿĂäÿłå■ŢçŃňåŕźèśąïijI
åŽääÿžåď’Žäÿłçž£çÍŃåŘŇæŮűèŕžåŠŇåĘŹçŽĎæŮűåĂŹäijŽäžğçŤ§æůůäźśãĂĆ
æIJňåIJřçž£çÍŃå■ŸåĆÍéĂŽè£Ğèőľ’è£ŹäžŽètĎæžŘåŔłèČ¡åIJÍècńä¡£çŤÍçŽĎçž£çÍŃäÿ■åŔŕèğĄæİèèğčåĘşè£Ź

æIJňèŁĆäÿ■ïijŇä¡£çŤÍ `thread.local()` åŔŕäžèèőľ `LazyConnection` çśżæŤŕæŇĄäÿĂäÿłçž£çÍŃäÿĂäÿłè£đæŐëïijŇ
èĂŇäÿ■æŸŕåŕźäžŐæL’ĂæIJL’çŽĎè£ŻçÍŃéČ¡åŘłæIJL’äÿĂäÿłè£đæŐëãĂĆ

åĚűåŐ§çŘĘæŸŕïijŇærŔäÿł `threading.local()` åőđä¿ŃäÿžærŔäÿłçž£çÍŃçżť’æŁď’çİĂäÿĂäÿłå■ŢçI
æL’ĂæIJL’æŹőéĂŽåőđä¿ŃæŞ■ä¡IJærŤåęĆèŐůåŔŰãĂĄä£őæŤźåŠŇåĹăéŹď’åĂijäžĚäžĚæŞ■ä¡IJè£Źäÿłå■Ůå
ærŔäÿłçž£çÍŃä¡£çŤÍäÿĂäÿłçŃňçńŇçŽĎå■ŮåĚÿářšåŔŕäžěä£İèŕĄæŢřæ■őçŽĎéŽŤçęžäžĘãĂĆ

12.7 åĹŻåžžäÿĂäÿłçž£çÍŃæśă

éŮőécŸ

ä¡ăåĹŻåžžäÿĂäÿłåůěä¡IJèĂĚçž£çÍŃæśăïijŇçŤÍæİěçŻÿåžŤåőćæĹůçńřèřůæśĆæĹŰæL’ğèąŇåĚűäžŰçŽĎå

èğčåĘşæŰźæąĹ

`concurrent.futures` åĞ¡æŢřåžŞæIJL’äÿĂäÿł `ThreadPoolExecutor` çśżåŔŕäžěèćńçŤÍæİěåőŇæĹŘè£ŹäÿłäžžåŁąãĂĆ äÿŃéİćæŸŕäÿĂäÿłçőĂå■ŢçŽĎTCPæIJ■åŁąåŹĺijŇä¡£çŤÍäžE

```
from socket import AF_INET, SOCK_STREAM, socket
from concurrent.futures import ThreadPoolExecutor

def echo_client(sock, client_addr):
    '''
    Handle a client connection
    '''
    print('Got connection from', client_addr)
    while True:
        msg = sock.recv(65536)
        if not msg:
            break
        sock.sendall(msg)
    print('Client closed connection')
    sock.close()

def echo_server(addr):
    pool = ThreadPoolExecutor(128)
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(addr)
    sock.listen(5)
```

```
    while True:
        client_sock, client_addr = sock.accept()
        pool.submit(echo_client, client_sock, client_addr)

echo_server(('',15000))
```

åęĆædIJä¡ăæČşæL’ŃåŁĺåĹŻåżžä¡äèĞłåůśçŽĎçž£çĺŃæśăïijŇ
éĂŽåÿÿåŘŕäžěä¡£çŤĺäÿĂäÿłQueueæİěè¡żæİ¿åőđçŐřãĂĆäÿŃéİćæŸŕäÿĂäÿłçĺ■å¿őäÿ■åŘŇä¡ĘæŸŕæL’ŃåŁĺåő

```
from socket import socket, AF_INET, SOCK_STREAM
from threading import Thread
from queue import Queue

def echo_client(q):
    '''
    Handle a client connection
    '''
    sock, client_addr = q.get()
    print('Got connection from', client_addr)
    while True:
        msg = sock.recv(65536)
        if not msg:
            break
        sock.sendall(msg)
    print('Client closed connection')

    sock.close()

def echo_server(addr, nworkers):
    # Launch the client workers
    q = Queue()
    for n in range(nworkers):
        t = Thread(target=echo_client, args=(q,))
        t.daemon = True
        t.start()

    # Run the server
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(addr)
    sock.listen(5)
    while True:
        client_sock, client_addr = sock.accept()
        q.put((client_sock, client_addr))

echo_server(('',15000), 128)
```

ä¡£çŤĺ `ThreadPoolExecutor` çŻÿåŕźäžŐæL’ŃåŁĺåőđçŐřçŽĎäÿĂäÿłåě¡åď’ĎåIJĺäžŐåőČä¡£å¿Ů
äżżåŁąæŔŘäžď’èĂĚæŻť’æŰźä¿£çŽĎäżŐèćńèřČçŤĺåĞ¡æŢřäÿ■èŐůåŔŰè£ŤåŻđåĂijãĂĆä¿ŃåęĆïijŇä¡ääŔŕèČ

```
from concurrent.futures import ThreadPoolExecutor
import urllib.request

def fetch_url(url):
    u = urllib.request.urlopen(url)
    data = u.read()
    return data

pool = ThreadPoolExecutor(10)
# Submit work to the pool
a = pool.submit(fetch_url, 'http://www.python.org')
b = pool.submit(fetch_url, 'http://www.pypy.org')

# Get the results back
x = a.result()
y = b.result()
```

ä¿Ńå■Řäÿ■è£ŤåŻđçŽĎhandleářźèśąäijŽåÿőä¡ăåď'ĎçŘĘæĽ'ĂæIJĽ'çŽĎéŸżåąđäÿŐå■Ŕä¡IJïijŇçĎűåŘŐä
çĽźåĹńçŽĎïijŇ`a.result()` æŞ■ä¡IJäijŽéŸżåąđè£ŻçÍŃçŻť'åĹřářźåžŤçŽĎåĞ¡æŢřæĽ'ğèąŇåőŇæĹŘåźűè£Ť

èőÍèőž

éĂŽåÿÿæİěèőšïijŇä¡ăåžŤèŕěéĄ£åĚ■çijŰåĘŹçž£çÍŃæŢřéĞŔåŔŕäżěæŮäéŹŘåĹűåćđéŢ£çŽĎçÍŃåžŔãĂĆ

```
from threading import Thread
from socket import socket, AF_INET, SOCK_STREAM

def echo_client(sock, client_addr):
    '''
    Handle a client connection
    '''
    print('Got connection from', client_addr)
    while True:
        msg = sock.recv(65536)
        if not msg:
            break
        sock.sendall(msg)
    print('Client closed connection')
    sock.close()

def echo_server(addr, nworkers):
    # Run the server
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(addr)
    sock.listen(5)
    while True:
        client_sock, client_addr = sock.accept()
        t = Thread(target=echo_client, args=(client_sock, client_
↪addr))
```

```
        t.daemon = True
        t.start()

echo_server(('',15000))
```

åř¡çőąè£ŹäÿłäźšåŘŕäżěåůěä¡IJïijŇ ä¡ĘæŸŕåőČäÿ■èČ¡æŁţå¿ąæIJL’äžžèŕŢåŻ¿éĂŽè£ĞåĹŻåżžåď’ğéĞŘçžéĂŽè£Ğä¡£çŤĺécĎåĚĹåĹİåğŃåŇŰçŽĎçž£çÍŃæśäïijŇä¡ååŘŕäżěèő¿ç¡őåŘŇæŮűè£ŘèąŇçž£çÍŃçŽĎäÿŁéŹŘa

ä¡ääŘŕèČ¡äijŽåĚşå£ČåĹŻåżžåď’ğéĞŘçž£çÍŃäijŽæIJL’äžĂäźĹåŘŐæđIJãĂĆ
çŐřäžčæŞ■ä¡IJçşżçż§åŘŕäżěå¿Ĺè¡żæİ¿çŽĎåĹŻåżžåĞäå■Čäÿłçž£çÍŃçŽĎçž£çÍŃæśäãĂĆ
çŤŽèĞşïijŇåŘŇæŮűåĞää■Čäÿłçž£çÍŃç■L’å¿Ěåůěä¡IJåźűäÿ■äijŽåŕźåĚűäżŮäžčçăĄäžğçŤ§æĂğèČ¡å¡śåŞ■ãĂĆ
å¡ŞçĎűäžĘïijŇåęĆæđIJæL’ĂæIJL’çž£çÍŃåŘŇæŮűèćńåŤď’éĘŠåźűçńŇå■şåIJÍCPUäÿŁæL’ğèąŇïijŇéĆčåřśäÿ■
éĂŽåÿÿïijŇä¡ääžŤèřěåŘłåIJÍI/Oåď’ĎçŘĘçŻÿåĚşäžčçăĄäÿ■ä¡£çŤÍçž£çÍŃæśäãĂĆ

åĹŻåżžåď’ğçŽĎçž£çÍŃæśăçŽĎäÿĂäÿłåŘŕèČ¡éIJĂèęĄåĚşæşÍçŽĎéŮőécŸæŸŕåĘĚå■ŸçŽĎä¡£çŤÍãĂĆ
ä¿ŃåęĆïijŇåęĆæđIJä¡ääIJÍOS Xçşżçż§äÿŁéİćåĹŻåżž2000äÿłçž£çÍŃïijŇçşżçż§æŸ¿çď’žPythonè£ŻçÍŃä¡£çŤÍã
äÿ■è£ĞïijŇè£ŹäÿłèőąçőŮéĂŽåÿÿæŸŕæIJL’èŕŕåůőçŽĎãĂĆå¡ŞåĹŻåżžäÿĂäÿłçž£çÍŃæŮűïijŇæŞ■ä¡IJçşżçż§äij
æŤ¿ç¡őçž£çÍŃçŽĎæL’ğèąŇæăĹïijĹéĂŽåÿÿæŸŕ8MBåď’ğåřŘïijL’ãĂĆä¡ĘæŸŕè£ŹäÿłåĘĚå■ŸåŘłæIJL’äÿĂåřŔ
åŻăæ■d’ïijŇPythonè£ŻçÍŃä¡£çŤÍåĹřçŽĎçIJ§åőđåĘĚå■ŸåĚűåőđå¿ĹåřŔ
ïijĹæŕŤåęĆïijŇåŕźäžŐ2000äÿłçž£çÍŃæİèèőšïijŇåŘłä¡£çŤÍåĹřäžĘ70MBçŽĎçIJ§åőđåĘĚå■ŸïijŇèĂŇäÿ■æŸŕ
åęĆæđIJä¡ääŃĚå£ČèŹŻæŃ§åĘĚå■Ÿåď’ğåřŘïijŇåŘŕäżěä¡£çŤÍ `threading.stack_size()` åĞ¡æŢřæİěéŹ■ä¡ŐåőČãĂĆä¿ŃåęĆïijŽ

```
import threading
threading.stack_size(65536)
```

åęĆæđIJä¡ääŁääÿŁè£Źæİąèŕ■åŘěåźűåĘ■æňąè£ŘèąŇåL’■éİćçŽĎåĹŻåżž2000äÿłçž£çÍŃèŕŢéłŇïijŇ
ä¡ääijŽåŘŠçŐřPythonè£ŻçÍŃåŘłä¡£çŤÍåĹřäžĘåď’ğæęĆ210MBçŽĎèŹŻæŃ§åĘĚå■ŸïijŇèĂŇçIJ§åőđåĘĚå■Ÿ
æşÍæĎŘçž£çÍŃæĂĹåď’ğåřŘå£Ěéążè Ğşår Šäÿž32768å■ŮèŁĆïijŇéĂŽåÿÿæŸŕçşżçż§åĘĚå■Ÿéąţåď’ğåřŘïijĹ409

12.8 çőĂå■ŢçŽĎåźűèąŇçijŰçÍŃ

éŮőécŸ

ä¡ăæIJL’äÿłçÍŃåžŘèęĄæL’ğèąŇCPUåŕĘéŻĘåđŃåůěä¡IJïijŇä¡ăæČşèőľ’äžŰåĹľçŤÍåď’ŽæäÿCPUçŽĎäijŸå

èğčåĘşæŰźæąĹ

`concurrent.futures` åžŞæŘŘä¿ŻäžĚäÿĂäÿł `ProcessPoolExecutor` çśżïijŇ
åŘŕèćńçŤÍæİěåIJĺäÿĂäÿłå■ŢçŃňçŽĎPythonèğčéĞŁåŹĺäÿ■æL’ğèąŇèőąçőŮåŕĘéŻĘåđŃåĞ¡æŢřãĂĆ
äÿ■è£ĞïijŇèęĄä¡£çŤÍåőČïijŇä¡ăéęŰåĚĹèęĄæIJL’äÿĂäžŻèőąçőŮåŕĘéŻĘåđŃçŽĎäżżåŁąãĂĆ
æĹŚäžňéĂŽè£ĞäÿĂäÿłçőĂå■ŢèĂŇåőđéŹĚçŽĎä¿Ńå■ŘæİěæijŤçď’žåőČãĂĆåĄĞåőŽä¡ăæIJL’äÿłApache
webæIJ■åŁąåŹĺæŮěå£ŮçŽőå¡ŢçŽĎgzipåŐŃçijľåŇĚïijŽ

```
logs/
   20120701.log.gz
   20120702.log.gz
```

```
   20120703.log.gz
   20120704.log.gz
   20120705.log.gz
   20120706.log.gz
   ...
```

è£ŻäÿĂæ■ěåĄĞèő¿æŕŔäÿłæŮěå£ŮæŰĞäżűåĘĚåőźçśżäijijäÿŃéİćè£Źæăůïijž

```
124.115.6.12 - - [10/Jul/2012:00:18:50 -0500] "GET /robots.txt ..."␣
↪200 71
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /ply/ ..." 200␣
↪11875
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /favicon.ico ..
↪." 404 369
61.135.216.105 - - [10/Jul/2012:00:20:04 -0500] "GET /blog/atom.xml␣
↪..." 304 -
...
```

äÿŃéİćæŸŕäÿĂäÿłèĎŽæIJňïijŇåIJÍè£ŹäžŻæŮěå£ŮæŰĞäżűäÿ■æ§ěæL'¿åĞžæL'ĂæIJL'èő£éŮőè£Ğrobot

```
# findrobots.py

import gzip
import io
import glob

def find_robots(filename):
    '''
    Find all of the hosts that access robots.txt in a single log␣
↪file
    '''
    robots = set()
    with gzip.open(filename) as f:
        for line in io.TextIOWrapper(f,encoding='ascii'):
            fields = line.split()
            if fields[6] == '/robots.txt':
                robots.add(fields[0])
    return robots

def find_all_robots(logdir):
    '''
    Find all hosts across and entire sequence of files
    '''
    files = glob.glob(logdir+'/*.log.gz')
    all_robots = set()
    for robots in map(find_robots, files):
        all_robots.update(robots)
    return all_robots

if __name__ == '__main__':
    robots = find_all_robots('logs')
```

```
    for ipaddr in robots:
        print(ipaddr)
```

åL’■éİćçŽĎçÍŃåžŘä¡£çŤÍäžĘéĂŽåÿÿçŽĎmap-reduceéčŐæäijæİěçijŰåĘŹãĂĆ åĞ¡æŢř `find_robots()` åIJÍäÿĂäÿłæŰĞäżűåŘ■éŻĘåŘĹäÿŁåĄŽmapæŞ■ä¡IJïijŇåźűåřĘçżŞæđIJæśĞæĂżäÿžäÿĂä äź§åřśæŸŕ `find_all_robots()` åĞ¡æŢřäÿ■çŽĎ `all_robots` éŻĘåŘĹãĂĆ çŐřåIJÍïijŇåĄĞèő¿ä¡äæČşèęĄä£őæŤźè£ŹäÿłçÍŃåžŘèőľ åőČä¡£çŤÍåd’Žæäÿ CPUãĂĆ å¿ĹçőĂå■ŢâĂŤâĂŤåŘłéIJĂèęĄåřĘmap()æŞ■ä¡IJæŻ£æ■ćäÿžäÿĂäÿł `concurrent.futures` åžŞäÿ■çŤ§æĹŘçŽĎçśżäijijæŞ■ä¡IJå■şåŘřãĂĆ äÿŃéİćæŸŕäÿĂäÿłçőĂå■Ţä£őæŤźçL’ĹæIJñïijŽ

```
# findrobots.py

import gzip
import io
import glob
from concurrent import futures

def find_robots(filename):
    '''
    Find all of the hosts that access robots.txt in a single log␣
↪file

    '''
    robots = set()
    with gzip.open(filename) as f:
        for line in io.TextIOWrapper(f,encoding='ascii'):
            fields = line.split()
            if fields[6] == '/robots.txt':
                robots.add(fields[0])
    return robots

def find_all_robots(logdir):
    '''
    Find all hosts across and entire sequence of files
    '''
    files = glob.glob(logdir+'/*.log.gz')
    all_robots = set()
    with futures.ProcessPoolExecutor() as pool:
        for robots in pool.map(find_robots, files):
            all_robots.update(robots)
    return all_robots

if __name__ == '__main__':
    robots = find_all_robots('logs')
    for ipaddr in robots:
        print(ipaddr)
```

éĂŽè£Ğè£ŹäÿłäŁőæŤźåŘŐïijŇè£ŘèąŇè£ŹäÿłèĎŽæIJñäžğçŤ§åŘŇæăůçŽĎçżŞæđIJïijŇä¡ĘæŸŕåIJĺåŽŻä åőđéŹĚçŽĎæĂğèČ¡äijŸåŇŰæŢĹæđIJæăźæ■őä¡ăçŽĎæIJžåŹĺCPUæŢřéĞŘçŽĎäÿ■åŘŇèĂŇäÿ■åŘŇãĂĆ

èőÍèőž

ProcessPoolExecutor çŽĎåËÿåđŃçŤÍæşŢåęĆäÿŃïijŽ

```
from concurrent.futures import ProcessPoolExecutor

with ProcessPoolExecutor() as pool:
    ...
    do work in parallel using pool
    ...
```

åĚűåŐ§çŘĘæŸŕïijŇäÿĂäÿł ProcessPoolExecutor åĹŻåžžNäÿłçŃňçńŃçŽĎPythonèğčéĞŁåŹĺïijŇ NæŸŕçşżçż§äÿŁéÍćåŘŕçŤÍCPUçŽĎäÿłæŢřãĂĆä¡ääŘŕäżěéĂŽ ProcessPoolExecutor(N) æÍëä£őæŤź åď ĎçŘĘåŹĺæŢřéĞŘãĂĆè£Źäÿłåď ĎçŘĘæšääijŽäÿĂçŻť è£Řèą çĎűåŘŐåď ĎçŘĘæšäèćńåĚşéŮ■ãĂĆäÿ■è£ĞïijŇçÍŃäžŘäijŽäÿĂçŻť ç■L'å¿ĚçŻť åĹřæL'ĂæIJL'æŘŘäžď çŽĎå

èćńæŘŘäžď åĹřæšääÿ■çŽĎåůěä¡IJå£ĚéążèćńåőŽäźL'äÿžäÿĂäÿłåĞ¡æŢřãĂĆæIJL'äÿď çğ■æŰźæşŢåŐżæ åęĆæđIJä¡ăæČşèőľäÿĂäÿłåĹŮèąĺæŐĺåŕijæĹŰäÿĂäÿł map() æŞ■ä¡IJåźűèąŇæL'ğèąŇçŽĎèŕİïijŇåŘŕä¡£çŤÍ pool.map() :

```
# A function that performs a lot of work
def work(x):
    ...
    return result

# Nonparallel code
results = map(work, data)

# Parallel implementation
with ProcessPoolExecutor() as pool:
    results = pool.map(work, data)
```

åŘęåď ŰïijŇä¡ääŘŕäżěä¡£çŤÍ pool.submit() æÍěæL'ŃåŁÍçŽĎæŘŘäžď å■ŢäÿłäżżåŀąïijŽ

```
# Some function
def work(x):
    ...
    return result

with ProcessPoolExecutor() as pool:
    ...
    # Example of submitting work to the pool
    future_result = pool.submit(work, arg)

    # Obtaining the result (blocks until done)
    r = future_result.result()
    ...
```

åęĆæđIJä¡ăæL'ŃåŁÍæŘŘäžď äÿĂäÿłäżżåŀąïijŇçżŞæđIJæŸŕäÿĂäÿł Future åőđä¿ŃãĂĆ èęĄèŐůåŘŰæIJĂçżĹçżŞæđIJïijŇä¡ăéIJĂèęĄèřČçŤĺåőČçŽĎ result() æŰźæşŢãĂĆ åőČäijŽéŸżåąđè£ŽçÍŃçŻť åĹřçżŞæđIJècńè£ŤåŽđæÍěãĂĆ

åęĆædIJäÿ■æČşéŸżåąđïijŇä¡ăè£ŸåŘŕäžěä¡£çŤĺäÿĂäÿłåŻđèřČåĞ¡æŢřïijŇä¿ŃåęĆïijŽ

```
def when_done(r):
    print('Got:', r.result())

with ProcessPoolExecutor() as pool:
     future_result = pool.submit(work, arg)
     future_result.add_done_callback(when_done)
```

åŽđèřČåĞ¡æŢřæŐěåŔŮäÿĂäÿł `Future` åőđä¿ŃïijŇèćńçŤĺæİěèŐůåŔŮæIJĂçżĹçŽĎçżŞædIJïijĹærŤåęĆ
ař¡çőąåđ'ĎçŘĘæśăå¿ĹåőźæŸŞä¡£çŤĺïijŇåIJĺèő¿èőąåđ'ğçÍŃåžŔçŽĎæŮűåĂŹè£ŸæŸŕæIJL'å¿Ĺåđ'ŽéIJĂèęĄæ

- è£Źçğ■ázűèąŇåđ'ĎçŘĘæŁĂæIJŕåŔłéĂĆçŤĺäžŐéĆčäžŻåŔŕäžěèćńåĹĘèğčäÿžäžŠçŻÿçŃňçńŃéČĺåĹĘçŽ
- èćńæŔŘäžđ'çŽĎäżżåŁąå£Ěéążæ ŸŕçőĂå■ŢåĞ¡æŢřå¡ćåijŔãĂĆåŕźäžŐæŰźæşŢãĂĄéŮ■åŇĚåŠŇåĚűäżŮ
- åĞ¡æŢřåŔĆæŢřåŠŇè£ŤåŻđåÄijå£ĚéążåijåőźpickleïijŇåŻääÿžèęĄä¡£çŤĺåĹřè£ŻçÍŃéŮt'çŽĎéĂŽä£ąï
- èćńæŔŘäžđ'çŽĎäżżåŁąåĞ¡æŢřäÿ■åžŤä£İçŢŹçŁűæĂĄæĹŰæIJL'åL'ŕä¡IJçŤĺãĂĆéŹđ'äžĘæL'Şå■řæŮěå

äÿĂæŮęåŘŕåŁĺä¡ääÿ■èČ¡æŐğåĹűå■Řè£ŻçÍŃçŽĎäżżä¡Ţèą Ňäÿžïij ŇåŻäæ■đ'æIJĂåě¡ä£İæŇĄçőĂå■ŢåŠ

- åIJÍUnixäÿŁè£ŻçÍŃæśäéĂŽè£ĞèřČçŤÍ `fork()` çşżçż§èřČçŤĺèćńåĹŽåżžïijŇ

åőČäijŽåĚŃéŽĘPythonèğčéĞŁåŹĺïijŇåŇĚæŃňforkæŮűçŽĎæL'ĂæIJL'çÍŃåžŘçŁűæĂĄãĂĆ
èĂŇåIJÍWindowsäÿŁïijŇåĚŃéŽĘèğčéĞŁåŹĺæŮűäÿ■äijŽåĚŃéŽĘçŁűæĂĄãĂĆ åőđéŹĚçŽĎ-
forkæŞ■ä¡IJäijŽåIJÍçňňäÿĂæňąèřČçŤÍ `pool.map()` æĹŰ `pool.submit()`
åŘŐåŘŚçŤ§ãĂĆ

- å¡Şä¡ăæůůåŘĹä¡£çŤĺè£ŻçÍŃæśăåŠŇåd'Žçž£çÍŃçŽĎæŮűåŹèęĄçL'źåĹńářŘå£ČãĂĆ

ä¡ăåžŤèŕěåIJÍåĹŻåżžäżżä¡Ţçž£çÍŃäźŃåL'■åĚĹåĹŻåżžåźűæ£Ăæť'è£ŻçÍŃæśäïijĹærŤåęĆåIJÍçÍŃåžŘåŘŕ

12.9 PythonçŽĎåĚÍåśĂéŤĄéŮőécŸ

éŮőécŸ

ä¡ăåůšçżŘåŘňèŕť'è£ĞåĚÍåśĂèğčéĞŁåŹĺéŤĄGILïijŇæŃĚå£ČåőČäijŽå¡śåŞ■åĹřåď'Žçž£çÍŃçÍŃåžŘçŽĎæ

èğčåĘşæŰźæąĹ

åř¡çőąPythonåőŇåĚÍæŤŕæŇĄåď'Žçž£çÍŃçijŰçÍŃïijŇ ä¡ĘæŸŕèğčéĞŁåŹĺçŽĎCèŕ■èÍĂåőđçŐŕéČÍåĹĘåIJÍ
åőđéŹĚäÿŁïijŇèğčéĞŁåŹÍèćńäÿĂäÿłåĚÍåśĂèğčéĞŁåŹĺéŤĄä£İæŁd'çİĂïijŇåőČçąőä£İäżżä¡ŢæŮűåĂŹéČ¡åŔ
GILæIJĂåď'ğçŽĎéŮőécŸåŕśæŸŕPythonçŽĎåď'Žçž£çÍŃçÍŃåžŘåźűäÿ■èČ¡åĹl'çŤĺåď'ŽæäÿCPUçŽĎäijŸåŁ£
ïijĹærŤåęĆäÿĂäÿłä¡£çŤĺäžĘåď'Žäÿłçž£çÍŃçŽĎèőąçőŮåŕĘéŽĘåđŃçÍŃåžŘåŘłäijŽåIJÍäÿĂäÿłå■ŢCPUäÿŁéÍć

åIJÍèőÍèőžæŹőéĂŽçŽĎGILäźŃåL'■ïijŇæIJL'äÿĂçĆźèęĄåijžèŕČçŽĎæŸŕGILåŘłäijŽå¡śåŞ■åĹřéĆčäžŽäÿ
åęĆædIJä¡ăçŽĎçÍŃåžŘåd'ğéČĺåĹĘåŘłäijŽæűL'åŘŁåĹřI/OïijŇæŕŤåęĆç¡ŚçżIJäžđ'äžŠïijŇéĆčäźĹä¡£çŤĺåd'Žç
åŽääÿžåőČäżňåď'ğéČĺåĹĘæŮűéŮt'éČ¡åIJÍç■L'å¿ĚãĂĆåőđéŹĚäÿŁïijŇä¡ăåőŇåĚÍåŘŕäžěæŤ¿å£ČçŽĎåĹŽåżžå
çŐŕäžčæŞ■ä¡IJçşżçż§è£ŘèąŇè£ŹäźĹåd'Žçž£çÍŃæšąæIJL'äżżä¡ŢåŐŃåŁŻïijŇæšąåŢěåŘŕæŃĚå£ČçŽĎãĂĆ

èĂŇåŕźäžŐä¿İèţŰCPUçŽĎçÍŃåžŔïijŇä¡ăéIJĂèęĄåijĎæÿĚæěŽæĽğèąŇçŽĎèőąçőŮçŽĎçĽźçĆźãĂĆ
ä¿ŃåęĆïijŇäijŸåŇŰåžŢåśĆçőŮæşŢèęĄæŕŤä¡£çŤĺåď´Žçž£çÍŃè£ŘèąŇå£ńå¿Ůåď´ŽãĂĆ
çśżäijijçŽĎïijŇçŤśäžŐPythonæŸŕèğčéĞŁæĽğèąŇçŽĎïijŇåęĆæđIJä¡ăåřĘéĆčäžŽæĂğèČ¡çŞűéćĹäžčçăĄçğżă
éĂ§åžęäź§äijŽæŔŘå■ĞçŽĎå¿Ĺå£ńãĂĆåęĆæđIJä¡ăèęĄæŞ■ä¡IJæŢřçżĎïijŇéĆčäźĹä¡£çŤĺNumPyè£ŹæăůçŽĎ
æIJĂåŘŐïijŇä¡ăè£ŸåŔŕäžěèĂČèŹŚäÿŃåĚűäžŰåŔŕéĂĽ´ăőđçŐřæŰźæąĹïijŇærŤåęĆPyPyïijŇåőČéĂŽè£ĞäÿĂ
ïijĹäÿ■è£ĞåIJĺåĘŹè£ŹæIJňäźęçŽĎæŮűåĂŹåőČè£Ÿäÿ■èČ¡æŤŕæŇĄPython 3ïijĽ´ãĂĆ

è£ŸæIJĽäÿĂçĆźèęĄæşĺæĎŔçŽĎæŸŕïijŇçž£çÍŃäÿ■æŸŕäÿŞéŮĺçŤĺæİěäijŸåŇŰæĂğèČ¡çŽĎãĂĆ
äÿĂäÿłCPUä¿İèţŰåđŃçÍŃåžŔåŔŕèČ¡äijŽä¡£çŤĺçž£çÍŃæİěçőąçŘĘäÿĂäÿłåŻ¿å¡ćçŤĺæĹůçŢŇéİcãĂĄäÿĂäÿłç¡
è£ŹæŮűåĂŹïijŇGILäijŽäžğçŤ§äÿĂäžŹéŮőécŸïijŇåŽääÿžåęĆæđIJäÿĂäÿłçž£çÍŃéŢ£æIJ§æŇĄæIJĽGILçŽĎ
äžŇåődäÿŁïijŇäÿĂäÿłåĘŹçŽĎäÿ■åě¡çŽĎCèŕ■èÍĂæĽľ´åśŢäijŽåŕijèĞť’è£ŹäÿłéŮőécŸæŻť’åŁääÿěéĞ■ïijŇ
ař¡çőąäžčçăĄçŽĎèőąçőŮéČÍåĹĘäijŽæŕŤäźŇåĽ■è£ŘèąŇçŽĎæŻť’å£ńäžŽãĂĆ

èŕť’äžĘè£ŹäźĹåď´ŽïijŇçŐřåIJÍæČşèŕť’çŽĎæŸŕæĹŚäžňæIJĽäÿd’çğ■ç■ŰçŢěæİěèğčåĘşGILçŽĎçijžçĆźã
éęŰåĚĹïijŇåęĆæđIJä¡ăåőŇåĚĺåůěä¡IJäžŐPythonçŐŕåćČäÿ■ïijŇä¡ăåŔŕäžěä¡£çŤĺ
`multiprocessing` æĺąåİŮæİěåĹŻåžžäÿĂäÿłè£ŻçÍŃæśäïijŇ
åźűåČŔå■ŔåŘŇåď´ĎçŘĘåŹĺäÿĂæăůçŽĎä¡£çŤĺåőČãĂĆä¿ŃåęĆïijŇåĄĞåęĆä¡ăæIJĽåęĆäÿŇçŽĎçž£çÍŃäžčç

```
# Performs a large calculation (CPU bound)
def some_work(args):
    ...
    return result

# A thread that calls the above function
def some_thread():
    while True:
        ...
        r = some_work(args)
    ...
```

ä£őæŤźäžčçăĄïijŇä¡£çŤĺè£ŻçÍŃæśäïijŽ

```
# Processing pool (see below for initiazation)
pool = None

# Performs a large calculation (CPU bound)
def some_work(args):
    ...
    return result

# A thread that calls the above function
def some_thread():
    while True:
        ...
        r = pool.apply(some_work, (args))
        ...

# Initiaze the pool
if __name__ == '__main__':
    import multiprocessing
    pool = multiprocessing.Pool()
```

è£Źäÿłéžè£Ğä¡£çŤĺäÿĂäÿłæŁĂåůğåĹľçŤĺè£ŻçÍŃæśăèğčåĘşäžĘGILçŽĎéŮőécŸãĂĆ
å¡ŞäÿĂäÿłçž£çÍŃæČşèęĄæĽ'ğèąŇCPUåŕĘéŻĘåđŃåůëä¡IJæŮűïijŇäijŽåřĘäżżåŁąåŔŚçżŹè£ŻçÍŃæśăãĂĆ
çĎűåŘŐè£ŻçÍŃæśăäijŽåIJĺåŔęåď'ŰäÿĂäÿłè£ŻçÍŃäÿ■åŘŕåŁĺäÿĂäÿłå■ŢçŃňçŽĎPythonèğčéĞŁåŹĺæİěåůëä¡I
å¡Şçž£çÍŃç■Ľ'å¿ĚçżŞæđIJçŽĎæŮűåĂŹäijŽéĞŁæŤ¿GILãĂĆ åźűäÿŤïijŇçŤśäžŐèőąçőŮäżżåŁąåIJĺå■ŢçŃňèğ
åIJĺäÿĂäÿłåď'Žæäÿçşżçż§äÿŁéİćïijŇä¡ăäijŽåŔŚçŐřè£Źäÿłæ Ł ĂæIJŕåŔŕäžěèľä¡ăå¿Łåě¡çŽĎåĹľçŤĺåď'ŽCPU

åŔęåď'ŰäÿĂäÿłèğčåĘşGILçŽĎç■ŰçŢěæŸŕä¡£çŤĺCæĽ'ľåśŢçijŰçÍŃæŁĂæIJŕãĂĆ
äÿżèęĄæĂİæČşæŸŕářĘèőąçőŮářĘéŻĘåđŃäżżåŁąè¡ňçğżçżŹCïijŇèů§PythonçŃňçńŃïijŇåIJĺåůëä¡IJçŽĎæŮűŻ
è£ŹåŔŕäżěéĂŽè£ĞåIJĺCäžčçăĄäÿ■æŔŠåĚëäÿŃéİćè£ŹæăůçŽ Ďç Ľ'źæőŁåőŔæİěåőŇæĹŘïijŽ

```
#include "Python.h"
...

PyObject *pyfunc(PyObject *self, PyObject *args) {
   ...
   Py_BEGIN_ALLOW_THREADS
   // Threaded C code
   ...
   Py_END_ALLOW_THREADS
   ...
}
```

åęĆæđIJä¡ăä¡£çŤĺåĚűäžŮåůěåĚůèő£éŮőCèŕ■èĺĂïijŇæŕŤåęĆåŕźäžŐCythonçŽĎctypesåžŞïijŇä¡ăäÿ■éIJĂ
ä¿ŃåęĆïijŇctypesåIJĺèřČçŤĺCæŮűäijŽèĞłåŁĺéĞŁæŤ¿GILãĂĆ

èőĺèőž

èőÿåď'ŽçÍŃåžŔåŚŸåIJĺéİćåŕźçž£çÍŃæĂğèČ¡éŮőécŸçŽĎæŮűåĂŹïijŇél'ňäÿŁåřśäijŽæĂłç¡łGILïijŇäżĂ
åĚűåőđè£Źæăůå■Řåď'łäÿ■åŐŽéĄŞäź§åď'łåď'l'çIJ§äžĘçĆźãĂĆ
ä¡IJäÿžäÿĂäÿłçIJ§åőđçŽĎä¿Ńå■ŘïijŇåIJĺåď'Žçž£çÍŃçŽĎç¡ŚçżIJçijŰçÍŃäÿ■çěđçğŸçŽĎ
stalls åŔŕèČ¡æŸŕåŻääÿžåĚűäžŮåŐ§åŻăæŕŤåęĆäÿĂäÿłDNSæ§ěæĽ'¿åžűæŮűïijŇèĂŇèů§GILæŕńæŮăäĚş
æIJĂåŘŐä¡ăçIJ§çŽĎéIJĂèęĄåĚĹåŐżæŘđæĞĆä¡ăçŽĎäžčçăĄæŸŕåŘęçIJ§çŽĎèćńGILå¡śåŞ■åĹřãĂĆ
åŘŇæŮűè£ŸèęĄæŸŐçŹ¡GILåď'ğéČĺåĹĘéČ¡åžŤèŕřěåŘłåĚşæşÍCPUçŽĎåď'ĎçŘĘèĂŇäÿ■æŸŕI/O.

åęĆæđIJä¡ăåĞĘåď'Ğä¡£çŤĺäÿĂäÿłåď'ĎçŘĘåŹÍæśăïijŇæşÍæĎŘçŽĎæŸŕè£Źæăůå ĄŽæűĽ'åŘŁåĹřæŢřæ■ć
èćńæĽ'ğèąŇçŽĎæŞ■ä¡IJéIJĂèęĄæŤ¿åIJĺäÿĂäÿłéĂŽè£Ğdefèŕ■åŘěåőŽäźĽçŽĎPythonåĞ¡æŢřäÿ■ïijŇäÿ■èČ¡
åźűäÿŤåĞ¡æŢřåŔĆæŢřåŠŇè£ŤåŻđåĂijå£ĚéążèęĄåĚijåőźpickleãĂĆ
åŘŇæăůïijŇèęĄæĽ'ğèąŇçŽĎäżżåŁąéĞŘå£Ěéążèűşåď'§åď'ğäžěåijèèąéécİåď'ŰçŽĎéĂŽä£ąåijÄéŤĂãĂĆ

åŔęåď'ŰäÿĂäÿłéŽ¿çĆźæŸŕå¡Şæůůåŕĺä¡£çŤĺçž£çÍŃåŠŇè£ŻçÍŃæśăçŽĎæŮűåĂŹäijŽèőľä¡ăå¿Łåď't'çŰi
åęĆæđIJä¡ăèęĄåŘŇæŮűä¡£çŤĺäÿď'èĂĚïijŇæIJĂåě¡åIJĺçÍŃåžŘåŘŕåŁĺæŮűïijŇåĹŻåżžäżżä¡Ţçž£çÍŃäźŃåĽ■å
çĎűåŘŐçž£çÍŃä¡£çŤĺåŘŇæăůçŽĎè£ŻçÍŃæśæİěè£ŻèąŇåőČäžńçŽĎèőąçőŮåŕĘéŻĘåđŃåůëä¡IJãĂĆ

CæĽ'ľåśŢæIJĂéĞ■èęĄçŽĎçĽźå¿ĄæŸŕåőČäžňåŠŇPythonèğčéĞŁåŹĺæŸŕä£ĺæŇĄçŃňçńŃçŽĎãĂĆ
äź§åŕśæŸŕèŕťïijŇåęĆæđIJä¡ăåĞĘåď'ĞářĘPythonäÿ■çŽĎäżżåŁąåĹĘéĚ■åĹřCäÿ■åŐżæĽ'ğèąŇïijŇ
ä¡ăéIJĂèęĄçąőä£ĺCäžčçăĄçŽĎæŞ■ä¡IJèů§Pythonä£ĺæŇĄçŃňçńŃïijŇ
è£ŹåŕśæĎŘåŚşçİĂäÿ■èęĄä¡£çŤĺPythonæŢřæ■őçžŞæđĎäžěåŔŁäÿ■èęĄèŕČçŤĺPythonçŽĎC
APIãĂĆ åŔęåď'ŰäÿĂäÿłåŕśæŸŕä¡ăèęĄçąőä£ĺCæĽ'ľåśŢæĽ'ĂåĄŽçŽĎåůěä¡IJæŸŕèűşåď'§çŽĎïijŇåĂijå¿Ůä¡ăè
äź§åŕśæŸŕèŕťCæĽ'ľåśŢæŃĚèť'§ètůäžĘåď'ğéĞŘçŽĎèőąçőŮäżżåŁąïijŇèĂŇäÿ■æŸŕåŕŚæŢřåĞääÿłèőąçőŮãĂĆ

è£ŹäžŻèğčåĘşGILçŽĎæŰźæąĹåźűäÿ■èČ¡éĂĆçŤĺäžŐæĽĂæIJL’éŮőécŸãĂĆ
ä¿ŇåęĆïijŇæ§ŘäžŻçśżåđŃçŽĎåžŤçŤĺçĺŃåžŔåęĆæđIJèćńåĹĘèğčäÿžåđ’Žäÿłè£ŻçĺŃåđ’ĎçŘĘçŽĎèŕĺåźűäÿ■è
äź§äÿ■èČ¡åřĘåőČçŽĎéČĺåĹĘäžčçăĄæŤźæĹŘCèŕ■èĺĂæĽ’ğèąŇãĂĆ
åŕźäžŐè£ŹäžŻåžŤçŤĺçĺŃåžŔïijŇä¡ăářśèęĄèĞłåůśéIJĂæśĆèğčåĘşæŰźæąĹäžĘ
ïijĹæŕŤåęĆåđ’Žè£ŻçĺŃèő£éŮőåĚśäžńåĘĚå■ŸåŇžïijŇåđ’Žèğčæđŕåźĺè£ŘèąŇäžŐåŘŇäÿĂäÿłè£ŻçĺŃç■L’ïijL
æĹŰèĂĚïijŇä¡ăè£ŸåŔŕäžěèĂČèŹŚäÿŃåĚűäžŰçŽĎèğčéĞŁåŹĺåőđçŐřïijŇæŕŤåęĆPyPyãĂĆ

äžĘèğčæŻťåđ’ŽåĚşäžŐåIJĺCæL’l’åśŢäÿ■éĞŁæŤ¿GILïijŇèŕůåŔĆèĂČ15.7åŠŇ15.10åřŔèŁĆãĂĆ

12.10 åőŽäźĽäÿĂäÿłActoräżżåŁą

éŮőécŸ

ä¡ăæČşåőŽäźL’èů§actoræĺąåijŔäÿ■çśżäijijâĂIJactorsâĂİèğŠèL’šçŽĎäżżåŁą

èğčåĘşæŰźæąĹ

actoræĺąåijŔæŸŕäÿĂçğ■æIJĂåŔđ’èĂĄçŽĎäź§æŸŕæIJĂçőĂå■ŢçŽĎåźűèąŇåŠŇåĹĘåÿČåijŔèőąçőŮèğč
äžŇåőđäÿŁïijŇåőČåđ’l’çŤ§çŽĎçőĂå■ŢæĂğæŸŕåőČåęĆæ■đ’åŔŮæňćè£ŐçŽĎéĞ■èęĄåŐ§åŻääźŃäÿĂãĂĆ
çőĂå■ŢæĺèèőšïijŇäÿĂäÿłactoråřśæŸŕäÿĂäÿłåźűåŔŚæĽ’ğèąŇçŽĎäżżåŁąïijŇåŔłæŸŕçőĂå■ŢçŽĎæĽ’ğèąŇåŔ
åŞ■åžŤè£ŹäžŻæűĹæĄŕæŮűïijŇåőČåŔŕèČ¡è£ŸäijŽçżŹåĚűäżŰactoråŔŚéĂĄæŻť’è£ŻäÿĂæ■ěçŽĎæűĹæĄŕãĂ
actoräźŃéŮť’çŽĎéĂŽä£ąæŸŕå■ŢåŘŚåŠŇåijĆæ■ěçŽĎãĂĆåŻăæ■đ’ïijŇæűĹæĄŕåŘŚéĂĄèĂĚäÿ■ç§ěéĄŞæűĹ
äź§äÿ■äijŽæŐěæŤűåĹřäÿĂäÿłæűĹæĄŕåůšèćńåđ’ĎçŘĘçŽĎåŻđåžŤæĹŰéĂŽç§ěãĂĆ

çżŞåŘĹä¡£çŤĺäÿĂäÿłçž£çĺŃåŠŇäÿĂäÿłéŸ§åĹŮåŔŕäžěå¿ĹåőźæŸ§çŽĎåőŽäźL’actorïijŇä¿ŃåęĆïijŽ

```
from queue import Queue
from threading import Thread, Event

# Sentinel used for shutdown
class ActorExit(Exception):
    pass

class Actor:
    def __init__(self):
        self._mailbox = Queue()

    def send(self, msg):
        '''
        Send a message to the actor
        '''
        self._mailbox.put(msg)

    def recv(self):
        '''
        Receive an incoming message
        '''
        msg = self._mailbox.get()
```

```
        if msg is ActorExit:
            raise ActorExit()
        return msg

    def close(self):
        '''
        Close the actor, thus shutting it down
        '''
        self.send(ActorExit)

    def start(self):
        '''
        Start concurrent execution
        '''
        self._terminated = Event()
        t = Thread(target=self._bootstrap)

        t.daemon = True
        t.start()

    def _bootstrap(self):
        try:
            self.run()
        except ActorExit:
            pass
        finally:
            self._terminated.set()

    def join(self):
        self._terminated.wait()

    def run(self):
        '''
        Run method to be implemented by the user
        '''
        while True:
            msg = self.recv()

# Sample ActorTask
class PrintActor(Actor):
    def run(self):
        while True:
            msg = self.recv()
            print('Got:', msg)

# Sample use
p = PrintActor()
p.start()
p.send('Hello')
p.send('World')
```

```
p.close()
p.join()
```

è£Źäÿłä¿Ńå■Řäÿ■ïijŇä¡ää¡£çŤÍactoråðđä¿ŃçŽĎ send() æŰźæşŢåŘŠéĂĄæűĹæĄŕçżŹåőČäžñãĂĆ åĚűæIJžåĹűæŸŕïijŇè£ŹäÿłæŰźæşŢäijŽåřĘæűĹæĄŕæŤ¿åĚëäÿĂäÿçĎűåŘŐåřĘåĚűè¡ňäžď˙çżŹåď˙ĎçŘĘèćńæŐëåŘŮæűĹæĄŕçŽĎäÿĂäÿłåĘĚéČĺçž£çĺŃãĂĆ close() æŰźæşŢéĂŽè£ĞåIJĺéŸ§åĹŮäÿ■æŤ¿åĚëäÿĂäÿłçL'źæőŁçŽĎåŞĺåĚţåĂijïijĹActorExitïijL'æİëåĚşéŰçŤĺæĹůåŘŕäžëéĂŽè£Ğçżğæl'£ActoråźűåőŽäźL'åőđçŐřèĞłåůšåď˙ĎçŘĘéĂżè¿Śrun()æŰźæşŢæİëåőŽäźL'æŰřŕ ActorExit åijĆåÿÿçŽĎä¡£çŤĺåřśæŸŕçŤĺæĹůèĞłåőŽäźL'äžčçăĄåŘŕäžëåIJĺéIJĂèęĄçŽĎæŮűåĂŹæİëæ■ŢèĆïijĹåijĆåÿÿèćńget()æŰźæşŢæŁŻåĞžåźűäijäæŠ■åĞžåŐżïijL'ãĂĆ

åęĆædIJä¡ăæŤ¿åő¡åŕźäžŐåŘŇæ■ěåŠŇåijĆæ■ěæűĹæĄŕåŘŚéĂĄçŽĎèęĄæśĆïijŇ çśżactorårźèśąè£ŸåŘŕäžëéĂŽè£ĞçŤ§æĹŘåŹĺæİëçőĂåŇŰåőŽäźL'ãĂĆä¿ŃåęĆïijŽ

```
def print_actor():
    while True:

        try:
            msg = yield       # Get a message
            print('Got:', msg)
        except GeneratorExit:
            print('Actor terminating')

# Sample use
p = print_actor()
next(p)      # Advance to the yield (ready to receive)
p.send('Hello')
p.send('World')
p.close()
```

èőÍèőž

actoræĺąåijŔçŽĎé■ĚåŁŻåřśåIJĺäžŐåőČçŽĎçőĂå■ŢæĂğãĂĆ åőđéŹĚäÿŁïijŇè£ŹéĞŇäžĚäžĚåŘłæIJL'äÿĂäÿłæăÿå£ČæŞ■ä¡IJ send() . çŤŽèĞşïijŇåŕźäžŐåIJĺå§žäžŐactorçşżçż§äÿ■çŽĎâĂIJæűĹæĄŕâĂİçŽĎæşŻåŇŰæęĆå£ţåŘŕäžëåůšåď˙Žçğ■æŰä¿ŃåęĆïijŇä¡ääŘŕäžëäžëåĚČçżĎå¡ćåijŔäijäéĂŠæăĞç■¿æűĹæĄŕïijŇèől'actoræL'ğèąŇäÿ■åŘŇçŽĎæŞ■ä¡IJïij

```
class TaggedActor(Actor):
    def run(self):
        while True:
            tag, *payload = self.recv()
            getattr(self,'do_'+tag)(*payload)

    # Methods correponding to different message tags
    def do_A(self, x):
        print('Running A', x)

    def do_B(self, x, y):
        print('Running B', x, y)
```

```
# Example
a = TaggedActor()
a.start()
a.send(('A', 1))       # Invokes do_A(1)
a.send(('B', 2, 3))    # Invokes do_B(2,3)
```

ä¡IJäÿžåŔęåď'ŰäÿĂäÿłä¿Ńå■ŘïijŇäÿŃéİćçŽĎactoråĚĄèőÿåIJĺäÿĂäÿłåůěä¡IJèĂĚäÿ■è£ŘèąŇäżżæĎŔçŽåźűäÿŤéĂŽè£ĞäÿĂäÿłçĽźæőŁçŽĎResultåŕźèśąè£ŤåŻđçżŞæđIJïijŽ

```
from threading import Event
class Result:
    def __init__(self):
        self._evt = Event()
        self._result = None

    def set_result(self, value):
        self._result = value

        self._evt.set()

    def result(self):
        self._evt.wait()
        return self._result

class Worker(Actor):
    def submit(self, func, *args, **kwargs):
        r = Result()
        self.send((func, args, kwargs, r))
        return r

    def run(self):
        while True:
            func, args, kwargs, r = self.recv()
            r.set_result(func(*args, **kwargs))

# Example use
worker = Worker()
worker.start()
r = worker.submit(pow, 2, 3)
print(r.result())
```

æIJĂåŘŐïijŇâĂIJåŔŚéĂĄâĂİäÿĂäÿłäżżåŁąæűĹæĄŕçŽĎæęĆå£ţåŔŕäżèèćńæĽľåśŢåĹřåď'Žè£ŻçÍŃçŤŽäż¿ŃåęĆïijŇäÿĂäÿłçśżactoråŕźèśąçŽĎ `send()` æŰźæşŢåŔŕäżèèćńçijŰçÍŃèőľåőČèČ¡åIJĺäÿĂäÿłåëŮæŐëå■ŮæĹŰéĂŽè£Ğæ§ŘäžŻæűĹæĄŕäÿ■éŮť'äżűïijĹæŕŤåęĆAMQPãĂĄZMQç■Ľ'ïijĽ'æİëåŔŚéĂĄãĂĆ

12.11 åőđçŐřæűĹæĄŕåŔŚåÿČ/èőćéŸĚæĺąåđŃ

éŮőécŸ

ä¡ăæIJL’äÿĂäÿłå§žäžŐçž£çÍŃéĂŽä£ąçŽĎçÍŃåžŕïijŇæČşèőľ’åőČäžňåőđçŐřåŔŚåÿČ/èőćéŸĚæÍąåijŔçŽÌ

èğčåĘşæŰźæąĹ

èęĄåőđçŐřåŔŚåÿČ/èőćéŸĚçŽĎæűĹæĄŕéĂŽä£ąæÍąåijŔïijŇ
ä¡ăéĂŽåÿÿèęĄåijŢåĚěäÿĂäÿłå■ŢçŃňçŽĎâĂIJäžď’æ■ćæIJžâĂİæĹŰâĂIJç¡ŚåĚşâĂİåŕźèśąä¡IJäÿžæL’ĂæIJL’æ
äź§åřśæŸŕèŕť’ïijŇäÿ■çŻť’æŐěåřĘæűĹæĄŕäżŐäÿĂäÿłäżżåŁąåŔŚéĂĄåĹřåŔęäÿĂäÿłïijŇèĂŇæŸŕåřĘåĚűåŔŚé
çĎűåŘŐçŤśäžď’æ■ćæIJžåřĘåőČåŔŚéĂĄçżŹäÿĂäÿłæĹŰåď’Žäÿłèćńåĕşè ĄŤäżżåŁąãĂĆäÿŃéİćæŸŕäÿĂäÿłéİđ

```
from collections import defaultdict

class Exchange:
    def __init__(self):
        self._subscribers = set()

    def attach(self, task):
        self._subscribers.add(task)

    def detach(self, task):
        self._subscribers.remove(task)

    def send(self, msg):
        for subscriber in self._subscribers:
            subscriber.send(msg)

# Dictionary of all created exchanges
_exchanges = defaultdict(Exchange)

# Return the Exchange instance associated with a given name
def get_exchange(name):
    return _exchanges[name]
```

äÿĂäÿłäžď’æ■ćæIJžåřśæŸŕäÿĂäÿłæŹőéĂŽåŕźèśąïijŇèt’§èt’čçżt’æŁď’äÿĂäÿłæt’żèůČçŽĎèőćéŸĚèĂĚéŻĺ
æŕŔäÿłäžď’æ■ćæIJžéĂŽè£ĞäÿĂäÿłåŘ■çğřåőŽä¡■ïijŇget_exchange()
éĂŽè£ĞçżŹåőŽäÿĂäÿłåŘ■çğřè£ŤåŻđçŻÿåžŢçŽĎ Exchange åőđä¿ŃãĂĆ

äÿŃéİćæŸŕäÿĂäÿłçőĂå■Ţä¿Ńå■ŘïijŇæijŢçď’žäžĘåęĆä¡Ţä¡£çŤĺäÿĂäÿłäžď’æ■ćæIJžïijŽ

```
# Example of a task.  Any object with a send() method

class Task:
    ...
    def send(self, msg):
        ...

task_a = Task()
task_b = Task()
```

```
# Example of getting an exchange
exc = get_exchange('name')

# Examples of subscribing tasks to it
exc.attach(task_a)
exc.attach(task_b)

# Example of sending messages
exc.send('msg1')
exc.send('msg2')

# Example of unsubscribing
exc.detach(task_a)
exc.detach(task_b)
```

åř¡çőąåŕźäžŐè£ŹäýłéŮőécŸæIJL'å¿Ĺåd'ŽçŽĎåŔŸçğ■ïijŇäÿ■è£ĞäÿĞåŔŸäÿ■çężåĚűåőŮãĂĆ
æűĹæĄŕäijŽècńåŔŚéĂĄçżŹäÿĂäýłäžd'æ■ćæIJžïijŇçĎűåŘŐäžd'æ■ćæIJžäijŽářĘåőČäžňåŔŚéĂĄçżŹècńçżŚã

èőĺèőž

éĂŽè£ĞéŸ§åĹŮåŔŚéĂĄæűĹæĄŕçŽĎäżżåŁąæĹŰçž£çĺŇçŽĎæĺąåijŔå¿Ĺåőźæ ŸŞècńåőđçŐřåźűäÿŤäź§
äÿ■è£ĞïijŇä¡£çŤĺåŔŚåÿČ/èőćéŸĚæĺąåijŔçŽĎåě¡åd'ĎæŻť'åŁäæŸŐæŸ¿ãĂĆ

éęŰåĚĹïijŇä¡£çŤĺäÿĂäýłäžd'æ■ćæIJžåŔŕäżěçőĂåŇŰåd'ğéČĺåĹĘæűL'åŔŁåĹřçž£çĺŇéĂŽä£ąçŽĎåůěä¡I
æŮăéIJĂåŐżåĘŹéĂŽè£Ğåd'Žè£ŽçĺŇæĺąåİŮæİěæŞ■ä¡IJåd'Žäÿłçž£çĺŇïijŇä¡ääŔłéIJĂèęĄä¡£çŤĺè£Źäÿłäžd'æ
æ§Řçğ■çĺŇåžęäÿŁïijŇè£Źäÿłåŕśèů§æŮěå£ŮæĺąåİŮçŽĎåůěä¡IJåŐ§çŘĘçśżäijijãĂĆ
åőđéŹĚäÿŁïijŇåőČåŔŕäżěè¡żæİ¿çŽĎèğčèĂęçĺŇåžŔäÿ■åd'ŽäÿłäżżåąãĂĆ

åĚűæňąïijŇäžd'æ■ćæIJžåź£æŠ■æűĹæĄŕçżŹåd'ŽäÿłèőćéŸĚèĂĚçŽĎèČ¡åŁŽåÿęæİěäžĘäÿĂäýłåĚĺæŰřçŽ
ä¿ŇåęĆïijŇä¡ääŔŕäżěä¡£çŤĺåd'Žäżżåąçşżçż§ãĂĄåź£æŠ■æĹŰæL'ĞåĞžãĂĆ
ä¡ăè£ŸåŔŕäżěéĂŽè£ĞäżěæŹőéĂŽèőćéŸĚèĂĚèžńäż¡çżŚåőŽæİěædĎåżżèřČèŕŢåŠŇèŕŁæŰ■åůěåĚůãĂĆ
ä¿ŇåęĆïijŇäÿŇéİćæŸŕäÿĂäýłçőĂå■ŢçŽĎèŕŁæŰ■çśžïijŇåŔŕäżěæŸ¿çd'żècńåŔŚéĂĄçŽĎæűĹæĄŕïijŽ

```
class DisplayMessages:
    def __init__(self):
        self.count = 0
    def send(self, msg):
        self.count += 1
        print('msg[{}]: {!r}'.format(self.count, msg))

exc = get_exchange('name')
d = DisplayMessages()
exc.attach(d)
```

æIJĂåŘŐïijŇèŕěåőđçŐřçŽĎäÿĂäýłéĞ■èęĄçL'źçĆźæŸŕåőČèČ¡åĚijåőźåd'Žäýłâ ĂIJtask-
likeâĂİåŕźèśąãĂĆ ä¿ŇåęĆïijŇæűĹæĄŕæŐěåŘŮèĂĚåŔŕäżěæŸŕactorïijĹ12.10åřŔèŁĆäżŃçż■ïijL'ãĂĄå■ŘçĺŇ
`send()` æŰźæşŢçŽĎäÿIJèě£ãĂĆ

åĚşäžŐäžd'æ■ćæIJžçŽĎäÿĂäýłåŔŕèČ¡éŮőécŸæŸŕåŕźäžŐèőćéŸĚèĂĚçŽĎæ■čçąőçżŚåőŽåŠŇèğčçżŚãĂ
äÿžäžĘæ■čçąőçŽĎçőąçŘĘèţĎæžŘïijŇæŕŘäÿĂäýłçżŚåőŽçŽĎèőćéŸĚèĂĚå£ĚéążæIJĂçżĹèęĄèğčçżŚãĂĆ

åIJÍäžčçăĄäÿ■éĂŽåÿÿäijŽæŸŕåČŔäÿŃéİćè£Źæăůç ŽĎæÍąåijŔïijŽ

```
exc = get_exchange('name')
exc.attach(some_task)
try:
    ...
finally:
    exc.detach(some_task)
```

æ§Řçğ■æĎŔäźĽäÿŁïijŇè£ŹäÿłåŠŇä¡£çŤÍæŰĞäžűãĂĄéŤĄåŠŇçśżäijijåŕźèśąå¿ĹåČŔãĂĆ éĂŽåÿÿå¿Ĺåőźæ ŸŞäijŽå£Ÿèőřæ IJĂåŘŐçŽĎ `detach()` æ■ééłď ãĂĆ äÿžäžĘçőĂåŇŰè£ŹäÿłïijŇä¡ăåŔŕäžěèĂČèŹŠä¡£çŤÍäÿŁäÿŃæŰĞçőąçŘĘåŹĺå■Ŕèőőã ĂĆ ä¿ŃåęĆïijŇåIJÍäžď æ■ćæIJžåŕźèśąäÿŁåćđåŁääÿĂäÿł `subscribe()` æŰźæşŢïijŇåęĆäÿŃïijŽ

```
from contextlib import contextmanager
from collections import defaultdict

class Exchange:
    def __init__(self):
        self._subscribers = set()

    def attach(self, task):
        self._subscribers.add(task)

    def detach(self, task):
        self._subscribers.remove(task)

    @contextmanager
    def subscribe(self, *tasks):
        for task in tasks:
            self.attach(task)
        try:
            yield
        finally:
            for task in tasks:
                self.detach(task)

    def send(self, msg):
        for subscriber in self._subscribers:
            subscriber.send(msg)

# Dictionary of all created exchanges
_exchanges = defaultdict(Exchange)

# Return the Exchange instance associated with a given name
def get_exchange(name):
    return _exchanges[name]

# Example of using the subscribe() method
exc = get_exchange('name')
```

```
with exc.subscribe(task_a, task_b):
    ...
    exc.send('msg1')
    exc.send('msg2')
    ...

# task_a and task_b detached here
```

æIJĂåŘŐè£ŸåžŧèŕěæşĺæĎŔçŽĎæŸŕåĔşäžŐäžďæ■ćæIJžçŽĎæĂİæČşæIJL'å¿Ĺåď'ŽçğçŽĎæL'l'åśŢåőđ
ä¿ŇåęĆïijŇäžď'æ■ćæIJžåŔŕäžěåőđçŐřäÿĂæŢt'äÿłæűĹæĄŕéĂŽéĄŞéŻĘåŘĹæĹŰæŔŘä¿Żäžď'æ■ćæIJžåŘçğ
äžď'æ■ćæIJžè£ŸåŕäžěèćńæL'l'åśŢåĹřåĹĘåÿČåijŔèőąçőŮçĺŃåžŔäÿ■ïijĹærŧåęĆïijŇářĘæűĹæĄrèůrçŧśåĹřä

12.12 ä¡£çŤĺçŤ§æĹŘåŹĺäżčæŻ£çž£çĺŃ

éŮőécŸ

ä¡ăæČşä¡£çŤĺçŤ§æĹŘåŹĺïijĹå■ŔçĺŃïijL'æŻ£äžčçşżçż§çž£çĺŃæİěåőđçŐřåźűåŔŠãĂĆè£Źäÿłæ IJL'æŮűåŀ

èğčåĘşæŰźæąĹ

èęĄä¡£çŤĺçŤ§æĹŘåŹĺåőđçŐřèĞłåůśçŽĎåźűåŔŠïijŇä¡ăéęŰåĚĹèęĄåŕźçŤ§æĹŘåŹĺåĞ¡æŢřåŠŇ
`yield`èŕ■åŔěæIJL'æůśåĹżçŘĘèğčãĂĆ `yield`èŕ■åŔëäijŽèőľäÿĂäÿłçŤ§æĹŘåŹĺæŇĆèţůåőČçŽĎæL'ğèąŇïi
åřĘçŤ§æĹŘåŹĺå¡ŞåĄŽæ§Řçğ■âĂIJäżżåŁąâĂİåźűä¡£çŤĺäżżåŁąå■Ŕä¡IJåĹĞæ■ćæİěæŻ£æ■ćåőČäžñçŽĎæL'ğ
èęĄæijŤçď'žè£Źçğ■æĂİæČşïijŇèĂČèŹŚäÿŃéİćäÿď'äÿłä¡£çŤĺçőĂå■ŢçŽĎ `yield`
èŕ■åŔěçŽĎçŤ§æĹŘåŹĺåĞ¡æŢřïijŽ

```
# Two simple generator functions
def countdown(n):
    while n > 0:
        print('T-minus', n)
        yield
        n -= 1
    print('Blastoff!')

def countup(n):
    x = 0
    while x < n:
        print('Counting up', x)
        yield
        x += 1
```

è£ŹäžŻåĞ¡æŢřåIJĺåĘéĆĺä¡£çŤĺyieldèŕ■åŔëijŇäÿŃéİćæŸŕäÿĂäÿłåőđçŐřäžEçőĂå■ŢäżżåŀąèřĆåžęåŹĺ

```
from collections import deque

class TaskScheduler:
    def __init__(self):
```

```
        self._task_queue = deque()

    def new_task(self, task):
        '''
        Admit a newly started task to the scheduler

        '''
        self._task_queue.append(task)

    def run(self):
        '''
        Run until there are no more tasks
        '''
        while self._task_queue:
            task = self._task_queue.popleft()
            try:
                # Run until the next yield statement
                next(task)
                self._task_queue.append(task)
            except StopIteration:
                # Generator is no longer executing
                pass

# Example use
sched = TaskScheduler()
sched.new_task(countdown(10))
sched.new_task(countdown(5))
sched.new_task(countup(15))
sched.run()
```

TaskScheduler çśżåIJÍäÿĂäÿłå¿łçŐŕäÿ■è£ŘèąŇçŢ§æĹŘåŹĺéŻĘåŘĹâĂŤâĂŤæŕŔäÿłéČ¡è£ŘèąŇåĹřç
è£ŘèąŇè£ŹäÿłäżŃå■ŘïijŇè¿ŞåĞžåęĆäÿŃïijŽ

```
T-minus 10
T-minus 5
Counting up 0
T-minus 9
T-minus 4
Counting up 1
T-minus 8
T-minus 3
Counting up 2
T-minus 7
T-minus 2
...
```

åĹřæ■d'äÿžæ■ćïijŇæĹŚäżňåőđéŹĚäÿŁåůšçżŘåőđçŐřäžĘäÿĂäÿłâĂIJæŞ■ä¡IJçşżçż§âĂİçŽĎæIJĂåřŔæă
çŢ§æĹŘåŹĺåĞ¡æŢřåřśæŸŕèőd'äÿžïijŇèĂŇyieldèŕ■åŘěæŸŕäżżåŁąæŇĆèţůçŽĎä£ąåŘůãĂĆ
èřČåžęåŹĺå¿łçŐŕæčĂæ§ěäżżåŁąåĹŮèąĺçŻt'åĹřæšąæIJL'äżżåŁąèęĄæL'ğèąŇäÿžæ■ćãĂĆ

åőđéŹĚäÿŁïijŇä¡ăåŔŕèČ¡æČşèęĄä¡£çŤĺçŢ§æĹŘåŹĺæİěåőđçŐřçőĂå■ŢçŽĎåźűåŔŚãĂĆ

éĆčäźĹïijŇåIJĺåőđçŐřactoræĹŰç¡ŚçżIJæIJ■åŁąåŹĺçŽĎæŮűåĂŹä¡ăåŔŕäžěä¡£çŤĺçŢ§æĹŘåŹĺæİěæŻ£äžčçž£ç

äÿŃéİćçŽĎäžčçăĄæijŢçď žäžĘä¡£çŤĺçŢ§æĹŘåŹĺæİěåőđçŐřäÿĂäÿłäÿ■ä¿İèţŰçž£çĺŃçŽĎactorïijŽ

```
from collections import deque

class ActorScheduler:
    def __init__(self):
        self._actors = { }          # Mapping of names to actors
        self._msg_queue = deque()   # Message queue

    def new_actor(self, name, actor):
        '''
        Admit a newly started actor to the scheduler and give it a␣
↪name
        '''
        self._msg_queue.append((actor,None))
        self._actors[name] = actor

    def send(self, name, msg):
        '''
        Send a message to a named actor
        '''
        actor = self._actors.get(name)
        if actor:
            self._msg_queue.append((actor,msg))

    def run(self):
        '''
        Run as long as there are pending messages.
        '''
        while self._msg_queue:
            actor, msg = self._msg_queue.popleft()
            try:
                 actor.send(msg)
            except StopIteration:
                 pass

# Example use
if __name__ == '__main__':
    def printer():
        while True:
            msg = yield
            print('Got:', msg)

    def counter(sched):
        while True:
            # Receive the current count
            n = yield
            if n == 0:
                break
            # Send to the printer task
```

```
            sched.send('printer', n)
            # Send the next count to the counter task (recursive)

            sched.send('counter', n-1)

    sched = ActorScheduler()
    # Create the initial actors
    sched.new_actor('printer', printer())
    sched.new_actor('counter', counter(sched))

    # Send an initial message to the counter to initiate
    sched.send('counter', 10000)
    sched.run()
```

åőŇåĚĺåijĎæĞĆè£Źæőţäžčçă Ąé IJĂèęĄæŻť'æůśåĚěçŽĎå■ęäźăïijŇä¡ĘæŸŕåĚşéŤőçĆźåIJĺäžŐæŤűéŻĘæ
æIJňèť'ĺäÿŁïijŇèřČåžęåŹĺåIJĺæIJL'éIJĂèęĄåŔŠéĂĄçŽĎæűĹæĄŕæŮűäijŽäÿĂçŻť'è£ŘèąŇçİĂãĂĆ
èőąæŢřçŤ§æĹŘåŹĺäijŽçżŹèĞłåůśåŔŠéĂĄæűĹæĄŕåźűåIJĺäÿĂäÿłéĂŠå¡Šå¿łçŐŕäÿ■çżŞæİ§ãĂĆ

äÿŃéİćæŸŕäÿĂäÿłæŻť'åŁăéńŸçžğçŽĎä¿Ńå■ŘïijŇæijŤçď'žäžĘä¡£çŤĺçŤ§æĹŘåŹĺæİěåőđçŐŕäÿĂäÿłåźűå

```
from collections import deque
from select import select

# This class represents a generic yield event in the scheduler
class YieldEvent:
    def handle_yield(self, sched, task):
        pass
    def handle_resume(self, sched, task):
        pass

# Task Scheduler
class Scheduler:
    def __init__(self):
        self._numtasks = 0        # Total num of tasks
        self._ready = deque()     # Tasks ready to run
        self._read_waiting = {}   # Tasks waiting to read
        self._write_waiting = {} # Tasks waiting to write

    # Poll for I/O events and restart waiting tasks
    def _iopoll(self):
        rset,wset,eset = select(self._read_waiting,
                                self._write_waiting,[])
        for r in rset:
            evt, task = self._read_waiting.pop(r)
            evt.handle_resume(self, task)
        for w in wset:
            evt, task = self._write_waiting.pop(w)
            evt.handle_resume(self, task)

    def new(self,task):
        '''
```

```
        Add a newly started task to the scheduler
        '''

        self._ready.append((task, None))
        self._numtasks += 1

    def add_ready(self, task, msg=None):
        '''
        Append an already started task to the ready queue.
        msg is what to send into the task when it resumes.
        '''
        self._ready.append((task, msg))

    # Add a task to the reading set
    def _read_wait(self, fileno, evt, task):
        self._read_waiting[fileno] = (evt, task)

    # Add a task to the write set
    def _write_wait(self, fileno, evt, task):
        self._write_waiting[fileno] = (evt, task)

    def run(self):
        '''
        Run the task scheduler until there are no tasks
        '''
        while self._numtasks:
            if not self._ready:
                self._iopoll()
            task, msg = self._ready.popleft()
            try:
                # Run the coroutine to the next yield
                r = task.send(msg)
                if isinstance(r, YieldEvent):
                    r.handle_yield(self, task)
                else:
                    raise RuntimeError('unrecognized yield event')
            except StopIteration:
                self._numtasks -= 1

# Example implementation of coroutine-based socket I/O
class ReadSocket(YieldEvent):
    def __init__(self, sock, nbytes):
        self.sock = sock
        self.nbytes = nbytes
    def handle_yield(self, sched, task):
        sched._read_wait(self.sock.fileno(), self, task)
    def handle_resume(self, sched, task):
        data = self.sock.recv(self.nbytes)
        sched.add_ready(task, data)
```

```
class WriteSocket(YieldEvent):
    def __init__(self, sock, data):
        self.sock = sock
        self.data = data
    def handle_yield(self, sched, task):

        sched._write_wait(self.sock.fileno(), self, task)
    def handle_resume(self, sched, task):
        nsent = self.sock.send(self.data)
        sched.add_ready(task, nsent)

class AcceptSocket(YieldEvent):
    def __init__(self, sock):
        self.sock = sock
    def handle_yield(self, sched, task):
        sched._read_wait(self.sock.fileno(), self, task)
    def handle_resume(self, sched, task):
        r = self.sock.accept()
        sched.add_ready(task, r)

# Wrapper around a socket object for use with yield
class Socket(object):
    def __init__(self, sock):
        self._sock = sock
    def recv(self, maxbytes):
        return ReadSocket(self._sock, maxbytes)
    def send(self, data):
        return WriteSocket(self._sock, data)
    def accept(self):
        return AcceptSocket(self._sock)
    def __getattr__(self, name):
        return getattr(self._sock, name)

if __name__ == '__main__':
    from socket import socket, AF_INET, SOCK_STREAM
    import time

    # Example of a function involving generators.  This should
    # be called using line = yield from readline(sock)
    def readline(sock):
        chars = []
        while True:
            c = yield sock.recv(1)
            if not c:
                break
            chars.append(c)
            if c == b'\n':
                break
        return b''.join(chars)
```

```
    # Echo server using generators
    class EchoServer:
        def __init__(self,addr,sched):
            self.sched = sched
            sched.new(self.server_loop(addr))

        def server_loop(self,addr):
            s = Socket(socket(AF_INET,SOCK_STREAM))

            s.bind(addr)
            s.listen(5)
            while True:
                c,a = yield s.accept()
                print('Got connection from ', a)
                self.sched.new(self.client_handler(Socket(c)))

        def client_handler(self,client):
            while True:
                line = yield from readline(client)
                if not line:
                    break
                line = b'GOT:' + line
                while line:
                    nsent = yield client.send(line)
                    line = line[nsent:]
            client.close()
            print('Client closed')

    sched = Scheduler()
    EchoServer(('',16000),sched)
    sched.run()
```

è£ŹæőţäžčçăæIJL’çĆźåď■æİĆãĂĆäÿ■è£ĞïijŇåőČåőđçŐřäžĘäÿĂäÿłářŔåđŃçŽĎæŞ■ä¡IJçşžçż§ãĂĆ æIJL’äÿĂäÿłářšçżłçŽĎäżżåŁąéŸ§åĹŮïijŇåźűäÿŤè£ŸæIJL’åŻăI/OäijŚçIJăçŽĎäżżåŁąç■L’å¿ĚåŇžå§§ãĂĆ è£ŸæIJL’å¿Ĺåď’ŽèřČåžęåŹĺèť§èť’čåIJĺářšçżłéŸ§åĹŮåŠŇI/Oç■L’å¿ĚåŇžå§§äźŃéŮť’çğżåŁĺäżżåŁąãĂĆ

èőĺèőž

åIJĺæđĎåżžå§žäžŐçŤ§æĹŘåŹĺçŽĎåźűåŔŚæąĘæđűæŮűïijŇéĂŽåÿÿäijŽä¡£çŤĺæŻť’åÿÿèğĄçŽĎyieldå¡ćå

```
def some_generator():
    ...
    result = yield data
    ...
```

ä¡£çŤĺè£Źçğ■å¡ćåijŔçŽĎyieldèŕ■åŔěçŽĎåĞ¡æŢřéĂŽåÿÿèćńçğřäÿžâĂIJå■ŔçĺŃâĂİãĂĆ éĂŽè£ĞèřČåžęåŹĺïijŇyieldèŕ■åŔěåIJĺäÿĂäÿłå¿łçŐŕäÿ■èćńåď’ĎçŘĘïijŇåęĆäÿŃïijŽ

```
f = some_generator()

# Initial result. Is None to start since nothing has been computed
result = None
while True:
    try:
        data = f.send(result)
        result = ... do some calculation ...
    except StopIteration:
        break
```

è£ŹéĞÑçŽĎéĂżè¿ŚçÍ■å¿őæIJL'çĆžåď'■æÌĆãĂĆäÿ■è£ĞïijÑèćńäijăçżŹ
send() çŽĎåĂijåőŽäźL'äžĘåIJÍyieldèŕ■åŔéĘŠæÌěæŮűçŽĎè£ŤåŻđåĂijãĂĆ
åŻăæ■ď'ïijÑåęĆæđIJäÿĂäÿłyieldåĞĘåď'ĞåIJÍåŕźäźÑåL'■yield-
æŢřæ■őçŽĎåŻđåžŤäÿ■è£ŤåŻđçżŞæđIJæŮűïijÑäijŽåIJÍäÿÑäÿĂæňą send()
æŞ■ä¡IJè£ŤåŻđãĂĆ åęĆæđIJäÿĂäÿłçŤ§æĹŔåŹÍåĞ¡æŢřåĹŽåijÅåğÑè£ŘèąÑïijÑåŔŚéĂĄäÿĂäÿłNoneåĂijäij

éŹď'äžĘåŔŚéĂĄåĂijåď'ŰïijÑè£ŸåŕŕäžěåIJÍäÿĂäÿłçŤ§æĹŔåŹÍäÿŁéÌćæL'ğèąÑäÿĂäÿł
close() æŰźæşŢãĂĆ åőČäijŽåŕijèĞť'åIJÍæL'ğèąÑyieldèŕ■åŔěæŮűæŁŻåĞžäÿĂäÿł
GeneratorExit åijĆåÿÿïijÑäżŐèĂÑçżĹæ■ćæL'ğèąÑãĂĆ
åęĆæđIJè£ŻäÿĂæ■ěèő¿èőąïijÑäÿĂäÿłçŤ§æĹŔåŹÍåŔŕäžěæ■ŢèŐůè£ŹäÿłåijĆåÿÿåźűæL'ğèąÑæÿĚçŘĘæŞ■ä¡I
åŘÑæăůè£ŸåŔŕäžěä¡£çŤÍçŤ§æĹŔåŹÍçŽĎ throw() æŰźæşŢåIJÍyield-
èŕ■åŔěæL'ğèąÑæŮűçŤ§æĹŘäÿĂäÿłäżżæĎŘçŽĎæL'ğèąÑæÑĞäžď'ãĂĆ
äÿĂäÿłäżżåŁąèřČåžęåŹÍåŔŕåĹ'çŤÍåőČæÌěåIJÍè£ŘèąÑçŽĎçŤ§æĹŔåŹÍäÿ■åď'ĎçŘĘéŤŹèŕŕãĂĆ

æIJĂåŘŐäÿĂäÿłä¿Ńå■Řäÿ■ä¡£çŤÍçŽĎ yield from èŕ■åŘěèćńçŤÍæÌěåőđçŐŕå■ŘçÍÑïijÑåŔŕäžěèćńåĚ
æIJňèť'ÍäÿŁåřśæŸŕåŗĘæŐğåĹűæÌČéĂŔæŸÕçŽĎäijăè¿ŞçżŹæŰřçŽĎåĞ¡æŢřãĂĆ
äÿ■åČŔæŹőéĂŽçŽĎçŤ§æĹŔåŹÍïijÑäÿĂäÿłä¡£çŤÍ yield from
èćńèřČçŤÍçŽĎåĞ¡æŢřåŔŕäžěè£ŤåŻđäÿĂäÿłä¡IJäÿž yield from
èŕ■åŘężŞæđIJçŽĎåĂijãĂĆ åĚşäžŐ yield from çŽĎæŻť'åď'Žä£ąæĄŕåŔŕäžěåIJÍ PEP
380 äÿ■æL'¿åĹřãĂĆ

æIJĂåŘŐïijÑåęĆæđIJä¡£çŤÍçŤ§æĹŔåŹÍçijŰçÍÑïijÑèęĄæŔŘéĘŠä¡ăçŽĎæŸŕåőČè£ŸæŸŕæIJL'å¿Ĺåď'Žç
çL'źåĹńæŸŕïijÑä¡ăå¿Ůäÿ■åĹřäžżä¡Ţçž£çÍÑåŔŕäžěæŔŘä¿ŻçŽĎåě¡åď'ĎãĂĆä¿ÑåęĆïijÑåęĆæđIJä¡ăæL'ğèąÑĆ
åőČäijŽårĘæŢť'äÿłäżżåŁąæÑĆèţůç§ěéĄŞæŞ■ä¡IJåőÑæĹŘãĂĆäÿžäžĘèğčåĘşè£ŹäÿłéŮőécŸïijÑ
ä¡ăåŔłèČ¡éĂL'æŃl'årĘæŞ■ä¡IJåğŤæť'¿çżŹåŖęåď'ŰäÿĂäÿłåŔŕäžěçÑňçńÑè£ŘèąÑçŽĎçž£çÍÑæĹŰè£ŻçÍÑãĂĆ
åŖęåď'ŰäÿĂäÿłéŹŘåĹűæŸŕåď'ğéČÍåĹĘPythonåžŞåźűäÿ■èČ¡å¿Ĺåě¡çŽĎåĚijåőźå§žäžŐçŤ§æĹŘåŹÍçŽĎçž£çÍ
åęĆæđIJä¡ăéĂL'æŃl'è£ŹäÿłæŰźæĹïijÑä¡ăäijŽåŘŚçŐŕä¡ăéIJĂèęĄèĞłåůśæŤźåĘŹå¿Ĺåď'ŽæăĞåĞĘåžŞåĞ¡æ
ä¡IJäÿžæIJňèŁĆæŘŘåĹřçŽĎå■ŘçÍÑåŠÑçŻÿåĚşæŁĂæIJŕçŽĎäÿĂäÿłå§žçąĂèČÑæŹřïijÑåŔŕäžěæ§ěçIJÑ
PEP 342 åŠÑ âĂIJå■ŘçÍÑåŠÑåźűåŘŚçŽĎäÿĂéŮÍæIJL'èűčèŕ¿çÍÑâĂİ

PEP 3156 åŘÑæăůæIJL'äÿĂäÿłåĚşäžŐä¡£çŤÍå■ŘçÍÑçŽĎåijĆæ■ěI/OæÍąåđÑãĂĆ
çL'źåĹńçŽĎïijÑä¡ăäÿ■åŔŕèČ¡èĞłåůśåŐżåőđçŐŕäÿĂäÿłåžŢåśĆçŽĎå■ŘçÍÑèŘČåžęåŹÍãĂĆ
äÿ■è£ĞïijÑåĚşäžŐå■ŘçÍÑçŽĎæĂİæČşæŸŕå¿Ĺåď'ŽæţĄèąÑåžŞçŽĎå§žçąĂïijÑ åÑĚæÑň
gevent, greenlet, Stackless Python äžěåŔŁåĚűäžŰçśżäijijåůěçÍÑãĂĆ

12.13 åď'Žäÿłçž£çÍÑéŸ§åĹŮè¡őèŕć

éŮőécŸ

ä¡ăæIJL’äÿĂäÿłçž£çÍNéŸ§åĹŮéŻĘåŘĹïijŇæČşäÿžåĽřæİěçŽĎåĚČçť ăè¡őèŕćåőČäžñïijŇ åřśèů§ä¡ăäÿžäÿĂäÿłåőćæĹůçńŕèřůæśĆåŐżè¡őèŕćäÿĂäÿłç¡ŚçżIJè£đæŐěéŻĘåŘĹçŽĎæŰźåijŔäÿĂæăůãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐè¡őèŕćéŮőécŸçŽĎäÿĂäÿłåÿÿèğĄèğčåĘşæŰźæąĹäÿ■æIJL’äÿłå¿ĹåřŚæIJL’äžžç§ěéĄŞçŽĎæŁĂåů æIJňèť´íäÿŁèőšåĚűæĂİæČşåřśæŸŕïijŽåŕźäžŐæŕŔäÿłä¡ăæČşèęĄè¡őèŕćçŽĎéŸ§åĹŮïijŇä¡ăåĹŻåžžäÿĂåŕźè£đa çĎűåŘŐä¡ăåIJĺåĚűäÿ■äÿĂäÿłåěŮæŐěå■ŮäÿŁéİćçijŰåĘŹäžčçăĄæİěæăĞèŕĘå■ŸåIJĺçŽĎæŢřæ■őïijŇ åŔęåď´ŰäÿĂäÿłåěŮæŐěå■ŮècńäijăçžŹ `select()` æĹŰçśžäijijçŽĎäÿĂäÿłè¡őèŕćæŢřæ■őåĹřè¿¿çŽĎåĞ¡æŢř

```
import queue
import socket
import os

class PollableQueue(queue.Queue):
    def __init__(self):
        super().__init__()
        # Create a pair of connected sockets
        if os.name == 'posix':
            self._putsocket, self._getsocket = socket.socketpair()
        else:
            # Compatibility on non-POSIX systems
            server = socket.socket(socket.AF_INET, socket.SOCK_
↪STREAM)
            server.bind(('127.0.0.1', 0))
            server.listen(1)
            self._putsocket = socket.socket(socket.AF_INET, socket.
↪SOCK_STREAM)
            self._putsocket.connect(server.getsockname())
            self._getsocket, _ = server.accept()
            server.close()

    def fileno(self):
        return self._getsocket.fileno()

    def put(self, item):
        super().put(item)
        self._putsocket.send(b'x')

    def get(self):
        self._getsocket.recv(1)
        return super().get()
```

åIJĺè£Źäÿłäžčçăąäÿ■ïijŇäÿĂäÿłæŰřçŽĎ `Queue` åőđä¿ŃçśžåđŃèćńåőŽäźL’ïijŇåžŢåśĆæŸŕäÿĂäÿłèćńè£ åIJĺUnixæIJžåŹĺäÿŁçŽĎ `socketpair()` åĞ¡æŢřèČ¡è¡żæİ¿çŽĎåĹŻåžžè£ŹæăůçŽĎåěŮæŐěå■ŮãĂĆ åIJĺWindowsäÿŁéİćïijŇä¡ăå£Ěéąžä¡£çŤĺçśžäijijäžčçăĄæİěæĺąæŃ§åőČãĂĆ çĎűåŘŐåőŽäźL’æŹőéĂŽçŽĎ `get()` åŠŇ `put()` æŰźæşŢåIJĺè£ŹäžŽåěŮæŐěå■ŮäÿŁéİćæİěæL’ğèąŇI/OæŞ

put() æŰźæşŢåE■åřĘæŢřæ■őæŤ¿åĚéÝ§åĹŮåŘŐäijŽåĘŹäÿĂäÿłå■Ţå■ŮèŁĆåĹřæ§Řäÿłåě
èĂŇ get() æŰźæşŢåIJĺäžŐéÝ§åĹŮäÿ■çğżéŹd'äÿĂäÿłåĚČçť'ăæŮűäijŽäžŐåŕęåd'ŰäÿĂäÿłåěŮæŐěå■Ůäÿ■

fileno() æŰźæşŢä¡£çŤĺäÿĂäÿłåĞ¡æŢřæŕŤåęĆ select() æİěèől'è£ŹäÿłéŸ§åĹŮåŔřäžěèćńè¡őèŕćãĂĆ åőČäżĚäżĚåŔłæŸŕæŽť'éIJšäžĘåžŢåśĆèćń get() åĞ¡æŢřä¡£çŤĺåĹřçŽĎsocketçŽĎæŰĞäżűæŔŔè£řçňęèĂŇåůšãĂĆ

äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijŇåőŽäźL'äžĘäÿĂäÿłäÿžåĹřæİěçŽĎåĚČçť'ăçŻŚæŐğåd'ŽäÿłéŸ§åĹŮçŽĎæű

```
import select
import threading

def consumer(queues):
    '''
    Consumer that reads data on multiple queues simultaneously
    '''
    while True:
        can_read, _, _ = select.select(queues,[],[])
        for r in can_read:
            item = r.get()
            print('Got:', item)

q1 = PollableQueue()
q2 = PollableQueue()
q3 = PollableQueue()
t = threading.Thread(target=consumer, args=([q1,q2,q3],))
t.daemon = True
t.start()

# Feed data to the queues
q1.put(1)
q2.put(10)
q3.put('hello')
q2.put(15)
...
```

åęĆædIJä¡ăèŕŢçİÄè£ŘèąŇåőČïijŇä¡ăäijŽåŔŚçŐřè£ŹäÿłæűĹèť'źèĂĚäijŽæŐěåŮůåĹřæL'ĂæIJL'çŽĎèćń

èőÍèőž

åŕźäžŐè¡őèŕćéİđçśżæŰĞäżűåŕźèśąïijŇæŕŤåęĆéŸ§åĹŮéĂŽåÿÿéČ¡æŸŕæŕŤè¿ČæčŸæL'ŃçŽĎéŮőécŸãĂ
ä¿ŇåęĆïijŇåęĆædIJä¡ăäÿ■ä¡£çŤĺäÿŁéİćçŽĎåěŮæŐěå■ŮæŁĂæIJŕïijŇ
ä¡ăåŤŕäÿĂçŽĎéĂL'æŃl'åŕśæŸŕçijŰåĘŹäžčçăĄæİěå¿łçŐŕéĄ■åŐĘè£ŹäžŽéŸ§åĹŮåźűä¡£çŤĺäÿĂäÿłåőŽæŮűå

```
import time
def consumer(queues):
    while True:
        for q in queues:
            if not q.empty():
                item = q.get()
                print('Got:', item)
```

```
        # Sleep briefly to avoid 100% CPU
        time.sleep(0.01)
```

è£Źæăůå Ą Žå Ěűåőđäÿ■å ŘĹçŘ Ę ïijŇè£Ÿäij Žåij Ţå Ěěå Ěűäż ŰçŽĎæĂğèČ¡éŮőécŸãĂĆ
ä¿ŇåęĆïijŇåęĆæđIJæŰřçŽĎæŢřæ■őècńåŁăåĚěåĹřäÿĂäÿłéŸ§åĹŮäÿ■ïijŇèĞşåřŚèęĄèŁś10æŕńçğŠæL'■èČ¡è
åęĆæđIJä¡ăäźŇåL'■çŽĎè¡őèŕćè£ŸèęĄåŐżè¡őèŕćåĚűäżŰåŕźèśąïijŇæŕŤåęĆç¡ŚçżIJåěŮæŐěå■ŮéĆčè£Ÿäij Žæ
ä¿ŇåęĆïijŇåęĆæđIJä¡ăæČşåŘŇæŮűè¡őèŕćåěŮæŐěå■ŮåŠŇéŸ§åĹŮïijŇä¡ăåŔŕèČ¡èęĄåČŔäÿŃéİćè£Źæăůä¡£

```
import select

def event_loop(sockets, queues):
    while True:
        # polling with a timeout
        can_read, _, _ = select.select(sockets, [], [], 0.01)
        for r in can_read:
            handle_read(r)
        for q in queues:
            if not q.empty():
                item = q.get()
                print('Got:', item)
```

è£ŹäÿłæŰźæąĹéĂŽè£ĞåřĘéŸ§åĹŮåŠŇåěŮæŐěå■Ůç■L'åŘŇåŕźå¿ĚæİèèğčåĘşäžĘåd'ğéČĺåĹĘçŽĎéŮőé
äÿĂäÿłå■ŢçŃňçŽĎ `select()` èřČçŤĺåŔŕècńåŘŇæŮűçŤĺæİěè¡őèŕćĂĆ
ä¡£çŤĺèűĚæŮűæĹŰåĚűäżŰå§žäžŐæŰűéŮt'çŽĎæIJžåĹűæİěæL'ğèąŇåŚĺæIJ§æĂğæčĂæ§ěåźűæšąæIJL'å£Ěèę
çŤŽèĞşïijŇåęĆæđIJæŢřæ■őècńåŁăåĚěåĹřäÿĂäÿłéŸ§åĹŮïijŇæűĹèt'źèĂĚåĞääźŐåŔŕäžěåőđæŮűçŽĎècńéĂ
åř¡çőąäijŽæIJL'äÿĂçĆźçĆźåžŢåśĆçŽĎI/Oæ■§èĂŮïijŇä¡£çŤĺåőČéĂŽåÿÿäijŽèŐůå¿ŮæŽt'åě¡çŽĎåŞ■åžŤæŮ

12.14 åIJĺUnixçşżçż§äÿŁéİćåŘŕåŁĺåőĹæŁd'è£ŽçĺŃ

éŮőécŸ

ä¡ăæČşçijŰåĘŹäÿĂäÿłä¡IJäÿžäÿĂäÿłåIJĺUnixæĹŰçśżUnixçşżçż§äÿŁéİćè£ŘèąŇçŽĎåőĹæŁd'è£ŽçĺŃè£

èğčåĘşæŰźæąĹ

åĹŽåžžäÿĂäÿłæ■čçąőçŽĎåőĹæŁd'è£ŽçĺŃéIJĂèęĄäÿĂäÿłçš¿çąőçŽĎçşżçż§èřČçŤĺåžŔåĹŮäžěåŔŁåŕźäž
äÿŃéİćçŽĎäžčçăĄåśŢçd'žäžĘæĂŐæăůåőŽäźL'äÿĂäÿłåőĹæŁd'è£ŽçĺŃïijŇåŔŕäžěåŘŕåŁĺåŘŐå¿Ĺåőźæ ŸŞçŽĎ

```
#!/usr/bin/env python3
# daemon.py

import os
import sys

import atexit
import signal
```

```
def daemonize(pidfile, *, stdin='/dev/null',
                          stdout='/dev/null',
                          stderr='/dev/null'):

    if os.path.exists(pidfile):
        raise RuntimeError('Already running')

    # First fork (detaches from parent)
    try:
        if os.fork() > 0:
            raise SystemExit(0)   # Parent exit
    except OSError as e:
        raise RuntimeError('fork #1 failed.')

    os.chdir('/')
    os.umask(0)
    os.setsid()
    # Second fork (relinquish session leadership)
    try:
        if os.fork() > 0:
            raise SystemExit(0)
    except OSError as e:
        raise RuntimeError('fork #2 failed.')

    # Flush I/O buffers
    sys.stdout.flush()
    sys.stderr.flush()

    # Replace file descriptors for stdin, stdout, and stderr
    with open(stdin, 'rb', 0) as f:
        os.dup2(f.fileno(), sys.stdin.fileno())
    with open(stdout, 'ab', 0) as f:
        os.dup2(f.fileno(), sys.stdout.fileno())
    with open(stderr, 'ab', 0) as f:
        os.dup2(f.fileno(), sys.stderr.fileno())

    # Write the PID file
    with open(pidfile,'w') as f:
        print(os.getpid(),file=f)

    # Arrange to have the PID file removed on exit/signal
    atexit.register(lambda: os.remove(pidfile))

    # Signal handler for termination (required)
    def sigterm_handler(signo, frame):
        raise SystemExit(1)

    signal.signal(signal.SIGTERM, sigterm_handler)
```

```
def main():
    import time
    sys.stdout.write('Daemon started with pid {}\n'.format(os.
↪getpid()))
    while True:
        sys.stdout.write('Daemon Alive! {}\n'.format(time.ctime()))
        time.sleep(10)

if __name__ == '__main__':
    PIDFILE = '/tmp/daemon.pid'

    if len(sys.argv) != 2:
        print('Usage: {} [start|stop]'.format(sys.argv[0]),␣
↪file=sys.stderr)
        raise SystemExit(1)

    if sys.argv[1] == 'start':
        try:
            daemonize(PIDFILE,
                      stdout='/tmp/daemon.log',
                      stderr='/tmp/dameon.log')
        except RuntimeError as e:
            print(e, file=sys.stderr)
            raise SystemExit(1)

        main()

    elif sys.argv[1] == 'stop':
        if os.path.exists(PIDFILE):
            with open(PIDFILE) as f:
                os.kill(int(f.read()), signal.SIGTERM)
        else:
            print('Not running', file=sys.stderr)
            raise SystemExit(1)

    else:
        print('Unknown command {!r}'.format(sys.argv[1]), file=sys.
↪stderr)
        raise SystemExit(1)
```

èęĄåŘŕåŁĺè£Źäÿłåő́ĹæŁd’è£ŻçĺŃïijŇçŤĺæĹůéIJĂèęĄä¡£çŤĺåęĆäÿŃçŽĎåŚ¡äżd’ïijŽ

```
bash % daemon.py start
bash % cat /tmp/daemon.pid
2882
bash % tail -f /tmp/daemon.log
Daemon started with pid 2882
Daemon Alive! Fri Oct 12 13:45:37 2012
Daemon Alive! Fri Oct 12 13:45:47 2012
...
```

åőĹæŁd'è£ŽçÍŃåŔŕäžěåőŇåĚĺåIJĺåŘŐåŔřè£ŘèąŇïijŇåŽäæ■d'è£ŹäÿłåŚ¡äžd'äijŽçńŇå■şè£ŤåŻđãĂĆ
äÿ■è£ĞïijŇä¡ääŔŕäžěåČŔäÿŁéİćéĆčæăůæ§ěçIJŃäÿŐåőČçŻÿåĚşçŽĎpidæŰĞäžűåŠŇæŮěå£ŮãĂĆèęĄåĄIJæ

```
bash % daemon.py stop
bash %
```

èőĺèőž

æIJňèŁĆåőŽäźL'äžĘäÿĂäÿłåĞ¡æŢř daemonize() ïijŇåIJĺçÍŃåžŔåŘŕåŁĺæŮűèćńèřČçŤĺä¡£å¿ŮçÍŃåžŔ
daemonize() åĞ¡æŢřåŔłæŐěåŔŮåĚşéŤőå■ŮåŔĆæŢřïijŇè£Źæăůç ŽĎèŕĺåŘŕéĂL'åŔĆæŢřåIJĺèćńä¡£çŤĺæŮ
åőČäijŽåijžåĹűçŤĺæĹůåČŔäÿŃéİćè£Źæăůä¡£çŤĺåőČïijŽ

```
daemonize('daemon.pid',
          stdin='/dev/null,
          stdout='/tmp/daemon.log',
          stderr='/tmp/daemon.log')
```

èĂŇäÿ■æŸŕåČŔäÿŃéİćè£ŹæăůåŕńçşŁäÿ■æÿĚçŽĎèřČçŤĺïijŽ

```
# Illegal. Must use keyword arguments
daemonize('daemon.pid',
          '/dev/null', '/tmp/daemon.log','/tmp/daemon.log')
```

åĹŽåžžäÿĂäÿłåőĹæŁd'è£ŽçÍŃçŽĎæ■ěéłd'çIJŃäÿŁåŐžäÿ■æŸŕå¿ĹæŸŞæĞĆïijŇä¡ĘæŸŕåd'ğä¡ŞæĂİæČ
éęŰåĚĹïijŇäÿĂäÿłåőĹæŁd'è£ŽçÍŃå£ĚéąžèęĄäżŐçĹűè£ŽçÍŃäÿ■èĎśçęžãĂĆ è£ŹæŸŕçŤś
os.fork() æŞ■ä¡IJæİěåőŇæĹŘçŽĎïijŇåźűçńŇå■şèćńçĹűè£ŽçÍŃçžĹæ■ćãĂĆ

åIJĺå■Řè£ŽçÍŃåŔŸæĹŘå■d'åĎ£åŘŐïijŇèřČçŤĺ os.setsid()
åĹŽåžžäźĘäÿĂäÿłåĚĺæŰřçŽĎè£ŽçÍŃäijŽèŕİïijŇåźűèő¿ç¡őå■Řè£ŽçÍŃäÿžéęŰéćĘãĂĆ
åőČäijŽèő¿ç¡õè£Źäÿłåŕ■Řè£ŽçÍŃäÿžæŰřçŽĎè£ŽçÍŃçżĎçŽĎéęŰéćĘïijŇåźűçąőä£İäÿ■äijŽåĘ■æIJL'æŐğåĹűç
åęĆæđIJè£ŹäžŽåŘňäÿŁåŐžåd'łé■Ťåźžïij ŇåŽääÿžåőČéIJĂèęĄåřĘåőĹæŁd'è£ŽçÍŃåŘŇçžĹçńŕåĹĘçęžåijĂåźű
èřČçŤĺ os.chdir() åŠŇ os.umask(0) æŤźåŔŸäžĘå¡ŞåL'■åůëä¡IJçŽőå¡Ţåźűé Ğ■ç¡őæŰĞäžűæİČéŹŘæ
ä£őæŤźçŽőå¡ŢéĂŽåÿÿæŸŕäÿłåě¡äÿžæĎŔïijŇåŽääÿžè£Źæăůåŕŕäžěä¡£å¿ŮåőČäÿ■åĘ■åůëä¡IJåIJĺèćńåŘŕåŁĺæ

åŘęåd'ŰäÿĂäÿłèřČçŤĺ os.fork() åIJĺè£ŹéĞŇæŽt'åŁăçěđçğŸçĆźãĂĆ
è£ŹäÿĂæ■ěä¡£å¿ŮåőĹæŁd'è£ŽçÍŃåd'śåŐžäžĘèŐůåŔŰæŰřçŽĎæŐğåĹűçžĹçńŕçŽĎèČ¡åŁŽåźűäÿŤèől'åőČæ
ïijĹæIJňèt'ĺäÿŁïijŇèřědaemonæŤ¿åijČäžĘåőČçŽĎäijŽèŕİéęŰéćĘä¡Őä¡■ïijŇåŽäæ■d'åĘ■äź§æšąæIJL'æİČéŹŘ
år¡çőąä¡ääŔŕäžěå£¡çŢèè£ŹäÿĂæ■ěïijŇä¡ĘæŸŕæIJĂåě¡äÿ■èęĄè£ŹäźĹåĄŽãĂĆ

äÿĂæŮęåőĹæŁd'è£ŽçÍŃèćńæ■čçąőçŽĎåĹĘçęžïijŇåőČäijŽéĞ■æŰřåĹİåğŃåŇŰæăĞåĞĘI/OæţĄæŇĞåĚ
è£ŹäÿĂéČĺåĹĘæIJL'çĆźéŽ¿æĞĆãĂĆèů§æăĞåĞĘI/OæţĄçŽÿåĚşçŽĎæŰĞäžűåŕźèśąçŽĎåijŢçŤĺåIJĺèğčéĞŁå
ïijĹsys.stdout, sys.__stdout__ç■L'ïijL'ãĂĆ äžĚäžĚçőĂå■ŢçŽĎåĚşéŮ■
sys.stdout åźűéĞ■æŰřæŇĞåőŽåőČæŸŕèąŇäÿ■éĂŽçŽĎïijŇ
åŽääÿžæšąåŁđæşŢç§ěéĄŞåőČæŸŕåŘęåĚĺéČĺéČ¡æŸŕçŤĺçŽĎæŸŕ sys.stdout ãĂĆ
è£ŹéĞŇïijŇæĹŚäžňæL'ŞåijĂäžĘäÿĂäÿłå■ŢçŃňçŽĎæŰĞäžűåŕźèśąïijŇåźűèřČçŤĺ os.
dup2() ïijŇ çŤĺåőČæĚäžčæŽ£èćń sys.stdout ä¡£çŤĺçŽĎæŰĞäžűæŔŘè£řçňęãĂĆ
è£ŹæăůïijŇsys.stdout ä¡£çŤĺçŽĎåŐ§åğŃæŰĞäžűäijŽèćńåĚşéŮ■åźűçŤśæŰřçŽĎæİěæŽ£æ■ćãĂĆ
è£ŸèęĄåijžèřČçŽĎæŸŕäžżä¡ŢçŤĺäžŐæŰĞäžűçijŰçăĄæĹŰæŰĞæIJňåd'ĎçŘĘçŽĎæăĞåĞĘI/OæţĄè£ŸäijŽä

åőĹæŁd'è£ŽçÍŃçŽĎäÿĂäÿłéĂŽåÿÿåőđèůţæŸŕåIJĺäÿĂäÿłæŰĞäžűäÿ■åĘŹåĚěè£ŽçÍŃIDïijŇåŔŕäžěèćńåĚ

daemonize() åĞ¡æŢřçŽĎæIJĂåŘŐéČĺåĹĘåEŹäžEè£Źäÿ łæŰĞäžűïijÑä¡EæŸŕåIJĺçĺŃåžŔçżĹæ■ćæŮűåĹă
atexit.register() åĞ¡æŢřæşĺåĘŇäžĘäÿĂäÿłåĞ¡æŢřåIJĺPythonèğčéĞŁåŹĺçżĹæ■ćæŮűæL'ğèąÑãĂĆ
äÿĂäÿłåŕźäžŐSIGTERMçŽĎä£qåŔůåď'ĎçŘĘåŹĺçŽĎåőŽäźL'åŘŇæăůéIJĂèęĄèćńäijŸéŻĚçŽĎåĚşéŮ■ãĂĆ
ä£qåŔůåď'ĎçŘĘåŹĺçőĂå■ŢçŽĎæŁŻåĞžäžĘ SystemExit() åijĆåÿÿãĂĆ
æĹŰèőÿè£ŹäÿĂæ■ěçIJŃäÿŁåŐżæšąå£ĚèęĄïijÑä¡EæŸŕæšąæIJL'åőČïijŇ
çżĹæ■ćä£qåŔůäijŽä¡£å¿Ůäÿ■æL'ğèąŇ atexit.register()
æşĺåĘŇçŽĎæÿĚçŘĘæŞ■ä¡IJçŽĎæŮűåĂŹåřśæİĂæŐL'äžEèğčéĞŁåŹĺãĂĆ
äÿĂäÿłæİĂæŐL'è£ŻçĺŃçŽĎä¿Ńå■ŘäžčçăĄåŔŕäžěåIJĺçĺŃåžŔæIJĂåŘŐçŽĎ stop
åŚ¡äžď'çŽĎæŞ■ä¡IJäÿ■çIJŃåĹřãĂĆ

æŻť'åď'ŽåĚşäžŐçijŰåĘŹåőĹæŁď'è£ŻçĺŃçŽĎä£qæĄŕåŔŕäžěæ§ěçIJŃãĂŁUNIX çŐŕåćČéńŸçžğçijŰçĺŃãĂŃ, çňňäžŇçL'Ĺ by W. Richard Stevens and Stephen A. Rago (Addison-Wesley, 2005)ãĂĆ åř¡çőąåőČæŸŕåĚşæşĺäÿŐCèŕ■èĺĂçijŰçĺŃïijÑä¡EæŸŕæL'ĂæIJL'çŽĎåĘĚåőźéČ¡éĂĆçŤĺäžŐPythonïijÑ åŻääÿžæL'ĂæIJL'éIJĂèęĄçŽĎPOSIXåĞ¡æŢřéČ¡åŔŕäžěåIJĺæăĞåĞĘåžŞäÿ■æL'¿åĹřãĂĆ

çňňå■ĄäÿL'çńäïijŽèĎŽæIJňçijŰçĺŃäÿŐçşżçż§çőąçŘĘ

èőÿåď'Žäžžä¡£çŤĺPythonä¡IJäÿžäÿĂäÿłshellèĎŽæIJňçŽĎæŻ£äżčïijŇçŤĺæİěåőđçŐřåÿÿçŤĺçşżçż§äżžåŁą

Contents:

13.1 éĂŽè£ĞéĞ■åőŽåŘŚ/çőąéĄŞ/æŰĞäżűæŐěåŔůè¿ŞåĚě

éŮőécŸ

ä¡ăåÿŇæIJŻä¡ăçŽĎèĎŽæIJňæŐěåŔůäżžä¡ŢçŤĺæĹůèőď'äÿžæIJĂçőĂå■ŢçŽĎè¿ŞåĚěæŰźåijŔãĂĆåŇĚæ
éĞ■åőŽåŘŚæŰĞäżűåĹřèŕěèĎŽæIJňïijŇæĹŰåIJĺåŚ¡äžď'èąŇäÿ■äijäéĂŠäÿĂäÿłæŰĞäżűåŘ■æĹŰæŰĞäżűåŘ■

èğčåĘşæŰźæąĹ

PythonåĘĚç¡őçŽĎ fileinput æĺąåİŮèől'è£ŹäÿłåŔŸå¿ŮçőĂå■ŢãĂĆåęĆæđIJä¡ăæIJL'äÿĂäÿłäÿŃéİćè

```
#!/usr/bin/env python3
import fileinput

with fileinput.input() as f_input:
    for line in f_input:
        print(line, end='')
```

éĆčäźĹä¡ăåŕśèČ¡äžěåL'■éİćæŔŘåĹřçŽĎæL'ĂæIJL'æŰźåijŔæİěäÿžæ■ď'èĎŽæIJňæŔŘä¿Żè¿ŞåĚěãĂĆåĄ
filein.py åźűåřĘåĚűåŔŸäÿžåŔŕæL'ğèąŇæŰĞäžűïijÑ éĆčäźĹä¡ăåŔŕäžěåČŔäÿŃéİćè£Źæăůèřčç̀ŤĺåőČïijŇ

```
$ ls | ./filein.py          # Prints a directory listing to stdout.
$ ./filein.py /etc/passwd   # Reads /etc/passwd to stdout.
$ ./filein.py < /etc/passwd # Reads /etc/passwd to stdout.
```

èőÍèőž

fileinput.input() åĹŻåžžåźűè£ŤåŻđäÿĂäÿł FileInput çśžçŽĎåőđä¿ŃãĂĆ èŕěåőđä¿ŃéŹď äžĘæŃěæIJL'äÿĂäžŻæIJL'çŤĺçŽĎåÿőåŁľ æŰźæşŢåď ŰïijŇåőČè£ŸåŔŕèćńå¡ŞåĄŽäÿĂäÿłäÿŁ
åŻăæ■ď ïijŇæŢť åŘĹèţůæİěïijŇåęĆæđIJæĹŚäžňèęĄåĘŹäÿĂäÿłæL'Şå■řåď ŽäÿłæŰĞäžűè¿ŞåĞžçŽĎèĎŽæIJň

```
>>> import fileinput
>>> with fileinput.input('/etc/passwd') as f:
>>>     for line in f:
...         print(f.filename(), f.lineno(), line, end='')
...
/etc/passwd 1 ##
/etc/passwd 2 # User Database
/etc/passwd 3 #

<other output omitted>
```

éĂŽè£ĞåřĘåőČä¡IJäÿžäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺä¡£çŤĺïijŇåŔŕäžěçąőä£ÍåőČäÿ■åE■ä¡£çŤĺæŮűæÚ
èĂŇäÿŤæĹŚäžňåIJĺäźŃçśžè£ŸæijŤçď žäžĘ FileInput çŽĎäÿĂäžŻæIJL'çŤĺçŽĎåÿőåŁľ æŰźæşŢæİěèŐůå

13.2 çżĹæ■ćçĺŃåžŔåźűçżŹåĞžéŤŹèŕŕä£ąææĄŕ

éŮőécŸ

ä¡ăæČşåŘŚæăĞåĞĘéŤŹèŕŕæL'Şå■řäÿĂæİąæűĹæĄŕåźűè£ŤåŻđæ§ŘäÿłéİđéŻűçŁűæĂĄçăĄæİěçżĹæ■ćçĺ

èğčåĘşæŰźæąĹ

ä¡ăæIJL'äÿĂäÿłçĺŃåžŔåČŔäÿŃéİćè£ŹæăůçżĹæ■ćïijŇæŁŻåĞžäÿĂäÿł SystemExit
åijĆåÿÿïijŇä¡£çŤĺéŤŹèŕŕæűĹæĄŕä¡IJäÿžåŔĆæŢřãĂĆä¿ŃåęĆïijŽ

```
raise SystemExit('It failed!')
```

åőČäijŽåřĘæűĹæĄŕåIJĺ sys.stderr äÿ■æL'Şå■řïijŇçĎűåŘŐçĺŃåžŔäžěçŁűæĂĄçăĄ1éĂĂåĞžãĂĆ

èőÍèőž

æIJňèŁĆèŹ¡çĎűå¿Łç§■ářŔïijŇä¡ĘæŸŕåőČèČ¡èğčåĘşåIJĺåĘŹèĎŽæIJňæŮűçŽĎäÿĂäÿłåÿÿèğĄéŮőécŸã
äź§ářśæŸŕèŕťïijŇå¡Şä¡ăæČşèęĄçŻĹæ■ćæ§ŘäÿłçĺŃåžŔæŮűïijŇä¡ăåŔŕèČ¡äijŽåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
import sys
sys.stderr.write('It failed!\n')
raise SystemExit(1)
```

åęĆæđIJä¡ăçŻť æŐěářĘæűĹæĄŕä¡IJäÿžåŔĆæŢřäijăçżŹ SystemExit() ïijŇéĆčäźĹä¡ăåŔŕäžěçIJĄçŢěåĚűäžŰæ■ěéłď ïijŇ æŕŤåęĆimportèŕ■åŘěæĹŰářĘéŤŹèŕŕæűĹæĄŕåĘŹåĚě sys.stderr

13.3 èğčæđŘåŚ¡äžď'èąŇéĂĽéąź

éŮőécŸ

ä¡ăçŽĎçÍŃåžŘåęĆä¡ŢèČ¡åď§èğčæđŘåŚ¡äžď'èąŇéĂĽéąźïijĹä¡■äžŐsys.argväÿ■ïijĽ

èğčåĘşæŰźæąĹ

argparse æÍąåİŮåŘŕèćńçŤÍæİěèğčæđŘåŚ¡äžď'èąŇéĂĽéąźãĂĆäÿŃéİćäÿĂäÿłçőĂå■Ţä¿Ńå■ŘæijŤçď'ż

```
# search.py
'''
Hypothetical command-line tool for searching a collection of
files for one or more text patterns.
'''
import argparse
parser = argparse.ArgumentParser(description='Search some files')

parser.add_argument(dest='filenames',metavar='filename', nargs='*')

parser.add_argument('-p', '--pat',metavar='pattern', required=True,
                    dest='patterns', action='append',
                    help='text pattern to search for')

parser.add_argument('-v', dest='verbose', action='store_true',
                    help='verbose mode')

parser.add_argument('-o', dest='outfile', action='store',
                    help='output file')

parser.add_argument('--speed', dest='speed', action='store',
                    choices={'slow','fast'}, default='slow',
                    help='search speed')

args = parser.parse_args()

# Output the collected arguments
print(args.filenames)
print(args.patterns)
print(args.verbose)
print(args.outfile)
print(args.speed)
```

èŕěçÍŃåžŘåőŽäźĽäžĘäÿĂäÿłåęĆäÿŃä¡£çŤÍçŽĎåŚ¡äžď'èąŇèğčæđŘåŹÍïijŽ

```
bash % python3 search.py -h
usage: search.py [-h] [-p pattern] [-v] [-o OUTFILE] [--speed {slow,
↪fast}]
                 [filename [filename ...]]
```

```
Search some files

positional arguments:
  filename

optional arguments:
  -h, --help            show this help message and exit
  -p pattern, --pat pattern
                        text pattern to search for
  -v                    verbose mode
  -o OUTFILE            output file
  --speed {slow,fast}   search speed
```

äÿŃéÍćçŽĎéČĺåĹĘæijŤçď žäžĘçÍŃåžŔäÿ■çŽĎæŢřæ■őéČĺåĹĘãĂĆäżŤçżĘèğĆåŕ§print()èŕ■åŔěçŽĎæĽ’Ş

```
bash % python3 search.py foo.txt bar.txt
usage: search.py [-h] -p pattern [-v] [-o OUTFILE] [--speed {fast,
 ↪slow}]
                 [filename [filename ...]]
search.py: error: the following arguments are required: -p/--pat

bash % python3 search.py -v -p spam --pat=eggs foo.txt bar.txt
filenames = ['foo.txt', 'bar.txt']
patterns  = ['spam', 'eggs']
verbose   = True
outfile   = None
speed     = slow

bash % python3 search.py -v -p spam --pat=eggs foo.txt bar.txt -o␣
 ↪results
filenames = ['foo.txt', 'bar.txt']
patterns  = ['spam', 'eggs']
verbose   = True
outfile   = results
speed     = slow

bash % python3 search.py -v -p spam --pat=eggs foo.txt bar.txt -o␣
 ↪results \
             --speed=fast
filenames = ['foo.txt', 'bar.txt']
patterns  = ['spam', 'eggs']
verbose   = True
outfile   = results
speed     = fast
```

åŕźäžŐéĂĽ’éąźåĂijçŽĎè£ŻäÿĂæ■ěåď’ĎçŘĘçŤśçÍŃåžŔæÍěåĘşåőŽïijŇçŤĺä¡äèĞłåůśçŽĎéĂżè¿ŚæÍěæŻ£

`print()` åĞ¡æŢřãĂĆ

èőÍèőž

argparse æÍąåİŮæŸŕæăĞåĞĘåžŞäÿ■æIJĂåď˘ğçŽĎæÍąåİŮäźŃäÿĂïijŇæŇěæIJL’åď˘ğéĞŔçŽĎéĚ■ç¡őé
æIJňèŁĆåŔłæŸŕæijŤçď˘žäžĘąĚűäÿ■æIJĂå§žçąĂçŽĎäÿĂäžŻçL’źæĂğïijŇåÿőåŁľä¡ääĚěéŮÍãĂĆ

äÿžäžĘèğčæđŘåŚ¡äžď˘èąŇéĂL’éąźïijŇä¡äéęŰåĚĹèęĄåĹŻåžžäÿĂäÿł ArgumentParser åőđä¿ŃïijŇ åźűä¡£çŤÍ add_argument() æŰźæşŢåčřæŸŐä¡äæČşèęĄæŤŕæŇĄçŽĎéĂL’éąźãĂĆ åIJÍæŕŔäÿł add-argument() èřČçŤÍäÿ■ïijŇdest åŔĆæŢřæŇĞåőŽèğčæđŘçżŞæđIJèćńæŇĞæť˘¿çżŹåšđæĂğçŽĎåŘ■å■ŮãĂĆ metavar åŔĆæŢřèćńçŤÍæİěçŤ§æĹŘäÿőåŁľä£ąæĄŕãĂĆaction åŔĆæŢřæŇĞåőŽèů§åšđæĂğåŕźåžŢçŽĎåď˘ĎçŘĘéĂżè¿ŚïijŇ éĂŽåÿÿçŽĎåĂijäÿž store ,èćńçŤÍæİěå■ŸåĆÍæ§ŘäÿłåĂijæĹŰèőšåď˘ŽäÿłåŔĆæŢřåĂijæŤűéŻĘåĹřäÿĂäÿłåĹŮèąÍäÿ■ãĂĆ äÿŃéİćçŽĎåŔĆæŢřæŤűéŻĘæL’ĂæIJL’åL’ľä¡ŹçŽĎåŚ¡äžď˘èąŇåŔĆæŢřåĹřäÿĂäÿłåĹŮèąÍäÿ■ãĂĆåIJÍæIJňä¿Ń

```
parser.add_argument(dest='filenames',metavar='filename', nargs='*')
```

äÿŃéİćçŽĎåŔĆæŢřæăźæ■őåŔĆæŢřæŸŕåŘęå■ŸåIJÍæİěëő¿ç¡őäÿĂäÿł Boolean æăĞå£ŮïijŽ

```
parser.add_argument('-v', dest='verbose', action='store_true',
                    help='verbose mode')
```

äÿŃéİćçŽĎåŔĆæŢřæŐěåŔŮäÿĂäÿłå■ŢçŃňåĂijåźűåřĘåĚűå■ŸåĆÍäÿžäÿĂäÿłå■ŮçňęäÿšïijŽ

```
parser.add_argument('-o', dest='outfile', action='store',
                    help='output file')
```

äÿŃéİćçŽĎåŔĆæŢřèŕťæŸŐåĚĄèőÿæ§ŘäÿłåŔĆæŢřéĞ■åď˘■åĞžçŐŕåď˘ŽæňąïijŇåźűåřĘåőČäžňè£¡åŁääÍ required æăĞå£ŮèąÍçď˘žèŕěåŔĆæŢřèĞşåřŚèęĄæIJL’äÿĂäÿłãĂĆ-p åŠŇ --pat èąÍçď˘žäÿď˘äÿłåŔĆæŢřåŘ■å¡ćåijŔéČ¡åŔŕä¡£çŤÍãĂĆ

```
parser.add_argument('-p', '--pat',metavar='pattern', required=True,
                    dest='patterns', action='append',
                    help='text pattern to search for')
```

æIJĂåŘŐïijŇäÿŃéİćçŽĎåŔĆæŢřèŕťæŸŐæŐěåŔŮäÿĂäÿłåĂijïijŇä¡ĘæŸŕäijŽåřĘåĚűåŠŇåŔŕèČ¡çŽĎéĂ

```
parser.add_argument('--speed', dest='speed', action='store',
                    choices={'slow','fast'}, default='slow',
                    help='search speed')
```

äÿĂæŮęåŔĆæŢřéĂL’éąźèćńæŇĞåőŽïijŇä¡ăåřšåŔŕäžěæL’ğèąŇ parser.parse() æŰźæşŢäžĘąĂĆ åőČäijŽåď˘ĎçŘĘ sys.argv çŽĎåĂijåźűè£ŤåŽđäÿĂäÿłçżŞæđIJåőđä¿ŇãĂĆ æŕŘäÿłåŔĆæŢřåĂijäijŽèćńèő¿ç¡őæĹŘèŕěåőđä¿Ńäÿ■‘‘add_a
æŰźæşŢçŽĎ‘‘dest‘‘ åŔĆæŢřæŇĞåőŽçŽĎåšđæĂğåĂijãĂĆ

è£Ÿå¿Ĺåď˘Žçğ■åĚűäžŰæŰźæşŢèğčæđŘåŚ¡äžď˘èąŇéĂL’éąźãĂĆ ä¿ŇåęĆïijŇä¡ääŔŕèČ¡äijŽæL’ŇåŁĺçŽĎåď˘ĎçŘĘ sys.argv æĹŰèĂĚä¡£çŤÍ getopt æÍąåİŮãĂĆ ä¡ĘæŸŕïijŇåęĆæđIJä¡äéĞĞçŤÍæIJňèŁĆçŽĎæŰźåijŔïijŇårĘäijŽåĞŔårŚå¿Ĺåď˘ŽåĘŮä¡Źäžčçăąï
argparse æÍąåİŮåůšçżŔäÿőä¡ääď˘ĎçŘĘäžĘãĂĆ ä¡ääŔŕèČ¡è£ŸäijŽçćřåĹřä¡£çŤÍ

optparse åžŞèğčæđŘéĂL'éąźçŽĎäžčçăĄãĂĆ åř¡çőą optparse åŠŇ argparse å¿ĹåČŔïijŇä¡ĘæŸŕåŘŐèĂĚæŻť'åĚĹè£Żïij ŇåŻäæ■d'åIJÍæŰřçŽĎçÍŃåžŔäÿ■ä¡ääžŤèŕěä¡£çŤĺåőČãĂĆ

13.4 è£ŘèąŇæŮűåijźåĞžåŕĘçăĄè¿ŞåĚěæŔŘçd'ž

éŮőécŸ

ä¡ääEŹäžEäÿłèĎŽæIJňïijŇè£ŘèąŇæŮűéIJĂèęĄäÿĂäÿłåŕEçăĄãĂĆæ■d'èĎŽæIJňæŸŕäžd'äžŠåijŔçŽĎïij èĂŇæŸŕéIJĂèęĄåijźåĞžäÿĂäÿłåŕEçăĄè¿ŞåĚěæŔŘçd'žïijŇèőľçŤĺæĹůèĞłåůśè¿ŞåĚěãĂĆ

èğčåEşæŰźæąĹ

è£ŹæŮűåĂŹPythonçŽĎ getpass æĺąåİÛæ■čæŸŕä¡äæL'ĂéIJĂèęĄçŽĎãĂĆä¡ääŔŕäžěèőľä¡ää¿Ĺè¡żæİ¿ åźűäÿŤäÿ■äijŽåIJÍçŤĺæĹůçżĹçńŕåŽđæŸ¿åŕEçăĄãĂĆäÿŃéİćæŸŕåĚůä¡ŞäžčçăĄïijž

```
import getpass

user = getpass.getuser()
passwd = getpass.getpass()

if svc_login(user, passwd):     # You must write svc_login()
   print('Yay!')
else:
   print('Boo!')
```

åIJÍæ■d'äžčçăĄäÿ■ïijŇsvc_login() æŸŕä¡ăèęĄåőđçŐřçŽĎåd'ĎçŘEåŕEçăĄçŽĎåĞ¡æŢřïijŇåĚůä¡Şç

èőÍèőž

æşÍæĎŔåIJÍåĽ'■éİćäžčçăĄäÿ■ getpass.getuser() äÿ■äijŽåijźåĞžçŤĺæĹůåŘ■çŽĎè¿ŞåĚěæŔŘçd'žãĂĆ åőČäijŽæăźæ■őèŕěçŤĺæĹůçŽĎshellçŐŕåćČæĹŰèĂĚäijŽä¿Íæ■őæIJňåIJřçşżçż§çŽĎåŕEçăĄåžŞïijĹæŤŕæŇĄ *pwd* æĺąåİÛçŽĎåźşåŔřïijL'æİěä¡£çŤĺå¡ŞåL'■çŤĺæĹůçŽĎçŻżå¡ŢåŘ■ïijŇ

åęĆædIJä¡äæČşæŸ¿çd'žçŽĎåijźåĞžçŤĺæĹůåŘ■è¿ŞåĚěæŔŘçd'žïijŇä¡£çŤĺåĘĚç¡őçŽĎ input åĞ¡æŢřïijŽ

```
user = input('Enter your username: ')
```

è£ŸæIJL'äÿĂçĆźå¿ĹéĞ■èęĄïijŇæIJL'äžŻçşżçż§åŔŕèČ¡äÿ■æŤŕæŇĄ getpass() æŰźæşŢéŽŘèŮŔè¿ŞåĚěåŕEçăĄãĂĆ è£Źçğ■æČĚåĘţäÿŃïijŇPythonäijŽæŔŘåL'■è■ęåŚĽä¡äè£ŹäžŻéŮőécŸïij

13.5 èŐůåŔŰçżĹçńŕçŽĎåd'ğåřŔ

éÙőécŸ

ä¡ăéIJĂèęĄç§ěéĄŞå¡ŞåL’■çżĹçńŕçŽĎåď ğářŔäžěä¿£æ■čçąőçŽĎæăijåijŔåŇŰè¿ŞåĞžãĂĆ

èğčåĘşæŰźæąĹ

ä¡£çŤÍ `os.get_terminal_size()` åĞ¡æŢřæİěåĄŽåĹřè£ŹäÿĂçĆźãĂĆ

äžčçăĄçď žä¿ŃïijŽ

```
>>> import os
>>> sz = os.get_terminal_size()
>>> sz
os.terminal_size(columns=80, lines=24)
>>> sz.columns
80
>>> sz.lines
24
>>>
```

èőÍèőž

æIJL’åď łåď ŽæŰźåijŔæİěå¿Ůç§ěçżĹçńŕåď ğářŔäžĘïijŇäżŐèŕżåŔŰçŐŕåćČåŔŸéĞŔåĹřæL’ğèąŇåžŢåśĆç
`ioctl()` åĞ¡æŢřç■L’ç■L’ãĂĆ äÿ■è£ĞïijŇäÿžäżĂäźĹèęĄåŐżçăŤçľ űè£ŹäžŽåď ■æİĆçŽĎåŁđæşŢèĂŇäÿ■æ

13.6 æL’ğèąŇåď ŰéČĺåŚ¡äžď åźűèŐůåŔŰåőČçŽĎè¿ŞåĞž

éÙőécŸ

ä¡ăæČşæL’ğèąŇäÿĂäÿłåď ŰéČĺåŚ¡äžď åźűäžěPythonå■ŮçňęäÿšçŽĎå¡ćåijŔèŐůåŔŰæL’ğèąŇçżŞæđIJãĂ

èğčåĘşæŰźæąĹ

ä¡£çŤÍ `subprocess.check_output()` åĞ¡æŢřãĂĆä¿ŃåęĆïijŽ

```
import subprocess
out_bytes = subprocess.check_output(['netstat','-a'])
```

è£ŹæőţäžčçăĄæL’ğèąŇäÿĂäÿłæŇĞåőŽçŽĎåŚ¡äžď åźűåřĘæL’ğèąŇçżŞæđIJäžěäÿĂäÿłå■ŮèŁĆå■Ůçňęäÿ
åęĆæđIJä¡ăéIJĂèęĄæŰĞæIJňå¡ćåijŔè£ŤåŽđïijŇåŁääÿĂäÿłèğčçăĄæ■ěéłď å■şåŔŕãĂĆä¿ŃåęĆïijŽ

```
out_text = out_bytes.decode('utf-8')
```

åęĆæđIJèćńæL’ğèąŇçŽĎåŚ¡äžď äžěéİđéŽűçăĄè£ŤåŽđïijŇåřśäijŽæŁŽåĞžåijĆåÿÿãĂĆ
äÿŃéİćçŽĎä¿Ńå■Řæ■ŢèŐůåĹřéŤŹèŕŕåźűèŐůåŔŰè£ŤåŽđçăĄïijŽ

```
try:
    out_bytes = subprocess.check_output(['cmd','arg1','arg2'])
except subprocess.CalledProcessError as e:
    out_bytes = e.output       # Output generated before error
    code      = e.returncode   # Return code
```

éżŸèőď'æČĚåĘţäÿŃïijŇcheck_output() äžĚäžĚè£ŤåŻđè¿ŞåĚěåĹřæăĞåĞĘè¿ŞåĞžçŽĎåĂijãĂĆ åęĆæđIJä¡ăéIJĂèęĄåŘŇæŮűæŤűéŻĘæăĞåĞĘè¿ŞåĞžåŠŇéŤŹèŕŕè¿ŞåĞžïijŇä¡£çŤÍ stderr åŔĆæŢřïijŽ

```
out_bytes = subprocess.check_output(['cmd','arg1','arg2'],
                                    stderr=subprocess.STDOUT)
```

åęĆæđIJä¡ăéIJĂèęĄçŤÍäÿĂäÿłèűĚæŮűæIJžåĹűæĭěæL'ğèąŇåŚ¡äžd'ïijŇä¡£çŤÍ timeout åŔĆæŢřïijŽ

```
try:
    out_bytes = subprocess.check_output(['cmd','arg1','arg2'],␣
↪timeout=5)
except subprocess.TimeoutExpired as e:
    ...
```

éĂŽåÿÿæĭèèőšïijŇåŚ¡äžd'çŽĎæL'ğèąŇäÿ■éIJĂèęĄä¡£çŤÍåĹřåžŢåśĆshellçŐŕåćČïijĹæŕŤåęĆshãĂĄbash äÿĂäÿłå■ŮçňęäÿšåĹŮèąÍäijŽèćńäijăéĂŠçżŹäÿĂäÿłä¡Őçžğçşżçż§åŚ¡äžd'ïijŇæŕŤåęĆ os.execve() ãĂĆ åęĆæđIJä¡ăæČşèől'åŚ¡äžd'èćńäÿĂäÿłshellæL'ğèąŇïijŇäijăéĂŠäÿĂäÿłå■ŮçňęäÿšåŔĆæŢřïij shell=True. æIJL'æŮűåĂŹä¡ăæČşèęĄPythonåŐżæL'ğèąŇäÿĂäÿłåd'■æİĆçŽĎshellåŚ¡äžd'çŽĎæŮűåĂŹè

```
out_bytes = subprocess.check_output('grep python | wc > out',␣
↪shell=True)
```

éIJĂèęĄæşÍæĎŘçŽĎæŸŕåIJÍshelläÿ■æL'ğèąŇåŚ¡äžd'äijŽå■ŸåIJÍäÿĂåőŽçŽĎåőL'åĚÍéčŐéŹl'ïijŇçL'źåÍ è£ŹæŮűåĂŹåŘŕäžěä¡£çŤÍ shlex.quote() åĞ¡æŢřæİèèőšåŔĆæŢřæ■čçąőçŽĎçŤÍåŘŇåijŢçŤÍåijŢèţůæİěã

èőÍèőž

ä¡£çŤÍ check_output() åĞ¡æŢřæŸŕæL'ğèąŇåd'ŰéČÍåŚ¡äžd'åźűèŐůåŘŰåĚűè£ŤåŻđåĂijçŽĎæIJĂçő ä¡ĘæŸŕïijŇåęĆæđIJä¡ăéIJĂèęĄåŕźå■Řè£ŻçÍŇåĄŽæŻt'åd'■æİĆçŽĎäžd'äžŠïijŇæŕŤåęĆçżŹåőČåŘŚéĂÀè¿Şå è£ŹæŮűåĂŹåŘŕçŻt'æŐěä¡£çŤÍ subprocess.Popen çśżãĂĆä¿ŇåęĆïijŽ

```
import subprocess

# Some text to send
text = b'''
hello world
this is a test
goodbye
'''

# Launch a command with pipes
```

```
p = subprocess.Popen(['wc'],
          stdout = subprocess.PIPE,
          stdin = subprocess.PIPE)

# Send the data and get the output
stdout, stderr = p.communicate(text)

# To interpret as text, decode
out = stdout.decode('utf-8')
err = stderr.decode('utf-8')
```

subprocess æÍąåİŮåŕźäžŐä¿İèţŰTTYçŽĎåď ŰéČÍåŚ¡äżď äÿ■åŘĹéĂĆçŤÍãĂĆ ä¿ŃåęĆïijŇä¡ääÿ■èČ¡ä¡£çŤĺåőČæİèèĞłåŁĺåŇŰäÿĂäÿłçŤÍæĹůè¿ŞåĚěåŕĘçăĄçŽĎäżżåŁąïijĹæŕŤåęĆäÿĂäÿłss è£ŹæŮűåĂŹïijŇä¡ăéIJĂèęĄä¡£çŤÍåĹřçňňäÿL'æŰźæÍąåİŮäžĘïijŇæŕŤåęĆå§žäžŐèŚŮåŘ■çŽĎ expect åőűæŮŘçŽĎåůěåĚůïijĹpexpectæĹŰçśżäijijçŽĎïijL'

13.7 åď■åĹűæĹŰèĂĚçğżåŁĺæŰĞäżűåŠŇçŻőå¡Ţ

éŮőécŸ

ä¡ăæČşèęĄåď■åĹűæĹŰçğżåŁĺæŰĞäżűåŠŇçŻőå¡ŢïijŇä¡ĘæŸŕåŘĹäÿ■æČşèřČçŤĺshellåŚ¡äżď ãĂĆ

èğčåĘşæŰźæąĹ

shutil æÍąåİŮæIJL'å¿Ĺåď Žä¿£æ■ůçŽĎåĞ¡æŢřåŘŕäžěåď ■åĹűæŰĞäżűåŠŇçŻőå¡ŢãĂĆä¡£çŤĺèţůæİěé

```
import shutil

# Copy src to dst. (cp src dst)
shutil.copy(src, dst)

# Copy files, but preserve metadata (cp -p src dst)
shutil.copy2(src, dst)

# Copy directory tree (cp -R src dst)
shutil.copytree(src, dst)

# Move src to dst (mv src dst)
shutil.move(src, dst)
```

è£ŹäžŻåĞ¡æŢřçŽĎåŘĆæŢřéČ¡æŸŕå■Ůçňęäÿšå¡ćåijŘçŽĎæŰĞäżűæĹŰçŻőå¡ŢåŘ■ãĂĆ åžŢåśĆèŕ■äźĽæĺąæŃ§äžĘçśżäijijçŽĎUnixåŚ¡äżď ïijŇåęĆäÿŁéIćçŽĎæşléĞŁéČĺåĹĘãĂĆ

éżŸèőď æČĚåĘţäÿŃïijŇåŕźäžŐçňęåŘůéŞ¿æŐěèĂŇåůšè£ŹäžŻåŚ¡äżď åď ĎçŘĘçŽĎæŸŕåőČæŇĞåŘŚçŽ ä¿ŃåęĆïijŇåęĆæđIJæžŘæŰĞäżűæŸŕäÿĂäÿłçňęåŘůéŞ¿æŐěïijŇéĆčäźĹçŻőæăĞæŰĞäżűåřĘäijŽæŸŕçňęåŘůé åęĆæđIJä¡ååŘłæČşåď ■åĹűçňęåŘůéŞ¿æŐěæIJňèžńïijŇéĆčäźĹéIJĂèęĄæŇĞåőŽåĚşéŤőå■ŮåŘĆæŢř follow_symlinks ,åęĆäÿŃïijŽ

åęĆædIJä¡ăæČşä£ÌçŢŹèćńåď■åĹűçŻőå¡Ţäÿ■çŽĎçňęåŘůéŞ¿æŐëijŇåČŔè£ŹæăůåĄŽiijŽ

```
shutil.copytree(src, dst, symlinks=True)
```

copytree() åŔŕäżěèőľä¡ăåIJĺåď■åĹűè£ĞçÍŃäÿ■éĂĽæŃľæĂğçŽĎå£¡çŢěæ§ŘäžŻæŰĞäżűæĹŰçŻőå
ä¡ăåŔŕäżěæŔŘä¿ŻäÿĂäÿłå£¡çŢěåĞ¡æŢřiijŇæŐěåŘŮäÿĂäÿłçŻőå¡ŢåŘ■åŠŇæŰĞäżűåŘ■åĹŮèąĺä¡IJäÿžè¿ŞåĚ

```
def ignore_pyc_files(dirname, filenames):
    return [name in filenames if name.endswith('.pyc')]

shutil.copytree(src, dst, ignore=ignore_pyc_files)
```

Since ignoring filename patterns is common, a utility function ignore_patterns() has already been provided to do it. For example:

shutil.copytree(src, dst, ignore=shutil.ignore_patterns('~','.pyc'))

èőĺèőž

ä¡£çŤĺ shutil åď■åĹűæŰĞäżűåŠŇçŻőå¡Ţäź§å£ŠçőĂå■ŢäžEçĆźåŘğãĂĆ
äÿ■è£ĞïijŇåŕźäžŐæŰĞäżűåĚČæŢřæ■őä£qæĄŕïijŇcopy2() è£ŹæăůçŽĎåĞ¡æŢřåŔłèČ¡år¡èĞłåůśæIJĂåď ğè
èő£éŮőæŮűéŮť ãĂĄåĹŻåžžæŮűéŮť åŠŇæİČéŹŘè£ŹäžŻå§žæIJñä£qæĄŕäijŽèćńä£İçŢŹïijŇ
ä¡ĘæŸŕåŕźäžŐæL'ĂæIJL'èĂĚãĂĄACLsãĂĄètĎæžŘforkåŠŇåĚűäżŰæŻť'æůśåśĆæňąçŽĎæŰĞäżűåĚČä£qæA
è£Źäÿłè£Ÿå¿Ůä¿İètŰäžŐåžŢåśĆæŞ■ä¡IJçşżçż§çśžåđŃåŠŇçŤĺæĹůæL'ĂæŃěæIJL'çŽĎèő£éŮőæİČéŹŘãĂĆ
ä¡ăéĂŽåÿÿäÿ■äijŽåŐżä¡£çŤĺ shutil.copytree() åĞ¡æŢřæİěæL'ğèąŇçşżçż§åď Ğäż¡ãĂĆ
å¡Şåď ĎçŘĘæŰĞäżűåŘ■çŽĎæŮűåĂŹïijŇæIJĂåě¡ä¡£çŤĺ os.path
äÿ■çŽĎåĞ¡æŢřæİěçąőä£ÍæIJĂåď ğçŽĎåŕçğżæd'■æĂğïijĹçL'źåĹńæŸŕåŘŇæŮűèĘéĂĆçŤĺäžŐUnixåŠŇWi
ä¿ŃåęĆïijŽ

```
>>> filename = '/Users/guido/programs/spam.py'
>>> import os.path
>>> os.path.basename(filename)
'spam.py'
>>> os.path.dirname(filename)
'/Users/guido/programs'
>>> os.path.split(filename)
('/Users/guido/programs', 'spam.py')
>>> os.path.join('/new/dir', os.path.basename(filename))
'/new/dir/spam.py'
>>> os.path.expanduser('~/guido/programs/spam.py')
'/Users/guido/programs/spam.py'
>>>
```

ä¡£çŤĺ copytree() åď■åĹűæŰĞäżűåď źçŽĎäÿĂäÿłæčŸæL'ŃçŽĎéŮőécŸæŸŕåŕźäžŐéŤŹèŕŕçŽĎåď Ď
ä¿ŃåęĆïijŇåIJĺåď■åĹűè£ĞçÍŃäÿ■ïijŇåĞ¡æŢřåŘŕèČ¡äijŽçćřåĹŕæ■§åİŘçŽĎçňęåŘůéŞ¿æŐěïijŇåŽääÿžæİČéŹ
äÿžäžĘèğčåĘşè£ŹäÿłéŮőécŸïijŇæL'ĂæIJL'çćřåĹŕçŽĎéŮőécŸäijŽèćńæŤűéŽĘåĹŕäÿĂäÿłåĹŮèąĺäÿ■åźűæL'Ş
äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘïijŽ

```
try:
    shutil.copytree(src, dst)
```

```
except shutil.Error as e:
    for src, dst, msg in e.args[0]:
          # src is source name
          # dst is destination name
          # msg is error message from exception
          print(dst, src, msg)
```

åęĆædIJä¡ăæŔŘä¿ŻåĚşéŤőå■ŮåŔĆæŢř `ignore_dangling_symlinks=True` ïijŇè£ŹæŮűåĂŹ `copytree()` äijŽå£¡çŢěæŐĽ'æŮăæŢĹçňęåŘůéŞ¿æŐěãĂĆ

æIJňèŁĆæijŤçd'žçŽĎè£ŹäžŻåĞ¡æŢřéČ¡æŸŕæIJĂåÿÿèğĄçŽĎãĂĆäÿ■è£ĞïijŇshutil è£ŸæIJĽæŻť'åd'ŽçŽĎåŠŇåd'■åĹűæŢřæ■őçŻÿåĚşçŽĎæŞ■ä¡IJãĂĆ åőČçŽĎæŰĞæąčå¿ĹåĂijå¿ŮäÿĂçIJŃïijŇåŔĆèĂČ Python documentation

13.8 åĹŻåžžåŠŇèğčåŐŃå¡Šæą̈čæŰĞäžű

éŮőécŸ

ä¡ăéIJĂèęĄåĹŻåžžæĹŰèğčåŐŃåÿÿèğĄæăijåijŔçŽĎå¡Šæą̈čæŰĞäžűïijĹæŕŤåęĆ.tar, .tgzæĹŰ.zipïijĽ

èğčåĘşæŰźæąĹ

`shutil` æĺąåİŮæŃěæIJĽäÿd'äÿłåĞ¡æŢřâĂŤâĂŤ `make_archive()` åŠŇ `unpack_archive()` åŘŕæt'¿äÿŁçŤĺåIJžãĂĆ ä¿ŃåęĆïijŽ

```
>>> import shutil
>>> shutil.unpack_archive('Python-3.3.0.tgz')

>>> shutil.make_archive('py33','zip','Python-3.3.0')
'/Users/beazley/Downloads/py33.zip'
>>>
```

`make_archive()` çŽĎçňnäžŇäÿłåŔĆæŢřæŸŕæIJ§æIJŻçŽĎè¿ŞåĞžæăijåijŔãĂĆ åŘŕäžěä¡£çŤĺ `get_archive_formats()` èŐůåŘŰæĽ'ĂæIJĽ'æŤŕæŇAçŽĎå¡Šæą̈čæăijåijŔåĹŮèąĺãĂĆä¿Ň

```
>>> shutil.get_archive_formats()
[('bztar', "bzip2'ed tar-file"), ('gztar', "gzip'ed tar-file"),
 ('tar', 'uncompressed tar file'), ('zip', 'ZIP file')]
>>>
```

èőÍèőž

Pythonè£ŸæIJĽåĚűäžŰçŽĎæĺąåİŮåŘŕçŤĺæİěåd'ĎçŘEåd'Žçğ■å¡Šæą̈čæăijåijŔïijĹæŕŤåęĆtarfile, zipfile, gzip, bz2ïijĽçŽĎåžŢåśĆçżĘèŁĆãĂĆ äÿ■è£ĞïijŇåęĆædIJä¡ääžĚäžĚåŘłæŸŕèęĄåĹŻåžžæĹŰæŔŘåŔŰ åŘŕäžěçŻť'æŐěä¡£çŤĺ `shutil` äÿ■çŽĎè£ŹäžŻéńŸåśĆåĞ¡æŢřãĂĆ

è£ŹäžŽåĞ¡æŢřè£ŸæIJL'å¿Ĺåď'ŽåĚűäžŰéĂL'éąźïijŇçŢĺäžŐæŮěå£ŮæL'Şå■řãĂĄéćĎæčĂãĂĄæŰĞäžűa
åŔĆèĂČ shutilæŰĞæąč

13.9 éĂŽè£ĞæŰĞäžűåŘ■æ§ěæL'¿æŰĞäžű

éŮőécŸ

ä¡ăéIJĂèęĄåEŹäÿĂäÿłæűL'åŔŁåĹřæŰĞäžűæ§ěæL'¿æŞ■ä¡IJçŽĎèĎŽæIJñïijŇæŕŤåęĆåŕźæŮěå£Ůå¡Šæą
ä¡ăäÿ■æČşåIJĺPythonèĎŽæIJñäÿ■èřČçŢĺshellïijŇæĹŰèĂĚä¡ăèęĄåőđçŐřäÿĂäžŽshelläÿ■èČ¡åĄŽçŽĎåŁ§èČ

èğčåEşæŰźæąĹ

æ§ěæL'¿æŰĞäžűïijŇåŔŕä¡£çŢĺ `os.walk()` åĞ¡æŢřïijŇäijăäÿĂäÿłéąűçžğçŻőå¡ŢåŘ■çżŹåőČãĂĆ
äÿŇéIćæŸŕäÿĂäÿłä¿Ňå■ŘïijŇæ§ěæL'¿çL'źåőŽçŽĎæŰĞäžűåŘ■åźűç■ŤåžŤæL'ĂæIJL'çñęåŘĹæÌąäžűçŽĎæŰ

```
#!/usr/bin/env python3.3
import os

def findfile(start, name):
    for relpath, dirs, files in os.walk(start):
        if name in files:
            full_path = os.path.join(start, relpath, name)
            print(os.path.normpath(os.path.abspath(full_path)))

if __name__ == '__main__':
    findfile(sys.argv[1], sys.argv[2])
```

ä£Ìå■ŸèĎŽæIJñäÿžæŰĞäžűfindfile.pyïijŇçĎűåŘŐåIJĺåŚ¡äžd'èąŇäÿ■æL'ğèąŇåőČãĂĆ
æŇĞåőŽåĹÌåğŃæ§ěæL'¿çŻőå¡Ţäžěå ŔĹåŘ■å■Ůä¡IJäÿžä¡■ç¡ôåŔĆæŢřïijŇåęĆäÿŃïijŽ

èŏÍèŏž

`os.walk()` æŰźæşŢäÿžæĹŚäžňéĄ■åŐEçŻőå¡ŢæăŚïijŇ
æŕŔæňąè£ŻåĚëäÿĂäÿłçŻőå¡ŢïijŇåőČäijŽè£ŤåŻđäÿĂäÿłäÿL'åĚČçżĎïijŇåŇĚåŘńçŽÿåŕźäžŐæ§ěæL'¿çŻőå¡Ţ
äžěåŔŁéĆčäÿłçŻőå¡ŢäÿŃéÍćçŽĎæŰĞäžűåŘ■åĹŮèąÍãĂĆ

åŕźäžŐæŕŔäÿłåĚČçżĎïijŇåŔłéIJĂæčĂæţŃäÿĂäÿŃçŻőæăĞæŰĞäžűåŘ■æŸŕåŘęåIJÍæŰĞäžűåĹŮèąÍäÿ■
`os.path.join()` åŘĹåźűèůŕå¿ĎãĂĆ äÿžäžĘéĄ£åĚ■åëĞæĂłçŽĎèůŕå¿ĎåŘ■æŕŤåęĆ `./`
`./foo//bar` ïijŇä¡£çŢĺäžĘåŔęåd'Űäÿd'äÿłåĞ¡æŢřæÌëä£őæ■čçżŞædIJãĂĆ çñňäÿĂäÿłæŸŕ
`os.path.abspath()` ,åőČæŐěåŔŮäÿĂäÿłèůŕå¿ĎïijŇåŔŕèČ¡æŸŕçŽÿåŕźèůŕå¿ĎïijŇæIJĂåŘŐè£ŤåŻđçżÌå
çñňäžŇäÿłæŸŕ `os.path.normpath()` ïijŇçŢĺæÌëè£ŤåŻđæ■čåÿÿèůŕå¿ĎïijŇåŔŕäžëèğčåEşåŔŇæŰIJæÌĘã

åř¡çőąè£ŹäÿłèĎŽæIJñçŽÿåŕźäžŐUNIXåźşåŘŕäÿŁéÍćçŽĎå¿Ĺåď'Žæ§ěæL'¿æÌëëőšèęĄçőĂå■Ţå¿Ĺåď'Žïij
åźűäÿŤïijŇè£ŸèČ¡å¿Ĺè¡żæÌ¿çŽĎåŁäåĚëåĚűäžŰçŽĎåŁ§èČ¡ãĂĆ
æĹŚäžňåE■æijŤçď'žäÿĂäÿłä¿Ňå■ŘïijŇäÿŃéÍćçŽĎåĞ¡æŢřæL'Şå■řæL'ĂæIJL'æIJĂè£Śèćńä£őæŤźè£ĞçŽĎæŰ

```
#!/usr/bin/env python3.3

import os
import time

def modified_within(top, seconds):
    now = time.time()
    for path, dirs, files in os.walk(top):
        for name in files:
            fullpath = os.path.join(path, name)
            if os.path.exists(fullpath):
                mtime = os.path.getmtime(fullpath)
                if mtime > (now - seconds):
                    print(fullpath)

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 3:
        print('Usage: {} dir seconds'.format(sys.argv[0]))
        raise SystemExit(1)

    modified_within(sys.argv[1], float(sys.argv[2]))
```

åIJÍæ■d'åĞ¡æŢřçŽĎå§žçąĂäźŃäÿŁïijŇä¡£çŤÍos,os.path,globç■L'çśżäijijæÍąåİŮïijŇä¡ääřśèČ¡åőđçŐřæŻt
åŘřåŘĆèĂČ5.11åřŘèŁĆåŠŇ5.13åřŘèŁĆç■L'çŻÿåĚşçńäèŁĆãĂĆ

13.10 èŕżåŘŰéĚ■ç¡őæŰĞäżű

éŮőécŸ

æĂŐæäůèŕżåŘŰæŹőéĂŽ.iniæäijåijŔçŽĎéĚ■ç¡őæŰĞäżűïij§

èğčåĘşæŰźæąĹ

configparser æÍąåİŮèČ¡èćńçŤÍæİěèŕżåŘŰéĚ■ç¡őæŰĞäżűãĂĆä¿ŇåęĆïijŇåĄĞèő¿ä¡äæIJL'åęĆäÿŃç

```
; config.ini
; Sample configuration file

[installation]
library=%(prefix)s/lib
include=%(prefix)s/include
bin=%(prefix)s/bin
prefix=/usr/local

# Setting related to debug configuration
[debug]
```

```
log_errors=true
show_warnings=False

[server]
port: 8080
nworkers: 32
pid-file=/tmp/spam.pid
root=/www/root
signature:
    =================================
    Brought to you by the Python Cookbook
    =================================
```

äÿŃéÌćæŸŕäÿĂäÿłèŕżåŔŮåŠŇæŔŘåŔŮåĚűäÿ■åĂijçŽĎä¿Ńå■ŘïijŽ

```
>>> from configparser import ConfigParser
>>> cfg = ConfigParser()
>>> cfg.read('config.ini')
['config.ini']
>>> cfg.sections()
['installation', 'debug', 'server']
>>> cfg.get('installation','library')
'/usr/local/lib'
>>> cfg.getboolean('debug','log_errors')

True
>>> cfg.getint('server','port')
8080
>>> cfg.getint('server','nworkers')
32
>>> print(cfg.get('server','signature'))

\=================================
Brought to you by the Python Cookbook
\=================================
>>>
```

åęĆæđIJæIJL’éIJĂèęĄïijŇä¡ăè£ŸèČ¡ä£őæŤźéĚ■ç¡ő åźűä¡£çŤÍ `cfg.write()` æŰźæşŢåřĘåĚűåĘŹåŻđåĹřæŰĞäżűäÿ■ãĂĆä¿ŃåęĆïijŽ

```
>>> cfg.set('server','port','9000')
>>> cfg.set('debug','log_errors','False')
>>> import sys
>>> cfg.write(sys.stdout)
```

```
[installation]
library = %(prefix)s/lib
include = %(prefix)s/include
bin = %(prefix)s/bin
prefix = /usr/local
```

```
[debug]
log_errors = False
show_warnings = False

[server]
port = 9000
nworkers = 32
pid-file = /tmp/spam.pid
root = /www/root
signature =
          ================================
          Brought to you by the Python Cookbook
          ================================
>>>
```

èőÍèőž

éĚ■ç¡őæŰĞäžűä¡IJäÿžäÿĂçğ■åŕŕèŕżæĂğå¿Ĺåě¡çŽĎæăijåijŔïijÑéİđåÿÿéĂĆçŤĺäžŐå■ŸåĆĺçĺŃåžŔäÿ■çŽ
åIJĺæŕŔäÿłéĚ■ç¡őæŰĞäžűäÿ■ïijÑéĚ■ç¡őæŢřæ■őäijŽèćńåĹĘçżĎïijĹæŕŤåęĆä¿Ńå■Řäÿ■çŽĎâĂIJinstallationâ
âĂIJdebugâĂİ åŠÑ âĂIJserverâĂİïijL'ãĂĆ æŕŔäÿłåĹĘçżĎåIJĺåĚűäÿ■æÑĞåőŽåŕźåžŤçŽĎåŘĎäÿłåŔŸéĞŔåĂ

åŕźäžŐåŔŕåőđçŐřåŘÑæăůåŁ§èČ¡çŽĎéĚ■ç¡őæŰĞäżűåŠÑPythonæžŘæŰĞäżűæŸŕæIJL'å¿Ĺåď'ğçŽĎäÿ■
éęŰåĚĹïijÑéĚ■ç¡őæŰĞäżűçŽĎèŕ■æşŢèęĄæŻť'èĞłçŤśäžŻïijÑäÿŃéİćçŽĎèţŃåĂijèŕ■åŘěæŸŕç■L'æŢĹçŽĎïij

```
prefix=/usr/local
prefix: /usr/local
```

éĚ■ç¡őæŰĞäżűäÿ■çŽĎåŘ■å■ŮæŸŕäÿ■åŇžåĹĘåď'ğåřŔåĘŹçŽĎãĂĆä¿ŃåęĆïijŽ

```
>>> cfg.get('installation','PREFIX')
'/usr/local'
>>> cfg.get('installation','prefix')
'/usr/local'
>>>
```

åIJĺèğčædŘåĂijçŽĎæŮűåĂŹïijÑgetboolean() æŰźæşŢæ§ěæL'¿äżżä¡ŢåŕŕèąŇçŽĎåĂijãĂĆä¿ŃåęĆ

```
log_errors = true
log_errors = TRUE
log_errors = Yes
log_errors = 1
```

æĹŰèőÿéĚ■ç¡őæŰĞäżűåŠÑPythonäžčçăĄæIJĂåď'ğçŽĎäÿ■åŘÑåIJĺäžŐïijÑåőČåźűäÿ■æŸŕäżŐäÿŁèĂÑ
æŰĞäżűæŸŕåőL'èčĚäÿĂäÿłæŢť'ä¡ŞèćńèŕżåŕŰçŽĎãĂĆåęĆædIJçćřåĹřäžĘåŔŸéĞŔæŻ£æ■ćïijÑåőČåőđéŹĚä
ä¿ŃåęĆïijÑåIJĺäÿŃéİćè£ŹäÿłéĚ■ç¡őäÿ■ïijÑprefix åŔŸéĞŔåIJĺä¡£çŤĺåőČçŽĎåŔŸéĞŔäźŃåL'■æĹŰäźŃåŕ

```
[installation]
library=%(prefix)s/lib
```

```
include=%(prefix)s/include
bin=%(prefix)s/bin
prefix=/usr/local
```

ConfigParser æIJL'äÿłåőźæŸŞèćńå£¡èğĘçŽĎçL'źæĂğæŸŕåőČèČ¡äÿĂæňąèŕżåŔŰåd'Žäÿłéĕ■ç¡őæŰ
ä¿ŇåęĆïijŇåĄĞèő¿äÿĂäÿłçŤĺæĹůåČŔäÿŃéÍćè£ŹæăůædĎéĂääžĘäžŰäžňçŽĎéĕ■ç¡őæŰĞäžűïijŽ

```
; ~/.config.ini
[installation]
prefix=/Users/beazley/test

[debug]
log_errors=False
```

èŕżåŔŰè£ŹäÿłæŰĞäžűïijŇåőČåřśèČ¡èů§äźŇåL'■çŽĎéĕ■ç¡őåŘĹåźűèţůæÍěãĂĆåęĆïijŽ

```
>>> # Previously read configuration
>>> cfg.get('installation', 'prefix')
'/usr/local'

>>> # Merge in user-specific configuration
>>> import os
>>> cfg.read(os.path.expanduser('~/.config.ini'))
['/Users/beazley/.config.ini']

>>> cfg.get('installation', 'prefix')
'/Users/beazley/test'
>>> cfg.get('installation', 'library')
'/Users/beazley/test/lib'
>>> cfg.getboolean('debug', 'log_errors')
False
>>>
```

äžŤçżĘèğĆåŕ§äÿŃ prefix åŔŸéĞŔæŸŕæĂŐæăůèęĘçŻŰůåĔűäžŰçŻÿůĔşůŔŸéĞŔçŽĎïijŇærŤåęĆ
library çŽĎèő¿åőŽůĂijãĂĆ äžğçŤ§è£Źçğ■çżŞæđIJçŽĎůŐ§ůŻäæŸŕåŔŸéĞŔçŽĎæŤźůĘźéĞĞůŔŰçŽĎæ
ä¡ăůŔŕäžěůČŔäÿŃéÍćè£ŹæăůůĄŽèŕŢéłŇïijŽ

```
>>> cfg.get('installation','library')
'/Users/beazley/test/lib'
>>> cfg.set('installation','prefix','/tmp/dir')
>>> cfg.get('installation','library')
'/tmp/dir/lib'
>>>
```

æIJĂůŘŐè£ŸæIJL'å¿ĹéĞ■èęĄäÿĂçĆźèęĄæşÍæĎŔçŽĎæŸŕPythonåźűäÿ■èČ¡æŤŕæŇĄ.iniæŰĞäžűůIJÍůĺ
çąőä£Íä¡ăůůšçżŔůŘĆéŸĔäžĘconfigparseræŰĞæąčäÿ■çŽĎèŕ■æşŢèŕęæČĔäžěůŔŁæŤŕæŇĄçL'źæĂğãĂĆ

13.11 çżŹçőĂů■ŢèĎŽælJňůćđůŁăæŮěů£ŮůŁ§èČ¡

éŮőécŸ

ä¡ăåÿŇæIJŻåIJÍèĎŽæIJňåŠŇçÍŃåžŔäÿ■årĘèŕŁæŰ■ä£ąæĄŕåĘŹåĚěæŮěå£ŮæŰĞäżűãĂĆ

èğčåĘşæŰźæąĹ

The easiest way to add logging to simple programs is to use the logging module. For example: æĽŞå■řæŮěå£ŮæIJĂçőĂå■ŢæŰźåijŔæŸŕä¡£çŤÍ `logging` æÍąåİŮãĂĆä¿ŃåęĆïijŽ

```
import logging

def main():
    # Configure the logging system
    logging.basicConfig(
        filename='app.log',
        level=logging.ERROR
    )

    # Variables (to make the calls that follow work)
    hostname = 'www.python.org'
    item = 'spam'
    filename = 'data.csv'
    mode = 'r'

    # Example logging calls (insert into your program)
    logging.critical('Host %s unknown', hostname)
    logging.error("Couldn't find %r", item)
    logging.warning('Feature is deprecated')
    logging.info('Opening file %r, mode=%r', filename, mode)
    logging.debug('Got here')

if __name__ == '__main__':
    main()
```

äÿŁéİćäžŤäÿłæŮěå£ŮèřČçŤÍïijĹcritical(), error(), warning(), info(), debug()ïijĽäžěéŹ■åžŔæŰźåijŔèąÍçd'žäÿ■åŘŇçŽĎäÿěéĞ■çžğåĹńãĂĆ `basicConfig()` çŽĎ `level` åŔĆæŢřæŸŕäÿĂäÿłè£ĞæżďåŹÍãĂĆ æĽĂæIJĽçžğåĹńä¡ŐäžŐæ■d'çžğåĹńçŽĎæŮěå£ŮæűĹæĄŕéČ¡äijŽèćńå£¡çŢěæŐĽãĂĆ ærŔäÿłloggingæŞ■ä¡IJçŽĎåŔĆæŢřæŸŕäÿĂäÿłæűĹæĄŕå■ŮçňęäÿšïijŇåŘŐéİćåĘ■èů§äÿĂäÿłæĹŰåd'ŽäÿłåŔĆ æđĎéĂăæIJĂçżĹçŽĎæŮěå£ŮæűĹæĄŕçŽĎæŮűåĂŹæĹŚäžňä¡£çŤÍäžĘ%æŞ■ä¡IJçňęæİěæäijåijŔåŇŰæűĹæĄ

è£ŘèąŇè£ŹäÿłçÍŃåžŔåŘŐïijŇåIJÍæŰĞäżű `app.log` äÿ■çŽĎåĘĚåőźåžŤèŕěæŸŕäÿŃéİćè£ŹæăůïijŽ

```
CRITICAL:root:Host www.python.org unknown
ERROR:root:Could not find 'spam'
```

If you want to change the output or level of output, you can change the parameters to the basicConfig() call. For example: åęĆædIJä¡ăæČşæŤźåŔŸè¿ŞåĞžç■L'çžğïijŇä¡ăåŔŕäžěä£őæŤź `basicConfig()` èřČçŤÍäÿ■çŽĎåŔĆæŢřãĂĆä¿ŃåęĆïijŽ

```
logging.basicConfig(
     filename='app.log',
     level=logging.WARNING,
     format='%(levelname)s:%(asctime)s:%(message)s')
```

æIJĂåŘŐè¿ŞåĞžåŔŸæĹŘåęĆäÿŃïijŽ

```
CRITICAL:2012-11-20 12:27:13,595:Host www.python.org unknown
ERROR:2012-11-20 12:27:13,595:Could not find 'spam'
WARNING:2012-11-20 12:27:13,595:Feature is deprecated
```

äÿŁéİćçŽĎæŮěå£ŮéĚ■ç¡őéČ¡æŸŕçąñçijŰçăĄåĹřçÍŃåžŔäÿ■çŽĎãĂĆåęĆæđIJä¡ăæČşä¡£çŤĺéĚ■ç¡őæŰČåŔŕäżěåČŔäÿŃéİćè£Źæăůä£őæŤź `basicConfig()` èřČçŤĺïijŽ

```
import logging
import logging.config

def main():
    # Configure the logging system
    logging.config.fileConfig('logconfig.ini')
    ...
```

åĹŻåżžäÿĂäÿłäÿŃéİćè£ŹæăůçŽĎæŰĞäżűïijŇåŘ■å■ŮåŔń `logconfig.ini` ïijŽ

```
[loggers]
keys=root

[handlers]
keys=defaultHandler

[formatters]
keys=defaultFormatter

[logger_root]
level=INFO
handlers=defaultHandler
qualname=root

[handler_defaultHandler]
class=FileHandler
formatter=defaultFormatter
args=('app.log', 'a')

[formatter_defaultFormatter]
format=%(levelname)s:%(name)s:%(message)s
```

åęĆæđIJä¡ăæČşä£őæŤźéĚ■ç¡őïijŇåŔŕäżěçŻť æŐěçijŰè¿ŚæŰĞäżűlogconfig.iniå■şåŔŕãĂĆ

èőÍèőž

åř¡çőąåŕźäžŐ `logging` æÍąåİŮèĂŇåůšæIJL’å¿Ĺåď’ŽæŻť’énŸçžğçŽĎéĚ■ç¡őéĂL’éąźïijŇ
äÿ■è£Ğè£ŹéĞŇçŽĎæŰźæąĹåŕźäžŐçőĂå■ŢçŽĎçÍŃåžŘåŠŇèĎŽæIJňåůšçżŔèűşåď§äžĘãĂĆ
åŘłæČşåIJÍèřČçŤÍæŮěå£ŮæŞ■ä¡IJåL’■åĚĹæL’ğèąŇäÿŃbasicConfig()åĞ¡æŢřæŰźæşŢïijŇä¡äçŽĎçÍŃåžŘåřśè

åęĆæđIJä¡ăæČşèęĄä¡äçŽĎæŮěå£ŮæűĹæĄŕåĘŹåĹřæăĞåĞĘéŤŹèŕŕäÿ■ïijŇèĂŇäÿ■æŸŕæŮěå£ŮæŰĞäž
`basicConfig()` æŮűäÿ■äijäæŰĞäžűůåŘ■åŔĆæŢřå■şåŘŕãĂĆä¿ŃåęĆïijŽ

```
logging.basicConfig(level=logging.INFO)
```

`basicConfig()` åIJÍçÍŃåžŘäÿ■åŘłèČ¡èćńæL’ğèąŇäÿĂæňąãĂĆåęĆæđIJä¡äçĺ■åŘŐæČşæŤźåŔŸæŮěå
åřśéIJĂèęĄåĚĹèŐůåŘŰ `root logger` ïijŇçĎűåŘŐçŻť’æŐëä£őæŤźåőČãĂĆä¿ŃåęĆïijŽ

```
logging.getLogger().level = logging.DEBUG
```

éIJĂèęĄåijžèřČçŽĎæŸŕæIJňèŁĆåŔłæŸŕæijŢçđ’žäžĘ `logging`
æÍąåİŮçŽĎäÿĂäžŻå§žæIJňçŢÍæşŢãĂĆ åőČåŔŕäžěåĄŽæŻť’åđ’ŽæŻť’énŸçžğçŽĎåőŽåĹűãĂĆ
åĚşäžŐæŮěå£ŮåőŽåĹűåŇŰäÿĂäÿłå¿Ĺåě¡çŽĎèţĐæžŘæŸŕ Logging Cookbook

13.12 çżŹåĞ¡æŢřåžŞåćđåŁäæŮěå£ŮåŁ§èČ¡

éŮőécŸ

ä¡ăæČşçżŹæ§Řäÿłå Ğ¡æŢřåžŞåćđåŁäæŮěå£ŮåŁ§èČ¡ïijŇä¡ĘæŸŕåŔĹäÿ■èČ¡å¡śåŞ■åĹřéĆčäžŻäÿ■ä¡£çŢĺ

èğčåĘşæŰźæąĹ

åŕźäžŐæČşèęĄæL’ğèąŇæŮěå£ŮæŞ■ä¡IJçŽĎåĞ¡æŢřåžŞèĂŇåůšïijŇä¡ăåžŢèŕěåĹŻåżžäÿĂäÿłäÿŞåśđçŽĎ
`logger` åŕźèśąïijŇåźűäÿŤåČŔäÿŃéİćè£ŹæăůåĹİåğŃåŇŰéĚ■ç¡őïijŽ

```
# somelib.py

import logging
log = logging.getLogger(__name__)
log.addHandler(logging.NullHandler())

# Example function (for testing)
def func():
    log.critical('A Critical Error!')
    log.debug('A debug message')
```

ä¡£çŢÍè£ŹäÿłéĚ■ç¡őïijŇézŸèőď’æČĚåĘţäÿŃäÿ■äijŽæL’Şå■řæŮěå£ŮãĂĆä¿ŃåęĆïijŽ

```
>>> import somelib
>>> somelib.func()
>>>
```

äÿ■è£ĞïijŇåęĆæđIJéĚ■ç¡ő è£ĞæŮěå£Ůçşžçż§ïijŇéĆčäźĹæŮěå£ŮæűĹæĄŕæĽŞå■řářšåijĂåğŃçŤ§æŢĹï

```
>>> import logging
>>> logging.basicConfig()
>>> somelib.func()
CRITICAL:somelib:A Critical Error!
>>>
```

èőÍèőž

éĂŽåÿÿæİěèőšïijŇä¡ääÿ■åžŤèŕěåIJÍåĞ¡æŢřåžŞäžčçăĄäÿ■èĞłåůśéĚ■ç¡őæŮěå£Ůçşžçż§ïijŇæĹŰèĂĚæŸ

èřČçŤÍ `getLogger(__name__)` åĹŻåżžäÿĂäÿłåŠŇèřČçŤÍæĺąåİŮåŘŇåŘ■çŽĎlog-
geræĺąåİŮãĂĆ çŤšäžŐæĺąåİŮéČ¡æŸŕåŤŕäÿĂçŽĎïijŇåŻäæ■ď åĹŻåżžçŽĎloggeräź§åřĘæŸŕåŤŕäÿĂçŽĎãĂĆ

`log.addHandler(logging.NullHandler())` æŞ■ä¡IJåřĘäÿĂäÿłçľžåď ĎçŘĘåŹÍçżŚåőŽåĹřåĹ
äÿĂäÿłçľžåď ĎçŘĘåŹÍéżŸèőď äijŽå£¡çŢěěřČçŤÍæĽĂæIJĽçŽĎæŮěå£ŮæűĹæĄŕãĂĆ
åŻäæ■ď ïijŇåęĆæđIJä¡£çŤÍèŕěåĞ¡æŢřåžŞçŽĎæŮűåĂŹè£Ÿæšąæ IJĽéĚ■ç¡őæŮěå£ŮïijŇéĆčäźĹåřĘäÿ■äijŽæ

è£ŸæIJĽäÿĂçĆźåŕśæŸŕåŕźäžŐåŘĎäÿłåĞ¡æŢřåžŞçŽĎæŮěå£ŮéĚ■ç¡őåŔŕäžěæŸŕçŻÿäžŠçŃňçńŃçŽĎïij
ä¿ŇåęĆïijŇåŕźäžŐåęĆäÿŃçŽĎäžčçăĄïijŽ

```
>>> import logging
>>> logging.basicConfig(level=logging.ERROR)

>>> import somelib
>>> somelib.func()
CRITICAL:somelib:A Critical Error!

>>> # Change the logging level for 'somelib' only
>>> logging.getLogger('somelib').level=logging.DEBUG
>>> somelib.func()
CRITICAL:somelib:A Critical Error!
DEBUG:somelib:A debug message
>>>
```

åIJÍè£ŹéĞŇïijŇæăźæŮěå£ŮèćńéĚ■ç¡őæĹŘäžĚäžĚè¿ŞåĞžERRORæĹŰæŻť énŸçžğåĹńçŽĎæűĹæĄŕãĂ
äÿ■è£Ğ ïijŇsomelib çŽĎæŮěå£ŮçžğåĹńèćńå■ŢçŃňéĚ■ç¡őæĹŘåŔŕäžěè¿ŞåĞždebugçžğåĹńçŽĎæűĹæĄŕïi
åČŔè£ŹæăůæŻť æŤźå■ŢçŃňæĺąåİŮçŽĎæŮěå£ŮéĚ■ç¡őåŕźäžŐèřČèŕŢæİěèőšæŸŕå¿ĹæŰźä¿£çŽĎïijŇ
åŻääÿžä¡äæŮäéIJĂåŐżæŻť æŤźäżžä¡ŢçŽĎåĚĺåśĂæŮěå£ŮéĚ■ç¡őâĂŤâĂŤåŘłéIJĂèęĄä£őæŤźä¡äæČşèęĄæŻ

Logging HOWTO èŕęçżĘäžŃçż■äžĘåęĆä¡ŢéĚ■ç¡őæŮěå£ŮæĺąåİŮåŠŇåĚűäžŰæIJĽçŤÍæŁĂåůğïijŇåŔŕä

13.13 åőđçŐřäÿĂäÿłèőąæŮűåŹÍ

éŮőécŸ

ä¡äæČşèőřå¡ŢçÍŃåžŔæĽğèąŇåď ŽäÿłäżžåŀąæĽĂèŁśèť źçŽĎæŮűéŮť

èğčåĘşæŰźæąĹ

time æĺąåİŮåŇĚåŘńå¿Ĺåď˙ŽåĞ¡æŢřæİěæĽ˙ğèąŇèů§æŮűéŮť˙æIJĽ˙åĚşçŽĎåĞ¡æŢřãĂĆ
år¡çőąåęĆæ■ď˙ïijŇéĂŽåÿÿæĹŚäżňäijŽåIJĺæ■ď˙å§žçąĂäźŃäÿŁæđĎéĂääÿĂäÿłæŻť˙éńŸçžğçŽĎæŐěåŘčæİěæİ

```
import time

class Timer:
    def __init__(self, func=time.perf_counter):
        self.elapsed = 0.0
        self._func = func
        self._start = None

    def start(self):
        if self._start is not None:
            raise RuntimeError('Already started')
        self._start = self._func()

    def stop(self):
        if self._start is None:
            raise RuntimeError('Not started')
        end = self._func()
        self.elapsed += end - self._start
        self._start = None

    def reset(self):
        self.elapsed = 0.0

    @property
    def running(self):
        return self._start is not None

    def __enter__(self):
        self.start()
        return self

    def __exit__(self, *args):
        self.stop()
```

è£ŹäÿłçśżåőŽäźĽ˙äžĘäÿĂäÿłåŘŕäžěèćńçŤĺæĹůæăźæ■őéIJĂèęĄåŘŕåŁĺãĂĄåĄIJæ■ćåŠŇéĞ■ç¡őçŽĎèőąą
åőČäijŽåIJĺ elapsed åšđæĂğäÿ■èőřå¡ŢæŢť˙äÿłæűĹèĂŮæŮűéŮť˙ãĂĆ
äÿŃéİćæŸŕäÿĂäÿłä¿Ńå■ŘæİěæijŤçď˙žæĂŐæăůä¡£çŤĺåőČïijŽ

```
def countdown(n):
    while n > 0:
        n -= 1

# Use 1: Explicit start/stop
t = Timer()
t.start()
countdown(1000000)
```

```
t.stop()
print(t.elapsed)

# Use 2: As a context manager
with t:
    countdown(1000000)

print(t.elapsed)

with Timer() as t2:
    countdown(1000000)
print(t2.elapsed)
```

èőÍèőž

æIJňèŁĆæŔŘä¿ŻäžĘäÿĂäÿłçőÅå■ŢèĂŇåőđçŤÍçŽĎçśżæİěåőđçŐřæŮűéŮťèőřå¡ŢäžěåŔŁèĂŮæŮűèőąçć
åŘŇæŮűäź§æŸřåŕżä¡£çŤÍwithèŕ■åŔěäžěåŔŁäÿŁäÿŃæŰĞçőąçŘĘåŹÍå■Ŕèőőç ŽĎäÿĂäÿłå¿Ĺåě¡çŽĎæijŤçď ž

åIJÍèőąæŮűäÿ■èęĄèĂČèŹŚäÿĂäÿłåžŢåśĆçŽĎæŮűéŮťåĞ¡æŢřéŮőécŸãĂĆäÿĂèĹňæİěèřťïijŇ
ä¡£çŤÍ `time.time()` æĹŰ `time.clock()` èőąçőŮçŽĎæŮűéŮťçš¿åžęåŻăæŞ■ä¡IJçşżçż§çŽĎäÿ■åŘŇäijŽ
èĂŇä¡£çŤÍ `time.perf_counter()` åĞ¡æŢřåŔŕäžěçąőä£Ïä¡£çŤÍçşżçż§äÿŁéİćæIJĂçš¿çąőçŽĎèőąæŮűåŹ

äÿŁè£řäžčçăĄäÿ■çŤś `Timer` çśżèőřå¡ŢçŽĎæŮűéŮťæŸréŠ§èąĺæŮűéŮťïijŇåźűåŇĚåŘńäžĘæĽĂæIJĽä
åęĆædIJä¡ääŔłæČşèőąçőŮèřěè£ŻçÍŃæĽĂèŁśèťźçŽĎCPUæŮűéŮťïijŇåžŤèŕěä¡£çŤÍ `time.process_time()` æİěäžčæŻ£ïijŽ

```
t = Timer(time.process_time)
with t:
    countdown(1000000)
print(t.elapsed)
```

`time.perf_counter()` åŠŇ `time.process_time()` éČ¡äijŽè£ŤåŽđářŔæŢřå¡ćåijŔçŽĎçğŠæŢřæŮűéŮťãĂĆ åődđéŹĚçŽĎæŮűéŮťåĂijæšąæIJĽäżżä¡ŢæĎŔäźĽïij

æŻťåď ŽåĚşäžŐèőąæŮűåŠŇæĂğèČ¡åĹĘædŘçŽĎä¿Ńå■ŘèŕůåŔĆèĂČ14.13åřŔèŁĆãĂĆ

13.14 éŹŘåĹűåĘĚå■ŸåŠŇCPUçŽĎä¡£çŤÍéĞŔ

éŮőécŸ

ä¡ăæČşåŕźåIJÍUnixçşżçż§äÿŁéİćè£ŘèąŇçŽĎçÍŃåžŔèő¿ç¡őåĘĚå■ŸæĹŰCPUçŽĎä¡£çŤÍéŹŘåĹűãĂĆ

èğčåĘşæŰźæąĹ

`resource` æÍąåİŮèČ¡åŘŇæŮűæĽğèąŇè£ŹäÿďäÿłäżżåŁąãĂĆä¿ŃåęĆïijŇèęĄéŹŘåĹűCPUæŮűéŮťïijI

```
import signal
import resource
import os

def time_exceeded(signo, frame):
    print("Time's up!")
    raise SystemExit(1)

def set_max_runtime(seconds):
    # Install the signal handler and set a resource limit
    soft, hard = resource.getrlimit(resource.RLIMIT_CPU)
    resource.setrlimit(resource.RLIMIT_CPU, (seconds, hard))
    signal.signal(signal.SIGXCPU, time_exceeded)

if __name__ == '__main__':
    set_max_runtime(15)
    while True:
        pass
```

çÍŃåžŔè£ŘèąŇæŮűĭijŇSIGXCPU ä£qåŔůåIJÍæŮűéŮť'è£ĞæIJ§æŮűèćńçŤ§æĹŘĭijŇçĎűåŘŐæL'ğèąŇæý

èęĄéŹŘåĹűåĘĚå■Ÿä¡£çŤĺĭijŇèő¿ç¡őåŔŕä¡£çŤĺçŽĎæĂżåĘĚå■ŸåĂijå■şåŔŕĭijŇåęĆäÿŃĭijŽ

```
import resource

def limit_memory(maxsize):
    soft, hard = resource.getrlimit(resource.RLIMIT_AS)
    resource.setrlimit(resource.RLIMIT_AS, (maxsize, hard))
```

åČŔè£Źæăůèő¿ç¡őäžĘåĘĚå■ŸéŹŘåĹűåŘŐĭijŇçÍŃåžŔè£ŘèąŇåĹřæšąæIJL'åd'Žä¡ŹåĘĚå■ŸæŮűäijŽæŁŽ
MemoryError åijĆåÿÿãĂĆ

èőÍèőž

åIJÍæIJňèŁĆä¿Ńå■Řäÿ■ĭijŇsetrlimit() åĞ¡æŢřè¢ńçŤĺæİěèő¿ç¡őçL'źåőŽètĎæžŘäÿŁéİćçŽĎè¡réŹŘ
è¡réŹŘåĹűæŸŕäÿĂäÿłåĂijĭijŇå¡ŞèűĚè£ĞèŁŹäÿłåĂijçŽĎæŮűåĂŹæŞ■ä¡IJçşżçż§éĂŽåÿÿäijŽåŘŚéĂĄäÿĂäÿł
çąňéŹŘåĹűæŸŕçŤĺæİěæŇĞåőŽè¡réŹŘåĹűèČ¡èő¿åőŽçŽĎæIJĂåd'ğåĂijãĂĆéĂŽåÿÿæİěèőšĭijŇè£ŹäÿłçŤśçşż
år¡çőąçąňéŹŘåĹűåŘŕäžěæŤźårŘäÿĂçĆźĭijŇä¡ĘæŸŕæIJĂåě¡äÿ■èęĄä¡£çŤĺçŤĺæĹůè£ŽçĺŃåŐżä£őæŤźãĂĆ

setrlimit() åĞ¡æŢřè£ŸèČ¡èćńçŤĺæİěèő¿ç¡őå■Řè£ŽçĺŃæŢřéĞŘãĂĄæL'ŞåijĂæŰĞäžűæŢřäžěåŘŁç
æŻť'åd'Žèŕęæ ČĚèŕůåŘĆèĂČ resource æĺąåİŮçŽĎæŰĞæąčãĂĆ

éIJĂèęĄæşĺæĎŘçŽĎæŸŕæIJňèŁĆåĘĚåőźåŘłèČ¡éĂĆçŤĺäžŐUnixçşżçż§ĭijŇåźűäÿŤäÿ■ä£İèŕĄæL'ĂæIJL
æŕŤåęĆæĹŚäżňåIJĺæţŃèŕŢçŽĎæŮűåĂŹĭijŇåőĆèČ¡åIJĺLinuxäÿŁéİćæ■čåÿÿè£ŘèąŇĭijŇä¡ĘæŸŕåIJĺOS
XäÿŁå■ť'äÿ■èČ¡ãĂĆ

13.15 åŘŕåŁĺäÿĂäÿłWEBæţŔèğĹåŹĺ

éŮőécŸ

ä¡ăæČşéĂŽè£ĞèĎŽæIJňåŘŕåŁÍæţŔèğĹåŹÍåźűæL'ŞåijĂæŇĞåőŽçŽĎURLç¡Śéąţ

èğčåĘşæŰźæąĹ

webbrowser æÍąåİŮèČ¡èćńçŤĺæİěåŘŕåŁÍäÿĂäÿłæţŔèğĹåŹĺïijŇåźűäÿŤäÿŐåźşåŘřæŮăåĚşãĂĆä¿ŃåęĆ

```
>>> import webbrowser
>>> webbrowser.open('http://www.python.org')
True
>>>
```

åőČäijŽä¡£çŤĺéżŸèőd'æţŔèğĹåŹÍæL'ŞåijĂæŇĞåőŽç¡ŚéąţãĂĆåęĆæđIJä¡ăè£ŸæČşåŕźç¡ŚéąţæL'ŞåijĂæŰ

```
>>> # Open the page in a new browser window
>>> webbrowser.open_new('http://www.python.org')
True
>>>

>>> # Open the page in a new browser tab
>>> webbrowser.open_new_tab('http://www.python.org')
True
>>>
```

è£ŹæăůåřśåŔŕäžěæL'ŞåijĂäÿĂäÿłæŰřçŽĎæţŔèğĹåŹÍçłŮåŔčæĹŰèĂĚæăĞç■¿ïijŇåŔłèęĄæţŔèğĹåŹÍæŤ

åęĆæđIJä¡ăæČşæŇĞåőŽæţŔèğĹåŹÍçśžåđŇïijŇåŔŕäžěä¡£çŤÍ webbrowser.get() åĞ¡æŢřæİěæŇĞåőŽæ§ŘäÿłçL'źåőŽæţŔèğĹåŹÍãĂĆä¿ŃåęĆïijŽ

```
>>> c = webbrowser.get('firefox')
>>> c.open('http://www.python.org')
True
>>> c.open_new_tab('http://docs.python.org')
True
>>>
```

åŕźäžŐæŤŕæŇĄçŽĎæţŔèğĹåŹÍåŘ■çğřåĹŮèąÍåŔŕæ§ěéŸĚ'PythonæŰĞæąč <http://docs.python.org/3/library/webbrowser.html>'_

èőÍèőž

åIJÍèĎŽæIJňäÿ■æL'ŞåijĂæţŔèğĹåŹÍæIJL'æŮűåĂŹäijŽå¿ĹæIJL'çŤÍãĂĆä¿ŃåęĆïijŇæ§ŘäÿłèĎŽæIJňæL'
ä¡ăæČşå£ńéĂ§æL'ŞåijĂäÿĂäÿłæţŔèğĹåŹÍæİěçąőä£İåőČåůšçżŔæ■čåÿÿè£ŘèąŇäžĘãĂĆ
æĹŰèĂĚæŸŕæ§ŘäÿłçÍŇåžŔäžěHTMLç¡ŚéąţæăijåijŔè¿ŞåĞžæŢřæ■őïijŇä¡ăæČşæL'ŞåijĂæţŔèğĹåŹÍæ§ěçIJŇ
äÿ■çőąæŸŕäÿŁéİćåŞłçğ■æČĚåĘţïijŇä¡£çŤÍ webbrowser æÍąåİŮéČ¡æŸŕäÿĂäÿłçőÅå■Ţåőđç̀ŤÍçŽĎèğčåĘşa

çňňå■ĄåŻŻçńäïijŽæţŃèŕŢãĂĄèřČèŕŢåŠŇåijĆåÿÿ

èŕŢéłŇè£ŸæŸŕå¿ĹæčŠçŽĎïijŇä¡ĘæŸŕèřČèŕŢïij§åřśæšąéĆčäźĹæIJL'èűčäžĘãĂĆäžŃåőđæŸŕïijŇåIJĺPyt

Contents:

14.1 æţŃèŕŢstdoutè¿ŞåĞž

éŮőécŸ

ä¡ăçŽĎçÍŃåžŔäÿ■æIJL'äÿłæŰźæşŢäijŽè¿ŞåĞžåĽřæăĞåĞĘè¿ŞåĞžäÿ■ïijĹsys.stdoutïijL'ãĂĆäź§åřśæŸŕè
ä¡ăæČşåĘźäÿłæţŃèŕŢæÌěèŕĄæŸŐåőČïijŇçżŹåőŽäÿĂäÿłè¿ŞåĚěïijŇçŻÿåžŤçŽĎè¿ŞåĞžèČ¡æ■čåÿÿæŸ¿çď'žå

èğčåĘşæŰźæąĹ

ä¡£çŤÍ `unittest.mock` æÍąåİŮäÿ■çŽĎ `patch()` åĞ¡æŢřïijŇ
ä¡£çŤÍètůæİěéİđåÿÿçőĂå■ŢïijŇåŔŕäžěäÿžå■ŢäÿłæţŃèŕŢæÍąæŃ§ `sys.stdout`
çĎűåŘŐåŻđæżŽïijŇ åźűäÿŤäÿ■äžğçŤ§åď'ğéĞŔçŽĎäÿť'æŮűåŔŸéĞŔæĹŰåIJÍæţŃèŕŢçŤÍä¿ŃçŻť'æŐěæŽť'éIJ

ä¡IJäÿžäÿĂäÿłä¿Ńå■ŘïijŇæĹŚäžňåIJÍ `mymodule` æÍąåİŮäÿ■åőŽäźL'åęĆäÿŃäÿĂäÿłåĞ¡æŢřïijŽ

```
# mymodule.py

def urlprint(protocol, host, domain):
    url = '{}://{}.{}'.format(protocol, host, domain)
    print(url)
```

éżŸèőď'æČĚåĘţäÿŃåĘĚç¡őçŽĎ `print` åĞ¡æŢřäijŽåřĘè¿ŞåĞžåŔŚéĂĄåĹř `sys.stdout` ãĂĆ äÿžäžĘæţŃèŕŢè¿ŞåĞžçIJ§çŽĎåIJÍéĆčéĞŇïijŇä¡ääŔŕäžěä¡£çŤÍäÿĂäÿłæŻ£èžńåŕźèśąæİěæÍąæŃ
ä¡£çŤÍ `unittest.mock` æÍąåİŮçŽĎ `patch()` æŰźæşŢåŔŕäžěå¿ĹæŰźä¿£çŽĎåIJÍæţŃèŕŢè£ŘèąŇçŽĎäÿŁ
åźűäÿŤå¡ŞæţŃèŕŢåőŃæĹŘæŮűåĂŹèĞłåŁÍè£ŤåŻđåőČäžňçŽĎåŐ§æIJL'çŁűæĂĄãĂĆäÿŃéÍćæŸŕåŕź
`mymodule` æÍąåİŮçŽĎæţŃèŕŢäžčçăĄïijŽ

```
from io import StringIO
from unittest import TestCase
from unittest.mock import patch
import mymodule

class TestURLPrint(TestCase):
    def test_url_gets_to_stdout(self):
        protocol = 'http'
        host = 'www'
        domain = 'example.com'
        expected_url = '{}://{}.{}\n'.format(protocol, host, domain)

        with patch('sys.stdout', new=StringIO()) as fake_out:
            mymodule.urlprint(protocol, host, domain)
            self.assertEqual(fake_out.getvalue(), expected_url)
```

èőÍèőž

urlprint() åĞ¡æŢřæŐěåŔŮäÿL'äÿłåŔĆæŢřïijŇæţŃèŕŢæŰźæşŢåijĂåğŃäijŽåĚĹèő¿ç¡őæŕŔäÿĂäÿłål
expected_url åŔŸéĞŔèćńèő¿ç¡őæĹŘåŇĚåŘńæIJ§æIJŻçŽĎè¿ŞåĞžçŽĎå■Ůçñęäÿšã ĂĆ

unittest.mock.patch() åĞ¡æŢřèćńçŤĺä¡IJäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺïijŇä¡£çŤĺ
StringIO åŕźèśąæİěäžčæŻ£ sys.stdout . fake_out
åŔŸéĞŔæŸŕåIJÍèŕěè£ŻçÍŃäÿ■èćńåĹŻåžžçŽĎæĺąæŃ§åŕźèśąãĂĆ åIJÍwith-
èŕ■åŘěäÿ■ä¡£çŤĺåőČåŘŕäžěæL'ğèąŇåŘĎçğ■æčĂæ§ěãĂĆå¡Şwithèŕ■åŘęçżŞæİ§æŮűïijŇpatch
äijŽåřĘæL'ĂæIJL'äÿIJèě£æĄćåd'■åĹřæţŃèŕŢåijĂåğŃåL'■çŽĎçŁűæĂĄãĂĆ
æIJL'äÿĂçĆźéIJĂèęĄæşĺæĎŘçŽĎæŸŕæ§ŘäžŽåŕźPythonçŽĎCæL'l'åśŢåŘŕèČ¡äijŽå£¡çŢěæŐL'
sys.stdout çŽĎéĚ■ç¡őäžŇçŻť'æŐěåĘŹåĚěåĹřæăĞåĞĘè¿ŞåĞžäÿ■ãĂĆ
éŹŘäžŐçŕĞåźĚïijŇæIJňèŁĆäÿ■äijŽæűL'åŘŁåĹřě£ŹæŰźéIćçŽĎèőšèğčïijŇåőČéĂĆçŤĺäžŐçžŕPythonäżčçăĄă
åęĆæđIJä¡ăçIJ§çŽĎéIJĂèęĄåIJÍCæL'l'åśŢäÿ■æ■ŢèŐůI/OïijŇä¡ăåŘŕäżěåĚĹæL'Şåij Ăäÿäÿłäÿť'æŮűæŰĞäżű
æŻť'åd'ŽåĚşäžŐæ■ŢèŐůäżěå■Ůçñęäÿšå¡ćåijŘæ■ŢèŐůI/OåŠŇ StringIO
åŕźèśąèŕůåŔĆéŸĚ5.6åřŔèŁĆãĂĆ

14.2 åIJÍå■ŢåĚČæţŃèŕŢäÿ■çżŹåŕźèśąæL'Şèąěäÿą

éŮőécŸ

ä¡ăåĘŹçŽĎå■ŢåĚČæţŃèŕŢäÿ■éIJĂèęĄçżŹæŇĞåőŽçŽĎåŕźèśąæL'Şèąěäÿą¨ïijŇ
çŤĺæİěæŰ■èĺĂåőČäžňåIJÍæţŃèŕŢäÿ■çŽĎæIJ§æIJŻèąŇäÿžïijĹæŕŤåęĆïijŇæŰ■èĺĂèćńèřČçŤĺæŮűçŽĎåŔĆæŢ

èğčåĘşæŰźæąĹ

unittest.mock.patch() åĞ¡æŢřåŔŕèćńçŤĺæİěèğčåĘşè£ŹäÿłéŮőécŸãĂĆ
patch() è£ŸåŔŕèćńçŤĺä¡IJäÿĂäÿłèčĚéěřåŹĺãĂĄäÿŁäÿŃæŰĞçőąçŘĘåŹĺæĹŰå■ŢçŃñä¡£çŤĺïijŇář¡çőąåźűä
ä¿ŃåęĆïijŇäÿŃéIćæŸŕäÿĂäÿłářĘåőČå¡ŞåĄŽèčĚéěřåŹĺä¡£çŤĺçŽĎä¿Ńå■ŘïijŽ

```
from unittest.mock import patch
import example

@patch('example.func')
def test1(x, mock_func):
    example.func(x)       # Uses patched example.func
    mock_func.assert_called_with(x)
```

åőČè£ŸåŔŕäžěèćńå¡ŞåĄŽäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺïijŽ

```
with patch('example.func') as mock_func:
    example.func(x)      # Uses patched example.func
    mock_func.assert_called_with(x)
```

æIJĂåŘŐïijŇä¡ăè£ŸåŔŕäžěæL’ŃåŁĺçŽĎä¡£çŤĺåőČæL’Şèąěäÿ Ąïij Ž

```
p = patch('example.func')
mock_func = p.start()
example.func(x)
mock_func.assert_called_with(x)
p.stop()
```

åęĆæđIJåŘŕèČ¡çŽĎèŕÍïijŇä¡ăèČ¡åd’§åŘäåŁäèčĚéěřåŹĺåŠŇäÿŁäÿŃæŰĞçőąçŘĘåŹĺæÍěçżŹåd’Žäÿłåŕźèś

```
@patch('example.func1')
@patch('example.func2')
@patch('example.func3')
def test1(mock1, mock2, mock3):
    ...

def test2():
    with patch('example.patch1') as mock1, \
         patch('example.patch2') as mock2, \
         patch('example.patch3') as mock3:
    ...
```

èőÍèőž

patch() æŐěåŔŮäÿĂäÿłåůšå■ŸåIJĺåŕźèśąçŽĎåĚÍèůŕå¿ĎåŘ■ïijŇářĘåĚűæŻ£æ■ćäÿžäÿĂäÿłæŰřçŽĎåĂåŐ§æÍěçŽĎåĂijäijŽåIJĺèčĚéěřåŹĺåĞ¡æŢřæĹŰäÿŁäÿŃæŰĞçőąçŘĘåŹĺåőŇæĹŘåŘŐèĞłåŁĺæĄćåd’■åŻđæÍěãéżŸèőd’æČĚåĘţäÿŃïijŇæL’ĂæIJL’åĂijäijŽèćń MagicMock åőđä¿ŃæŻ£äžčãĂĆä¿ŃåęĆïijŽ

```
>>> x = 42
>>> with patch('__main__.x'):
...     print(x)
...
<MagicMock name='x' id='4314230032'>
>>> x
42
>>>
```

äÿ■è£ĞïijŇä¡ăåŔŕäžěéĂŽè£ĞçżŹ patch() æŔŘä¿Żçňňäž ŇäÿłåŔĆæŢřæÍěářĘåĂijæŻ£æ■ćæĹŘäżża¡Ţ

```
>>> x
42
>>> with patch('__main__.x', 'patched_value'):
...     print(x)
...
patched_value
>>> x
42
>>>
```

èćńçŤÍæİěä¡IJäÿžæŻ£æ■ćåĂijçŽĎ `MagicMock` åőđä¿ŃèČ¡åď§æĺąæŃ§åŕŕèřČçŤÍåŕźèśąåŠŇåőđä¿ŃãĂäžŰäžňèőřå¡Ţåŕźèśą
çŽĎä¡£çŤÍä£ąæĄŕåźűåĚĄèőÿä¡ăæĽğèąŇæŰ■èÍĂæčĂæ§ěïijŇä¿ŃåęĆïijŽ

```
>>> from unittest.mock import MagicMock
>>> m = MagicMock(return_value = 10)
>>> m(1, 2, debug=True)
10
>>> m.assert_called_with(1, 2, debug=True)
>>> m.assert_called_with(1, 2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File ".../unittest/mock.py", line 726, in assert_called_with
    raise AssertionError(msg)
AssertionError: Expected call: mock(1, 2)
Actual call: mock(1, 2, debug=True)
>>>

>>> m.upper.return_value = 'HELLO'
>>> m.upper('hello')
'HELLO'
>>> assert m.upper.called

>>> m.split.return_value = ['hello', 'world']
>>> m.split('hello world')
['hello', 'world']
>>> m.split.assert_called_with('hello world')
>>>

>>> m['blah']
<MagicMock name='mock.__getitem__()' id='4314412048'>
>>> m.__getitem__.called
True
>>> m.__getitem__.assert_called_with('blah')
>>>
```

äÿĂèĹňæİěèőšïijŇè£ŹäžŻæŞ■ä¡IJäijŽåIJĺäÿĂäÿłå■ŢåĚČæţŃèŕŢäÿ■åőŇæĹŘãĂĆä¿ŃåęĆïijŇåĄĞèő¿ä¡ä

```
# example.py
from urllib.request import urlopen
import csv

def dowprices():
    u = urlopen('http://finance.yahoo.com/d/quotes.csv?s=@^DJI&f=sl1
↪')
    lines = (line.decode('utf-8') for line in u)
    rows = (row for row in csv.reader(lines) if len(row) == 2)
    prices = { name:float(price) for name, price in rows }
    return prices
```

æ■čåÿÿæİěèőšïijŇè£ŹäÿłåĞ¡æŢřäijŽä¡£çŤÍ `urlopen()` äžŐWe-
bäÿŁéÍćèŐůåŔŰæŢřæ■őåźűèğčædŘåőČãĂĆ åIJĺå■ŢåĚČæţŃèŕŢäÿ■ïijŇä¡ääŔŕäžěçżŹåőČäÿĂäÿłécĎåĚĹåőŽ

```
import unittest
from unittest.mock import patch
import io
import example

sample_data = io.BytesIO(b'''\
"IBM",91.1\r
"AA",13.25\r
"MSFT",27.72\r
\r
''')

class Tests(unittest.TestCase):
    @patch('example.urlopen', return_value=sample_data)
    def test_dowprices(self, mock_urlopen):
        p = example.dowprices()
        self.assertTrue(mock_urlopen.called)
        self.assertEqual(p,
                         {'IBM': 91.1,
                          'AA': 13.25,
                          'MSFT' : 27.72})

if __name__ == '__main__':
    unittest.main()
```

æIJñä¿Ńäÿ■ïijÑä¡■äžŐ example æĺąåİŮäÿ■çŽĎ urlopen() åĞ¡æŢřèćńäÿĂäÿłæĺąæŃ§åŕźèśąæŻ£äžčïijŃ èŕěåŕźèśąäijŽè£ŤåŻđäÿĂäÿłåŇĚåŘńæţŃèŕŢæŢřæ■őçŽĎ ByteIO().

è£ŸæIJL'äÿĂçĆźïijÑåIJĺæL'Şèąěäÿ ĄæŮűæĹŚäžňä¡£çŤĺäžĘ example.urlopen æİěäžčæŻ£ urllib.request.urlopen ãĂĆ å¡Şä¡ääĹŻåžžèąěäÿĄçŽĎæŮűåĂŹïijÑä¡ää£ĚéążäĄ£çŤĺåőČäžňåIJĺæţŃèŕŢäžčçăĄäÿ■çŽĎåŘ■çğřãĂĆ çŤśäžŐæţŃèŕŢäžčçăĄä¡£çŤĺäžĘ from urllib.request import urlopen ,éĆčäźĹ dowprices() åĞ¡æŢř äÿ■ä¡£çŤĺçŽĎ urlopen() åĞ¡æŢřåőđéŹĚäÿŁářśä¡■äžŐ example æĺąåİŮäžĘãĂĆ

æIJñèŁĆåőđéŹĚäÿŁåŘłæŸŕářź unittest.mock æĺąåİŮçŽĎäÿĂæňąæţĚářİè¿Ďæ■ćãĂĆ æŻťåď ŽæŻťéńŸçžğçŽĎçL'źæĂğïijÑèŕůåŘĆèĂČ åőŸæŰźæŰĞæąč

14.3 åIJĺå■ŢåĚČæţŃèŕŢäÿ■æţŃèŕŢåijĆåÿÿæČĚåĘţ

éŮőécŸ

ä¡ăæČşåĘŹäÿłæţŃèŕŢçŤĺä¿ŃæİěåĞĘçąőçŽĎåĹď æŰ■æ§ŘäÿłåijĆåÿÿæŸŕåŘęèćńæŁŻåĞžãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐåijĆåÿÿçŽĎæţŃèŕŢåŔŕä¡£çŤĺ `assertRaises()` æŰźæşŢãĂĆä¿ŃåęĆïijŇåęĆæđIJä¡ăæČşæţŃèŕŢæ§ŘäÿłåĞ¡æŢřæŁŻåĞžäžĘ `ValueError` åijĆåÿÿïijŇåČŔäÿŃéİćè£ŹæăůåĘŹïijŽ

```
import unittest

# A simple function to illustrate
def parse_int(s):
    return int(s)

class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        self.assertRaises(ValueError, parse_int, 'N/A')
```

åęĆæđIJä¡ăæČşæţŃèŕŢåijĆåÿÿçŽĎåĚůä¡ŞåĂijïijŇéIJĂèęĄçŤĺåĹřåŔęåď Űäÿ Ăçğ■æŰźæşŢïijŽ

```
import errno

class TestIO(unittest.TestCase):
    def test_file_not_found(self):
        try:
            f = open('/file/not/found')
        except IOError as e:
            self.assertEqual(e.errno, errno.ENOENT)

        else:
            self.fail('IOError not raised')
```

èŐÍèŐž

`assertRaises()` æŰźæşŢäÿžæţŃèŕŢåijĆåÿÿå■ŸåIJÍæĂğæŔŘä¿ŻäžĘäÿĂäÿłçŐĂä¿£æŰźæşŢãĂĆäÿĂäÿłåÿÿèğĄçŽĎéŹůéŸśæŸŕæĽŃåŁĺåŐżè£ŻèąŇåijĆåÿÿæčĂæţŃãĂĆæŕŤåęĆïijŽ

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        try:
            r = parse_int('N/A')
        except ValueError as e:
            self.assertEqual(type(e), ValueError)
```

è£Źçğ■æŰźæşŢçŽĎéŮőécŸåIJÍäžŐåőČå¿Ĺåőźæ ŸŞéĄŮæijŔåĚűäżŰæČĚåĘţïijŇæŕŤåęĆæšąæIJL’äżżä¡éĆčäźĹä¡ăè£Ÿå¿ŮéIJĂèęĄåćđåŁääŔęåď ŰçŽĎæčĂæţŃè£ĞçÍŇïijŇåęĆäÿŃéİćè£ŹæăůïijŽ

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        try:
            r = parse_int('N/A')
```

```
        except ValueError as e:
            self.assertEqual(type(e), ValueError)
        else:
            self.fail('ValueError not raised')
```

assertRaises() æŰźæşŢäijŽåď'ĎçŘĘæĽ'ĂæIJĽçżĘèŁĆïijŇåŻăæ■d'ä¡ăåžŤèŕěä¡£çŤĺåőČãĂĆ

assertRaises() çŽĎäÿĂäÿłçijžçĆźæŸŕåőČætŃäÿ■äžĘåijĆåÿÿåĚůä¡ŞçŽĎåĂijæŸŕåď'ŽåŕŠãĂĆäÿžäžĘætŃèŕŢåijĆåÿÿåĂijïijŇåŔŕäžěä¡£çŤĺ assertRaisesRegex() æŰźæşŢïijŇåőČåŔŕåŘŇæŮűætŃèŕŢåijĆåÿÿçŽĎå■ŸåIJĺäžěåŔŁéĂŽè£Ğæ■čåĹŹåijŔåŇźéĚ■åijĆåÿÿçŽĎå■Ůçňęäÿšèąĺçd

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        self.assertRaisesRegex(ValueError, 'invalid literal .*',
                               parse_int, 'N/A')
```

assertRaises() åŠŇ assertRaisesRegex() è£ŸæIJĽäÿĂäÿłåőźæŸŞå£¡çŢěçŽĎåIJřæŰźåřśæŸŕåőČäžňè£ŸèČ¡èćńå¡ŞåĄŽäÿŁäÿŃæŰĞçőąçŘĘåŹĺä¡£çŤĺ

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        with self.assertRaisesRegex(ValueError, 'invalid literal .*
↪'):
            r = parse_int('N/A')
```

ä¡Ęä¡ăçŽĎætŃèŕŢæűL'åŔŁåĹřåď'Žäÿłæ L'ğèąŇæ■ěéłď'çŽĎæŮűåĂŹè£Źçğ■æŰźæşŢåřśå¿ĹæIJĽçŤĺäžĘ

14.4 åřĘætŃèŕŢè¿ŞåĞžçŤĺæŮěå£Ùèőřå¡ŢåĹřæŰĞäžűäÿ■

éŮőécŸ

ä¡ăåÿŇæIJŻåřĘå■ŢåĚČætŃèŕŢçŽĎè¿ŞåĞžåĘŹåĹřåĹřæ§ŘäÿłæŰĞäžűäÿ■åŐżïijŇèĂŇäÿ■æŸŕæL'Şå■řå

èğčåĘşæŰźæąĹ

è£ŘèąŇå■ŢåĚČætŃèŕŢäÿĂäÿłåÿÿèğĄæŁĂæIJŕåřśæŸŕåIJĺætŃèŕŢæŰĞäžűåžŢéČĺåŁăåĚěäÿŃéİćè£Źæőţ

```
import unittest

class MyTest(unittest.TestCase):
    pass

if __name__ == '__main__':
    unittest.main()
```

è£ŹæăůçŽĎèŕİætŃèŕŢæŰĞäžűåřśæŸŕåŔŕæL'ğèąŇçŽĎïijŇåźűäÿŤäijŽåřĘè£ŘèąŇætŃèŕŢçŽĎçżŞædIJæ
åęĆædIJä¡ăæČşéĞ■åőŽåŘŠè¿ŞåĞžïijŇåřśéIJĂèęĄåČŔäÿŃéİćè£Źæăůä£őæŤź main()
åĞ¡æŢřïijŽ

```
import sys

def main(out=sys.stderr, verbosity=2):
    loader = unittest.TestLoader()
    suite = loader.loadTestsFromModule(sys.modules[__name__])
    unittest.TextTestRunner(out,verbosity=verbosity).run(suite)

if __name__ == '__main__':
    with open('testing.out', 'w') as f:
        main(f)
```

èőÍèőž

æIJňèŁĆæĎ§åĚť'èűčçŽĎéČĺåĹĘåźűäÿ■æŸŕåřĘæţŃèŕŢçżŞæđIJéĞ■åőŽåŘŚåĹřäÿĂäÿłæŰĞäžűäÿ■ïijŇ
èĂŇæŸŕéĂŽè£Ğè£ŹæăůåĄŽåŘŚä¡ăåśŢçď'žäžĘ unittest
æÍąåİŮäÿ■äÿĂäžŽåĂijå¿ŮåĚşæşÍçŽĎåĘĚéČÍåůĕä¡IJåŐ§çŘĘãĂĆ

unittest æÍąåİŮéęŰåĚĹäijŽçżĎèčĔäÿĂäÿłæţŃèŕŢåĕŮäžűãĂĆ
è£ŹäÿłæţŃèŕŢåĕŮäžűåŇĚåŘńäžĘä¡ăåőŽäźL'çŽĎåŘĎçğ■æŰźæşŢãĂĆäÿĂæŮęåĕŮäžűçżĎèčĔåőŇæĹŘïijŇå

è£Źäÿďæ■ěæŸŕåĹĘåijĂçŽĎïijŇunittest.TestLoader
åőđä¿ŃèćńçŤĺæİěçżĎèčĔæţŃèŕŢåĕŮäžűãĂĆ loadTestsFromModule()
æŸŕåőČåőŽäźL'çŽĎæŰźæşŢäźŃäÿĂïijŇçŤĺæİěæŤűéŽĘæţŃèŕŢçŤĺä¿ŃãĂĆ åőČäijŽäÿž
TestCase çśžæĽ'ńæŘŕĂæ§Řäÿłæĺąå İŮåźűåřĘåĔűäÿ■çŽĎæţŃèŕŢæŰźæşŢæŘŕåŘŰåĞžæİěãĂĆ
åęĆæđIJä¡ăæČşè£ŽèąŇçżĘçšŠåžęçŽĎæŐğåĹűïijŇ åŘŕäžěä¡£çŤĺ
loadTestsFromTestCase() æŰźæşŢæİěäžŐæ§ŘäÿłçžğæĽ'£TestCaseçŽĎçśžäÿ■æŘŕåŘŰæţŃèŕŢæŰźæ
TextTestRunner çśžæŸŕäÿĂäÿłæţŃèŕŢè£ŘèąŇçśžçŽĎä¿Ńå■ŘïijŇ
è£ŹäÿłçśžçŽĎäÿžèęĄçŤĺéĂŤæŸŕæĽ'ğèąŇæ§ŘäÿłæţŃèŕŢåĕŮäžűäÿ■åŇĚåŘńçŽĎæţŃèŕŢæŰźæşŢãĂĆ
è£Źäÿłçśžèů§æĽ'ğèąŇ unittest.main() åĞ¡æŢřæĽ'Ăä¡£çŤĺçŽĎæţŃèŕŢè£ŘèąŇåŹĺæŸŕäÿĂæăůçŽĎãĂ
äÿ■è£ĞïijŇæĹŚäžňåIJĺè£ŹéĞŇåŕźåőČè£ŻèąŇäžĘäÿĂäžŽåĹŮåžŢåśĆéĚ■ç¡őïijŇåŇĚæŃňè¿ŞåĞžæŰĞäžűåŠ
ař¡çőąæIJňèŁĆä¿Ńå■ŘäžčçăĄå¿ĹåřŚïijŇä¡ĘæŸŕèČ¡æŇĞåŕijä¡ăåęĆä¡Ţåŕź
unittest æąĘæđűè£ŻèąŇæŻť'è£ŻäÿĂæ■ěçŽĎèĞłåőŽäźL'ãĂĆ
èęĄæČşèĞłåőŽäźL'æţŃèŕŢåĕŮäžűçŽĎèčĔéĚ■æŰźåijŕïijŇä¡ăåŘŕäžěåŕź TestLoader
çśžæL'ğèąŇæŻť'åď'ŽçŽĎæŞ■ä¡IJãĂĆ äÿžäžĘèĞłåőŽäźL'æţŃèŕŢè£ŘèąŇïijŇä¡ăåŘŕäžěæđĎéĂääÿĂäÿłèĞłåů
TextTestRunner çŽĎåŁ§èČ¡ãĂĆ èĂŇè£ŹäžŽåůšçżŘèűĚåĞžäžĘæIJňèŁĆçŽĎèŇČåŻť'ãĂĆunittest
æÍąåİŮçŽĎæŰĞæąčåŕźåžŢåśĆåőđçŐŕåŐ§çŘĘæIJL'æŻť'æůśåĚěçŽĎèőšèğčïijŇåŘŕäžěåŐžçIJŃçIJŃãĂĆ

14.5 å£¡çŢěæĹŰæIJ§æIJŻæţŃèŕŢåď'śèť'ě

éŮőécŸ

ä¡ăæČşåIJĺå■ŢåĚČæţŃèŕŢäÿ■å£¡çŢěæĹŰæăĞèőŕæ§ŘäžŻæţŃèŕŢäijŽæŇL'çĚğécĎæIJ§è£ŘèąŇåď'śèť'ěã

èğčåĘşæŰźæąĹ

unittest æÍąåİŮæIJL'èčĔéěŕåŹĺåŘŕçŤĺæİěæŐğåĹűåŕźæŇĞåőŽæţŃèŕŢæŰźæşŢçŽĎåď'ĎçŘĘïijŇä¿Ń

```
import unittest
import os
import platform

class Tests(unittest.TestCase):
    def test_0(self):
        self.assertTrue(True)

    @unittest.skip('skipped test')
    def test_1(self):
        self.fail('should have failed!')

    @unittest.skipIf(os.name=='posix', 'Not supported on Unix')
    def test_2(self):
        import winreg

    @unittest.skipUnless(platform.system() == 'Darwin', 'Mac␣
↪specific test')
    def test_3(self):
        self.assertTrue(True)

    @unittest.expectedFailure
    def test_4(self):
        self.assertEqual(2+2, 5)

if __name__ == '__main__':
    unittest.main()
```

åęĆædIJä¡ăåIJÍMacäÿŁè£ŘèąŇè£Źæőţäżčçă Ąïij Ňä¡ăäijŽå¿ŮåĹřåęĆäÿŃè¿ŞåĞžïijŽ

```
bash % python3 testsample.py -v
test_0 (__main__.Tests) ... ok
test_1 (__main__.Tests) ... skipped 'skipped test'
test_2 (__main__.Tests) ... skipped 'Not supported on Unix'
test_3 (__main__.Tests) ... ok
test_4 (__main__.Tests) ... expected failure

----------------------------------------------------------------------
↪--
Ran 5 tests in 0.002s

OK (skipped=2, expected failures=1)
```

èőÍèőž

`skip()` èčĚéěřåŹÍèČ¡èćńçŤÍæİěå£¡çŢěæ§Řäÿłä¡ăäÿ■æČşè£ŘèąŇçŽĎæţŃèŕŢãĂĆ `skipIf()` åŠŇ `skipUnless()` åŕźäžŐä¡ăåŘłæČşåIJÍæ§ŘäÿłçĽźáőŽåźşåŘřæĹŰPythonçĽĹæIJňæĹŰåĚű ä¡£çŤÍ `@expected` çŽĎåď śèť ěèčĚéěřåŹÍæİěæĂĞèőřéĆčäžŻçąőåőŽäijŽåď śèť ěçŽĎæţŃèŕŢïijŇåźűäÿŤåŕźè£

å£¡çŢěæŰźæşŢçŽĎèčĚéěřåŹÍè£ŸåŔŕäžèèćńçŤÍæİěèčĚéěřæŢt'äÿłæţŃèŕŢçśžïijŇæŕŤåęĆïijŽ

```
@unittest.skipUnless(platform.system() == 'Darwin', 'Mac specific␣
↪tests')
class DarwinTests(unittest.TestCase):
    pass
```

14.6 åď ĎçŘĘåď Žäÿłåij Ćåÿÿ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłäžčçăĄçL'Ğæőţå Ŕŕè Č¡äijŽæŁŻåĞžåď Žäÿłäÿ■åŘŇçŽĎåijĆåÿÿïijŇæĂŐæăůæL'■èČ¡äÿ■

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăåŔŕäžěçŤÍå■Ţäÿłäžčçă ĄåİŮåď ĎçŘĘäÿ■åŘŇçŽĎåijĆåÿÿïijŇåŔŕäžěårĘåőČäžňæŤ¿åĚěäÿĂä

```
try:
    client_obj.get_url(url)
except (URLError, ValueError, SocketTimeout):
    client_obj.remove_url(url)
```

åIJÍè£Źäÿłä¿Ńå■Řäÿ■ïijŇåĚČçěŰäÿ■äžžä¡ŢäÿĂäÿłåijĆåÿÿåŔŞçŤ§æŮűéČ¡äijŽæL'ğèąŇ
`remove_url()` æŰźæşŢãĂĆ åęĆædIJä¡ăæČşåŕźåĚűäÿ■æ§Řäÿłåij Ćåÿÿè£ŻèąŇäÿ■åŘŇçŽĎåď ĎçŘĘïijŇåŕ
`except` èŕ■åŔěäÿ■ïijŽ

```
try:
    client_obj.get_url(url)
except (URLError, ValueError):
    client_obj.remove_url(url)
except SocketTimeout:
    client_obj.handle_url_timeout(url)
```

å¿Ĺåď ŽçŽĎåijĆåÿÿäijŽæIJL'åśĆçžğåĚşçşżïijŇåŕźäžŐè£Źçğ■æČĚåĘţïijŇä¡ăåŔŕèČ¡ä¡£çŤĺåőČäžňçŽĎäÿ

```
try:
    f = open(filename)
except (FileNotFoundError, PermissionError):
    pass
```

åŔŕäžěèćńéĞ■åĘŹäÿžïijŽ

```
try:
    f = open(filename)
except OSError:
    pass
```

OSError æŸí FileNotFoundError åŠŇ PermissionError åijĆåÿÿçŽĎå§žçśżãĂĆ

èőÍèőž

åř¡çőąåď ĎçŘĘåď Žäÿłåij Ćåÿÿæ IJňèžńåźűæšąäžĂäźĹçL'źæőŁçŽĎïijŇäÿ■è£Ğä¡ääŔŕäžěä¡£çŤÍ as åĚşéŤőå■ŮæÍěèŐůå¿Ůèćńæ ŁŻåĞžåijĆåÿÿçŽĎåijŢçŤĺïijŽ

```
try:
    f = open(filename)
except OSError as e:
    if e.errno == errno.ENOENT:
        logger.error('File not found')
    elif e.errno == errno.EACCES:
        logger.error('Permission denied')
    else:
        logger.error('Unexpected error: %d', e.errno)
```

è£Źäÿłä¿Ńå■Řäÿ■ïijŇ e åŔŸéĞŔæŇĞåŘŠäÿĂäÿłèćńæŁŻåĞžçŽĎ OSError åijĆåÿÿåőđä¿ŃãĂĆ è£Źäÿłå IJĺä¡äæČşæŽť è£ŻäÿĂæ■ěåĹĘæđŘè£Źäÿłåij Ćåÿÿç ŽĎæŮűůåĂŹäijŽå¿ĹæIJL'çŤĺï

åŘŇæŮűè£Ÿè ęĄæşĺæĎŔçŽĎæŮűůåĂŹ except èí■åŔěæŸŕéąžåžŘæčĂæ§ěçŽĎïijŇçňňäÿĂäÿłåŇźéĚ■ç ä¡ääŔŕäžěå¿Ĺåőźæ ŸŞçŽĎæđĎéĂåđ'Žäÿł except åŘŇæŮűåŇźéĚ■çŽĎæČĚå¡ćïijŇærŤåęĆïijŽ

```
>>> f = open('missing')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: 'missing'
>>> try:
...     f = open('missing')
... except OSError:
...     print('It failed')
... except FileNotFoundError:
...     print('File not found')
...
It failed
>>>
```

è£ŹéĞŇçŽĎ FileNotFoundError èí■åŔěåźűæšąæIJL'æL'ğèąŇçŽĎåŐ§åŻăæŸŕ OSError æŽť äÿĂèĹňïijŇåőČåŔŕåŇźéĚ■ FileNotFoundError åijĆåÿÿïijŇ äžŐæŸŕåřśæŸŕçňňäÿĂäÿłåŇźéĚ■çŽĎãĂĆ åIJĺèřČèŕŢçŽĎæŮűůåĂŹïijŇąęĆæđIJä¡ääŕźæ§ŘäÿłçL'źåőŽåijĆåÿÿ ä¡ääŔŕäžěéĂŽè£Ğæ§ěçIJŃèřěåijĆåÿÿçŽĎ __mro__ åśđæĂğæÍěå£ńéĂ§æţŔèğĹãĂĆæŕŤåęĆïijŽ

```
>>> FileNotFoundError.__mro__
(<class 'FileNotFoundError'>, <class 'OSError'>, <class 'Exception'>
 ↪,
 <class 'BaseException'>, <class 'object'>)
>>>
```

äÿŁéĺćåĹŮèąĺäÿ■äżżä¡ŢäÿĂäÿłçŽť åĹř BaseException çŽĎçśžéČ¡èČ¡èćńçŤĺäžŐ

except èŕ■åŔěãĂĆ

14.7 æ■ŢèŐůæĽĂælJĽåijĆåÿÿ

éŮőécŸ

æĂŐæăůæ■ŢèŐůäžčçăĄäÿ■çŽĎæĽĂæIJĽåijĆåÿÿïij§

èğčåĘşæŰźæąĹ

æČşèęĄæ■ŢèŐůæĽĂæIJĽçŽĎåijĆåÿÿïijŇåŔŕäžěçŻťæŐěæ■ŢèŐů Exception å■şåŔŕïijŽ

```
try:
    ...
except Exception as e:
    ...
    log('Reason:', e)       # Important!
```

è£ŹäÿłåřĘäijŽæ■ŢèŐůéŹď'äžĘ SystemExit ãĂĄ KeyboardInterrupt åŠŇ GeneratorExit äźŇåď'ŰçŽĎæĽĂæIJĽåijĆåÿÿãĂĆ åęĆæđIJä¡ăè£ŸæČşæ■ŢèŐůè£ŹäÿL'äÿłåijĆåÿÿïijŇåřĘ Exception æŤźæĹŘ BaseException å■şåŔŕãĂĆ

èőÍèőž

æ■ŢèŐůæĽĂæIJĽåijĆåÿÿéĂŽåÿÿæŸŕçŤśäžŐçÍŇåžŔåŚŸåIJÍæ§ŘäžŻåď'■æİĆæŞ■ä¡IJäÿ■åźűäÿ■èČ¡èőř åęĆæđIJä¡ăäÿ■æŸŕå¿ĹçżĘå£ČçŽĎäžžïijŇè£Źäź§æŸŕçijŰåĘŹäÿ■æŸŞèřČèŕŢäžčçăĄçŽĎäÿĂäÿłçőĂå■ŢæŰź

æ■čåŽăåęĆæ■d'ïijŇåęĆæđIJä¡ăéĂĽ'æŃľæ■ŢèŐůæĽĂæIJĽåijĆåÿÿïijŇéĆčäźĹåIJÍæ§ŘäÿłåIJřæŰźïijÍæ åęĆæđIJä¡ăæšąæIJĽè£ŹæăůåĄŽïijŇæIJĽæŮűåĂŹä¡ăçIJŃåĹřåijĆåÿÿæĽŞå■řæŮűåŔŕèČ¡æŚÿäÿ■çİĂåď'èĎ

```
def parse_int(s):
    try:
        n = int(v)
    except Exception:
        print("Couldn't parse")
```

èŕŢçİĂè£ŘèąŇè£ŹäÿłåĞ¡æŢřïijŇçżŞæđIJåęĆäÿŃïijŽ

```
>>> parse_int('n/a')
Couldn't parse
>>> parse_int('42')
Couldn't parse
>>>
```

è£ŹæŮűåĂŹä¡ăářšäijŽæŇăåd'ť'æČşïijŽâĂIJè£ŹåŠŃåŻđäžŃåŢŁïij§âĂİ
åĄĞåęĆä¡ăåČŔäÿŃéİćè£ŹæăůéĞ■åĘŹè£ŹäÿłåĞ¡æŢřïijŽ

```
def parse_int(s):
    try:
        n = int(v)
    except Exception as e:
        print("Couldn't parse")
        print('Reason:', e)
```

è£ŹæŮűåĂŹä¡ăèČ¡èŐůåŔŰåęĆäÿŃè¿ŞåĞžïijŇæŇĞæŸŐäžĘæIJL'äÿłçijŰçÍŃéŤŹèŕřïijŽ

```
>>> parse_int('42')
Couldn't parse
Reason: global name 'v' is not defined
>>>
```

å¿ŁæŸŐæŸ¿ïijŇä¡ăåžŤèŕěář¡åŔŕèČ¡ářĘåijĆåÿÿåd'ĎçŘĘåŹĺåőŽäźL'çŽĎçš¿åĞĘäÿĂäžŻãĂĆ
äÿ■è£ĞïijŇèęĄæŸŕä¡ăå£Ěéążæ■ŢèŐůæL'ĂæIJL'åijĆåÿÿïijŇçąőä£ÍæL'Şå■ŕæ■čçąőçŽĎèŕŁæŰ■ä£ąæĄŕæŁŰå

14.8 åĹŻåżžèĞłåőŽäźL'åijĆåÿÿ

éŮőécŸ

åIJĺä¡ăædĎåżžçŽĎåžŤçŤĺçÍŃåžŔäÿ■ïijŇä¡ăæČşářĘåžŢåśĆåijĆåÿÿåŇĚèčĚæĹŘèĞłåőŽäźL'çŽĎåijĆåÿÿ

èğčåĘşæŰźæąĹ

åĹŻåżžæŰřçŽĎåijĆåÿÿå¿ŁçőĂå■ŢâĂŤâĂŤåőŽäźL'æŰřçŽĎçśżïijŇèől'åőČçżğæL'£èĞł
`Exception` ïijĹæĹŰèĂĚæŸŕäżżä¡ŢäÿĂäÿłåůšå■ŸåIJĺçŽĎåijĆåÿÿçśżåđŃïijL'ãĂĆ
ä¿ŃåęĆïijŇåęĆædIJä¡ăçijŰåĘŹç¡ŚçżIJçŽÿåĚşçŽĎçÍŃåžŔïijŇä¡ăåŔŕèČ¡äijŽåőŽäźL'äÿĂäžŻçśżäijijåęĆäÿŃçŽ

```
class NetworkError(Exception):
    pass

class HostnameError(NetworkError):
    pass

class TimeoutError(NetworkError):
    pass

class ProtocolError(NetworkError):
    pass
```

çĎűåŘŐçŤĺæĹůářšåŔŕäżěåČŔéĂŽåÿÿéĆčæăůä¡£çŤĺè£ŹäžŻåijĆåÿÿäžĘïijŇä¿ŃåęĆïijŽ

```
try:
    msg = s.recv()
```

```
except TimeoutError as e:
    ...
except ProtocolError as e:
    ...
```

èőÍèőž

èĞłåőŽäźL’åijĆåÿÿçśżåžŤèŕěæĂżæŸŕçżğæL’£èĞłåĘĚç¡őçŽĎ Exception çśżïijŇ æĹŰèĂĚæŸŕçżğæL’£èĞłéĆčäžŻæIJňèžńåřśæŸŕäżŐ Exception çżğæL’£èĂŇæİěçŽĎçśżãĂĆ åř¡çőąæL’ĂæIJL’çśżåŘŇæŮűäź§çżğæL’£èĞł BaseException ïijŇä¡Ęä¡ääÿ■åžŤèŕěä¡£çŤÍè£Źäÿłå§žçśżæİěåőŽäźL’æŰřçŽĎåijĆåÿÿãĂĆ BaseException æŸŕäÿžçşżçż§éĂĂåĞžåijĆåÿÿèĂŇä£İçŢŹçŽĎïijŇæŕŤåęĆ KeyboardInterrupt æĹŰ SystemExit äžěåŔŁåĚűäžŰéĆčäžŻäijŽçżŹåžŤçŤÍåŔŚéĂĄä£ąåŔůèĂŇéĂĂåĞžçŽĎåijĆåÿÿãĂĆ åŽäæ■ď’ïijŇæ■ŢèŐůè£ŹäžŽåijĆåÿÿæIJňèžńæšąäžĂäźĹæĎŔäźL’ãĂĆ è£ŹæăůçŽĎèŕİïijŇåĄĞåęĆä¡ăçżğæL’£ BaseException åŔŕèČ¡äijŽåŕijèĞť ä¡ăçŽĎèĞłåőŽäźL’åijĆåÿÿäÿ■ä

åIJÍçÍŃåžŔäÿ■åijŢåĚěèĞłåőŽäźL’åijĆåÿÿåŔŕäžěä¡£å¿Ůä¡ăçŽĎäżčçăĄæŻť’åĚůåŔŕèŕżæĂğïijŇèČ¡æÿĚæŽ è£ŸæIJL’äÿĂçğ■èő¿èőąæŸŕårĘèĞłåőŽäźL’åijĆåÿÿéĂŽè£ĞçżğæL’£çżĎåŘĹèţůæİěãĂĆåIJĺåď’■æİĆåžŤçŤÍçÍŃ ä¡£çŤÍå§žçśżæİěåĹĘçżĎåŘĎçğ■åijĆåÿÿçśżäź§æŸŕå¿ĹæIJL’çŤÍçŽĎãĂĆåőČåŔŕäžěèőľ’çŤÍæĹůæ■ŢèŐůäÿĂä

```
try:
    s.send(msg)
except ProtocolError:
    ...
```

ä¡ăè£ŸèČ¡æ■ŢèŐůæŽť’åď’ğèŇČåŽť’çŽĎåijĆåÿÿïijŇåřśåČŔäÿŃéİćè£ŹæăůïijŽ

```
try:
    s.send(msg)
except NetworkError:
    ...
```

åęĆæđIJä¡ăæČşåőŽäźL’çŽĎæŰřåijĆåÿÿéĞ■åĘŹäžĘ __init__() æŰźæşŢïijŇ çąőä£İä¡ăä¡£çŤÍæL’ĂæIJL’åŔĆæŢřèřČçŤÍ Exception.__init__() ïijŇä¿ŃåęĆïijŽ

```
class CustomError(Exception):
    def __init__(self, message, status):
        super().__init__(message, status)
        self.message = message
        self.status = status
```

çIJŃäÿŁåŐżæIJL’çĆźåěĞæĂłïijŇäÿ■è£ĞExceptionçŽĎéžŸèőď’èąŃäÿžæŸŕæŐěåŔŮæL’ĂæIJL’äijăéĂŠç .args åšđæĂğäÿ■. å¿Ĺåď’ŽåĚűäžŰåĞ¡æŢřåžŞåŠŃéČĺåĹĘPythonåžŞéżŸèőď’æL’ĂæIJL’åijĆåÿÿéČ¡å£Ěéąża .args åšđæĂğïijŇ åŽäæ■ď’åęĆæđIJä¡ăå£¡çŢëäžĘè£ŹäÿĂæ■ëïijŇä¡ăäijŽåŔŞçŐřæIJL’äžŽæŮűåĂŹä¡ăåőŽäź äÿžäžĘæijŤçd’ž .args çŽĎä¡£çŤÍïijŇèĂČèŹŠäÿŃäÿŃéİćè£ŹäÿłäçŢÍåĘĚç¡őçŽĎ *RunTimeError‘* åijĆåÿÿçŽĎäžď’äžŠäijŽèŕİïijŇ æşłæĎŔçIJŃraiseèŕ■åŘëäÿ■ä¡£çŤÍçŽĎåŔĆæŢřäÿłæŢřæŸŕæĂŐæă

```
>>> try:
...     raise RuntimeError('It failed')
```

```
... except RuntimeError as e:
...     print(e.args)
...
('It failed',)
>>> try:
...     raise RuntimeError('It failed', 42, 'spam')
... except RuntimeError as e:

...     print(e.args)
...
('It failed', 42, 'spam')
>>>
```

åËşäžŐåĹŻåżžèĞłåőŽäźL'åijĆåÿÿçŽĎæŻt'åd'Žä£ąæĄŕïijŇèŕůåŔĆèĂČ'PythonåőŸæŰźæŰĞæąč <https://docs.python.org/3/tutorial/errors.html>'_

14.9 æ■ŢèŐůåijĆåÿÿåŘŐæŁŻåĞžåŔęåd'ŰçŽĎåijĆåÿÿ

éŮőécŸ

ä¡ăæČşæ■ŢèŐůäÿĂäÿłåijĆåÿÿåŘŐæŁŻåĞžåŔęåd'ŰäÿĂäÿłäÿ■åŘŇçŽĎåijĆåÿÿïijŇåŘŇæŮűè£Ÿå¿ŮåIJ

èğčåĘşæŰźæąĹ

äÿžäžĘéŞ¿æŐěåijĆåÿÿïijŇä¡£çŤÍ `raise from` èŕ■åŔěæİěäžčæŻ£çőÅå■ŢçŽĎ `raise èŕ■åŔěãĂĆ` åőČäijŽèől'ä¡ääŘŇæŮűä£İçŢŹäÿd'äÿłåijĆåÿÿçŽĎä£ąæĄŕãĂĆä¿ŃåęĆïijŽ

```
>>> def example():
...     try:
...             int('N/A')
...     except ValueError as e:
...             raise RuntimeError('A parsing error occurred') from␣
 ↪e
...
>>> example()
Traceback (most recent call last):
  File "<stdin>", line 3, in example
ValueError: invalid literal for int() with base 10: 'N/A'
```

äÿŁéİćçŽĎåijĆåÿÿæŸŕäÿŃéİćçŽĎåijĆåÿÿäžğçŤ§çŽĎçŻt'æŐěåŐ§åŻäïijŽ

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in example
RuntimeError: A parsing error occurred
>>>
```

åIJÍåŻđæžŕäÿ■åŔŕäžěçIJŃåĹřïijŇäÿď äÿłåijĆåÿÿéČ¡èćńæ■ŢèŐůãĂĆ èęĄæČşæ■ŢèŐůè£Źæăůç ŽĎåijĆåÿÿïijŇä¡ăåŔŕäžěä¡£çŤĺäÿĂäÿłçőĂå■ŢçŽĎ except èŕ■åŔěãĂĆ äÿ■è£ĞïijŇä¡ăè£ŸåŔŕäžěéĂŽè£Ğæ§ěçIJŃåijĆåÿÿåŕźèśąçŽĎ __cause__ åśđæĂğæİěèů§èÿłåijĆåÿÿéŞ¿ãĂĆä¿ŃåęĆïijŽ

```
try:
    example()
except RuntimeError as e:
    print("It didn't work:", e)

    if e.__cause__:
        print('Cause:', e.__cause__)
```

å¡ŞåIJÍ except åİŮäÿ■åŔĹæIJL'åŔęåď ŰçŽĎåijĆåÿÿèćńæŁŻåĞžæŮűäijŽåŕijèĞť äÿĂäÿłéŽŘèŮŔçŽĎå

```
>>> def example2():
...     try:
...             int('N/A')
...     except ValueError as e:
...             print("Couldn't parse:", err)
...
>>>
>>> example2()
Traceback (most recent call last):
  File "<stdin>", line 3, in example2
ValueError: invalid literal for int() with base 10: 'N/A'
```

åIJÍåď ĎçŘĘäÿŁè£řåijĆåÿÿçŽĎæŮűåĂŹïijŇåŔęåď ŰäÿĂäÿłåijĆåÿÿåŔŠçŤ§äžĘïijŽ

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in example2
NameError: global name 'err' is not defined
>>>
```

è£ŹäÿłäżŃå■Řäÿ■ïijŇä¡ăåŘŇæŮűèŐůå¿ŮäžĘäÿď äÿłåijĆåÿÿçŽĎä£ąæĄŕïijŇä¡ĘæŸŕåŕźåijĆåÿÿçŽĎèğč è£ŹæŮűåĂŹïijŇNameError åijĆåÿÿèćńä¡IJäÿžçÍŃåžŔæIJĂçżĹåijĆåÿÿèćńæŁŻåĞžïijŇèĂŇäÿ■æŸŕä¡■äžŐ

åęĆædIJïijŇä¡ăæČşå£¡çŢěæŐL'åijĆåÿÿéŞ¿ïijŇåŔŕä¡£çŤÍ raise from None :

```
>>> def example3():
...     try:
...             int('N/A')
...     except ValueError:
...             raise RuntimeError('A parsing error occurred') from␣
 ↪None
...
>>>
example3()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in example3
```

```
RuntimeError: A parsing error occurred
>>>
```

èőÍèőž

åIJÍèő¿èőąäžčçăĄæŮűïijŇåIJÍåŘęåď'ŰäÿĂäÿł except äžčçăĄåİŮäÿ■ä¡£çŤÍ raise èŕ■åŘěçŽĎæŮűåĂŹä¡ăèęĄçL'źåĹńářŘå£ČäžĘãĂĆ åď'gåď'ŽæŢřæČĚåĘţäÿŃïijŇè£Źçğ■ raise èŕ■åŘěéČ¡åžŤèŕěèćńæŤźæĹŘ raise from èŕ■åŘěãĂĆäź§åřśæŸŕèŕť'ä¡ääžŤèŕěä¡£çŤÍäÿŃéİćè£ŹçğĄ

```
try:
    ...
except SomeException as e:
    raise DifferentException() from e
```

è£ŹæăůåĄžçŽĎåŐ§åŻăæŸŕä¡ääžŤèŕěæŸ¿çď'žçŽĎåřĘåŐ§åŻăéŞ¿æŐěèţůæİěãĂĆ äź§åřśæŸŕèŕť'ïijŇDifferentException æŸŕçŻť'æŐěäžŐ SomeException èą■çŤ§èĂŇæİěãĂĆ è£Źçğ■åĚşçşżåŘŕäžěäžŐåŻđæžŕçżŞæđIJäÿ■çIJŃåĞžæİěãĂĆ

åęĆæđIJä¡ääČŔäÿŃéİćè£ŹæăůåĘŹäžčçăĄïijŇä¡ääž■çĎűäijŽå¿ŮåĹřäÿĂäÿłéŞ¿æŐěåijĆåÿÿïijŇ äÿ■è£Ğè£ŹäÿłåźűæšąæIJL'å¿ĹæÿĚæŹŕçŽĎèŕť'æŸŐè£ŹäÿłåijĆåÿÿéŞ¿åĹřåžŢæŸŕåĘĚéČĺåijĆåÿÿè£ŸæŸŕæ§

```
try:
    ...
except SomeException:
    raise DifferentException()
```

å¡Şä¡ää¡£çŤÍ raise from èŕ■åŘěçŽĎèŕİïijŇåŕšå¿ĹæÿĚæěŽçŽĎèąĺæŸŐæŁŻåĞžçŽĎæŸŕçňňäžŇäÿłå

æIJĂåŘŐäÿĂäÿłä¿Ňå■Řäÿ■éŽŘèŮŔåijĆåÿÿéŞ¿ä£ąæĄŕãĂĆ åř¡çőąéŽŘèŮŔåijĆåÿÿéŞ¿ä£ąæĄŕäÿ■åĹl'äžŐåŻđæžŕïijŇåŘŇæŮűåőČäź§äÿćåď'śäžĘå¿Ĺåď'ŽæIJL'çŤÍçŽĎèŕČ äÿ■è£ĞäÿĞäžŇçŽĘáźşç■L'ïijŇæIJL'æŮűåĂŹåŘłä£İçŢŹéĂĆå¡ŞçŽĎä£ąæĄŕäź§æŸŕå¿ĹæIJL'çŤÍçŽĎãĂĆ

14.10 éĞ■æŰřæŁŻåĞžèćńæ■ŢèŐůçŽĎåijĆåÿÿ

éŮőécŸ

ä¡ääIJÍäÿĂäÿł except åİŮäÿ■æ■ŢèŐůäžĘäÿĂäÿłåijĆåÿÿïijŇçŐřåIJÍæČşéĞ■æŰřæŁŻåĞžåőČãĂĆ

èğčåĘşæŰźæąĹ

çőĂå■ŢçŽĎä¡£çŤÍäÿĂäÿłå■ŢçŃňçŽĎ rasie èŕ■åŘěå■şåŘŕïijŇä¿ŃåęĆïijŽ

```
>>> def example():
...     try:
...             int('N/A')
...     except ValueError:
```

```
...             print("Didn't work")
...             raise
...

>>> example()
Didn't work
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 3, in example
ValueError: invalid literal for int() with base 10: 'N/A'
>>>
```

èőÍèőž

è£ŹäÿłéŮőéćŸéĂŽåÿÿæŸŕå¡Şä¡ăéIJĂèęĄåIJÍæ■ŢèŐůåijĆåÿÿåŘŐæĽğèąŇæ§Řäÿłæş■ä¡IJïijĹæŕŤåęĆèčäÿĂäÿłå¿Ĺåÿÿè ğĄçŽĎçŤÍæşŢæŸŕåIJÍæ■ŢèŐůæĽĂæIJĽåijĆåÿÿçŽĎåď ĎçŘĘåŹÍäÿ■ïijŽ

```
try:
    ...
except Exception as e:
    # Process exception information in some way
    ...

    # Propagate the exception
    raise
```

14.11 è¿ŞåĞžè■ęåŚĽä£ąæĄŕ

éŮőéćŸ

ä¡ăåÿŇæIJŻèĞłåůśçŽĎçÍŃåžŔèČ¡çŤ§æĹŘè■ęåŚĽä£ąæĄŕïijĹæŕŤåęĆåž§åijČçĽźæĂğæĹŰä¡£çŤÍéŮőéć

èğčåĘşæŰźæąĹ

èęĄè¿ŞåĞžäÿĂäÿłè■ęåŚĽæűĹæĄŕïijŇåŔŕä¡£çŤÍ warning.warn() åĞ¡æŢřãĂĆä¿ŃåęĆïijŽ

```
import warnings

def func(x, y, logfile=None, debug=False):
    if logfile is not None:
        warnings.warn('logfile argument deprecated',␣
↪DeprecationWarning)
    ...
```

warn() çŽĎåŔĆæŢřæŸŕäÿĂäÿłè■ęåŚŁæűĹæĄŕåŠŇäÿĂäÿłè■ęåŚŁçśżiijŇè■ęåŚŁçśżæIJL'åęĆäÿŃåĞäć DeprecationWarning, SyntaxWarning, RuntimeWarning, ResourceWarning, æĹŰ FutureWarning.

åŕźè■ęåŚçŽĎåď'ĎçŘĘåŔŰåĘşäžŐä¡ăåęĆä¡Ţè£ŘèąŇèğčéĞŁåŹĺäżěåŔŁäÿĂäžŻåĚűäžŰéÉ■ç¡őãĂĆ ä¿ŃåęĆïijŇåęĆæđIJä¡ää¡£çŤĺ -W all éĂL'éąźåŐżè£ŘèąŇPythonïijŇä¡ääijŽå¿ŮåĹřåęĆäÿŃçŽĎè¿ŞåĞžïijŽ

```
bash % python3 -W all example.py
example.py:5: DeprecationWarning: logfile argument is deprecated
  warnings.warn('logfile argument is deprecated',␣
↪DeprecationWarning)
```

éĂŽåÿÿæÌěèőšïijŇè■ęåŚŁäijŽè¿ŞåĞžåĹřæăĞåĞĘéŤŹèŕŕäÿŁãĂĆåęĆæđIJä¡ăæČşèőšè■ęåŚŁè¡ñæ■ćäÿžå -W error éĂL'éąźïijŽ

```
bash % python3 -W error example.py
Traceback (most recent call last):
  File "example.py", line 10, in <module>
    func(2, 3, logfile='log.txt')
  File "example.py", line 5, in func
    warnings.warn('logfile argument is deprecated',␣
↪DeprecationWarning)
DeprecationWarning: logfile argument is deprecated
bash %
```

èőÍèőž

åIJÍä¡ăçżt'æŁd'è¡ŕäžűïijŇæŔŘçd'žçŤĺæĹůæ§ŘäžŻä£ąæĄŕïijŇä¡ĘæŸŕåŔĹäÿ■éIJĂèęĄåřĘåĚűäÿŁå■ĞäÿŽ ä¿ŃåęĆïijŇåĄĞèő¿ä¡ăåĞĘåď'Ğä£őæŤźæ§ŘäÿłåĞ¡æŢřåžŞæĹŰæąĘæđűçŽĎåŁ§èČ¡ïijŇä¡ääŔŕäžěåĚĹäÿžä¡ăč ä¡ăè£ŸåŔŕäžěè■ęåŚŁçŤĺæĹůäÿĂäžŻåŕźäžčçăĄæIJL'éŮőéćŸçŽĎä¡£çŤĺæŰźåijŔãĂĆ

ä¡IJäÿžåŘęåď'ŰäÿĂäÿłåĘĚç¡őåĞ¡æŢřåžŞçŽĎè■ęåŚŁä¡£çŤĺä¿Ńå■ŘïijŇäÿŃéÍćæijŢçď'žäžĘäÿĂäÿłæšąæ

```
>>> import warnings
>>> warnings.simplefilter('always')
>>> f = open('/etc/passwd')
>>> del f
__main__:1: ResourceWarning: unclosed file <_io.TextIOWrapper name=
↪'/etc/passwd'
 mode='r' encoding='UTF-8'>
>>>
```

éżŸèőď'æČĚåĘţäÿŃïijŇåźűäÿ■æŸŕæL'ĂæIJL'è■ęåŚŁæűĹæĄŕéČ¡äijŽåĞžçŐřãĂĆ-W éĂL'éąźèČ¡æŐğåĹűè■ęåŚŁæűĹæĄŕçŽĎè¿ŞåĞžãĂĆ -W all äijŽè¿ŞåĞžæL'ĂæIJL'è■ęåŚŁæűĹæĄŕïijŇ-W ignore å£¡çŢěæŐL'æL'ĂæIJL'è■ęåŚŁïijŇ-W error åřĘè■ęåŚŁè¡ñæ■ćæĹŘåijĆåÿÿãĂĆ åŘęåď'ŰäÿĂçğ■éĂL'æŃl'ïijŇä¡äè£ŸåŔŕäžěä¡£çŤĺ warnings.simplefilter() åĞ¡æŢřæŐğåĹűè¿ŞåĞžãĂĆ always åŔĆæŢřäijŽèől'æL'ĂæIJL'è■ęåŚŁæűĹæĄŕåĞžçŐřïijŇ` ignore å£¡çŢěèřČæL'ĂæIJL'çŽĎè■ęåŚŁïijŇerror åřĘè■ęåŚŁè¡ñæ■ćæĹŘåijĆåÿÿãĂĆ

åŕźäžŐçőĂå■ŢçŽĎçŤ§æĹŘè■ęåŚŁæűĹæĄŕçŽĎæČĚåĘţè£ŹäžŻåůšçżŔèűşåď§äžĘãĂĆ warnings æĺąåİŮåŕźè£Ğæžď åŠŇè■ęåŚŁæűĹæĄŕåď ĎçŘĘæŔŘä¿ŻäžĘåď ğéĞŔçŽĎæŻť énŸçžğçŽĎéĚ■ç¡ æŻť åď Žä£ąæĄŕèŕůåŔĆèĂČ PythonæŰĞæąč

14.12 èřČèŕŢå§žæIJňçŽĎçÍŃåžŔåťľæžČéŤŹèŕŕ

éŮőécŸ

ä¡ăçŽĎçÍŃåžŔåťľæžČåŘŐèŕěæĂŐæăůåŐżèřČèŕŢåőČïij§

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăçŽĎçÍŃåžŔåŻääÿžæ§ŘäÿłåijĆåÿÿèĂŇåťľæžČïijŇè£ŘèąŇ python3 -i someprogram.py åŔŕæĽğèąŇçőĂå■ŢçŽĎèřČèŕŢãĂĆ -i éĂĽéąźåŔŕèőľçÍŃåžŔçżŞæİ§åŘŐæĽŞåijĂäÿĂäÿłäžď äžŠåijŔshellãĂĆ çĎűåŘŐä¡ääřśèČ¡æ§ěçIJŇçŐŕåćČïijŇä¿ŃåęĆïijŇåĄĞèő¿ä¡ăæIJĽäÿŃéİćçŽĎäžčçăĄïijŽ

```
# sample.py

def func(n):
    return n + 10

func('Hello')
```

è£ŘèąŇ python3 -i sample.py äijŽæIJĽçśžäijijåęĆäÿŇçŽĎè¿ŞåĞžïijŽ

```
bash % python3 -i sample.py
Traceback (most recent call last):
  File "sample.py", line 6, in <module>
    func('Hello')
  File "sample.py", line 4, in func
    return n + 10
TypeError: Can't convert 'int' object to str implicitly
>>> func(10)
20
>>>
```

åęĆædIJä¡ăçIJŃäÿ■åĹřäŸŁéİćè£ŹæăůçŽĎïijŇåŔŕäžěåIJÍçÍŃåžŔåťľæžČåŘŐæĽŞåijĂPythonçŽĎèřČèŕŢ

```
>>> import pdb
>>> pdb.pm()
> sample.py(4)func()
-> return n + 10
(Pdb) w
  sample.py(6)<module>()
-> func('Hello')
> sample.py(4)func()
-> return n + 10
```

```
(Pdb) print n
'Hello'
(Pdb) q
>>>
```

åęĆæđIJä¡ăçŽĎäžčçăĄæĽĂåIJÍçŽĎçŐŕåćČå¿ĹéŽ¿èŐůåŔŰäžd'äžŠshellïijĹæŕŤåęĆåIJÍæ§Řäÿłæ IJ■åŁą
éĂŽåÿÿåŔŕäžěæ■ŢèŐůåijĆåÿÿåŘŐèĞłåůśæĽŞå■řèů§èÿłä£ąæĄŕãĂĆä¿ŃåęĆïijŽ

```
import traceback
import sys

try:
    func(arg)
except:
    print('**** AN ERROR OCCURRED ****')
    traceback.print_exc(file=sys.stderr)
```

èęĄæŸŕä¡ăçŽĎçÍŃåžŔæšąæIJL'åť l'æžČïijŇèĂŇåŔłæŸŕäžğçŤ§äžĘäÿĂäžŻä¡ăçIJŃäÿ■æĞĆçŽĎçżŞædI
ä¡ăåIJÍæĎ§åĚť èűčçŽĎåIJřæŰźæŘŠåĚěäÿĂäÿŃ `print()` èí■åŘěäź§æŸŕäÿłäÿ■éŤŹçŽĎéĂL'æŃl'ãĂĆ
äÿ■è£ĞïijŇèęĄæŸŕä¡ăæL'ŞçőŮè£ŹæăůåĄŽïijŇæIJL'äÿĂäžŻářŔæŁĂåůğåŔŕäžěåÿőåŁl'ä¡ããĂĆ
éęŰåĚĹïijŇ`traceback.print_stack()` åĞ¡æŢřäijŽä¡ăçÍŃåžŔè£ŘèąŇåĹřéĆčäÿłçĆźçŽĎæŮűåĂŹåĹŻ

```
>>> def sample(n):
...     if n > 0:
...             sample(n-1)
...     else:
...             traceback.print_stack(file=sys.stderr)
...
>>> sample(5)
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 3, in sample
  File "<stdin>", line 3, in sample
  File "<stdin>", line 3, in sample
  File "<stdin>", line 3, in sample
  File "<stdin>", line 3, in sample
  File "<stdin>", line 5, in sample
>>>
```

åŔęåď'ŰïijŇä¡ăè£ŸåŔŕäžěåČŔäÿŃéÍćè£ŹæăůäĄçŤÍ `pdb.set_trace()`
åIJÍäżżä¡ŢåIJřæŰźæL'ŃåŁÍçŽĎåŘŕåŁÍèřČèŕŢåŹÍïijŽ

```
import pdb

def func(arg):
    ...
    pdb.set_trace()
    ...
```

å¡ŞçÍŃåžŔæŕŤè¿Čåď'ğèĂŇä¡ăæČşèřČèŕŢæŐğåĹűæţĄçÍŃäžěåŔŁåĞ¡æŢřåŔĆæŢřçŽĎæŮűåĂŹè£Źäÿłåiäż¿ŃåęĆïijŇäÿĂæŮęèřČèŕŢåŹÍåijĂåğŃè£ŘèąŇïijŇä¡ăåřśèČ¡åď§ä¡£çŤÍ

print æİěèğĆætŃåŘŸéĞŘåĂijæĹŰæŢšåĞżæ§ŘäÿłåŚ¡äžď æŕŤåęĆ w æİěèŐůåŘŰè£¡èÿłä£ąæĄŕãĂĆ

èőÍèőž

äÿ■èęĄåřĘèřČèŕŢåijĎçŽĎè£ĞäžŐåď■æİĆåŃŰãĂĆäÿĂäžŻçőĂå■ŢçŽĎéŤŹèŕŕåŘłéĲĂèęĄèğĆåŕ§çĺŃå
åőđéŹĚçŽĎéŤŹèŕŕäÿĂèĹňæŸŕååĘæăĹçŽĎæĲĂåŘŐäÿĂèąŃãĂĆ
ä¡ăåĲĺåijĂåŘŚçŽĎæŮűåĂŹïijŇäź§åŘŕäžěåĲĺä¡ăéĲĂèęĄèřČèŕŢçŽĎåĲřæŰźæŘŠåĚěäÿĂäÿŃ
print() åĞ¡æŢřæİěèřŁæŰ■ä£ąæĄŕïijĹåŘłéĲĂèęĄæĲĂåŘŐåŘŚäÿČçŽĎæŮűåĂŹåĹăéŹď è£ŹäžŻæĽŞå■

èřČèŕŢåŹĺçŽĎäÿĂäÿłåÿÿèğĄçŤĺæşŢæŸŕèğĆætŃæ§ŘäÿłåůšçżŔåť ľ æžČçŽĎåĞ¡æŢřäÿ■çŽĎåŘŸéĞŘãĂ
ç§ěéĄŞæĂŐæăůåĲĺåĞ¡æŢřåť ľ æžČåŘŐè£ŻåĚěèřČèŕŢåŹĺæŸŕäÿĂäÿłå¿ĹæĲĽçŤĺçŽĎæŁĂèČ¡ãĂĆ

å¡Şä¡ăæČşèğčåĽ ŰäÿĂäÿłéİđåÿÿåď■æİĆçŽĎçĺŃåžŔïijŇåžŢåśĆçŽĎæŐğåĹűéĂżè¿Śä¡ääÿ■æŸŕå¿ĹæÿĚ
æŔŠåĚě pdb.set_trace() è£ŹæăůçŽĎèŕ■åŔěåřśå¿ĹæĲĽçŤĺäžĘãĂĆ

åőđéŹĚäÿŁïijŇçĺŃåžŔäijŽäÿĂçŻť è£Řèą ŃåĹřçćřåĹř set_trace()
èŕ■åŔěä¡■ç¡őïijŇçĎűåŘŐçńŃéľ ňè£ŻåĚěèřČèŕŢåŹĺãĂĆ çĎűåŘŐä¡ääřśåŔŕäžěåĄŽæŻť åď ŽçŽĎäžŃäžĘãĂĆ

åęĆæđIJä¡ää¡£çŤĺIDEæİěåĄŽPythonåijĂåŘŚïijŇéĂŽåÿÿIDEéČ¡äijŽæŘŘä¿ŻèĞłåůšçŽĎèřČèŕŢåŹĺæİěå
æŻť åď Žè£ŹæŰźéİćçŽĎä£ąæĄŕåŔŕäžěåŔĆèĂČä¡ää¡£çŤĺçŽĎIDEæĽŃåĘŃãĂĆ

14.13 çżŹä¡ăçŽĎçĺŃåžŔåĄŽæĂğèČ¡æţŃèŕŢ

éŮőéćŸ

ä¡ăæČşætŃèŕŢä¡ăçŽĎçĺŃåžŔè£Řèą ŃæĽ ĂèŁśèť źçŽĎæŮűéŮť åźűåĄŽæĂğèČ¡æţŃèŕŢãĂĆ

èğčåĘşæŰźæąĹ

åęĆæđIJä¡ăåŔłæŸŕçőĂå■ŢçŽĎæČşætŃèŕŢäÿŃä¡ăçŽĎçĺŃåžŔæŢť ä¡ŞèŁśèť źçŽĎæŮűéŮť ïijŇ
éĂŽåÿÿä¡£çŤĺUnixæŮűéŮť åĞ¡æŢřåřśèąŃäžĘïijŇæŕŤåęĆïijŽ

```
bash % time python3 someprogram.py
real 0m13.937s
user 0m12.162s
sys  0m0.098s
bash %
```

åęĆæđIJä¡ăè£ŸéĲĂèęĄäÿĂäÿłçĺŃåžŔåŘĎäÿłçżĘèŁĆçŽĎèŕęçżĘæŁěåŚŁïijŇåŔŕäžěä¡£çŤĺ
cProfile æĺąåİŮïijŽ

```
bash % python3 -m cProfile someprogram.py
         859647 function calls in 16.016 CPU seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall␣
↪filename:lineno(function)
```

```
   263169    0.080    0.000    0.080    0.000 someprogram.
↪py:16(frange)
      513    0.001    0.000    0.002    0.000 someprogram.
↪py:30(generate_mandel)
   262656    0.194    0.000   15.295    0.000 someprogram.py:32(
↪<genexpr>)
        1    0.036    0.036   16.077   16.077 someprogram.py:4(
↪<module>)
   262144   15.021    0.000   15.021    0.000 someprogram.py:4(in_
↪mandelbrot)
        1    0.000    0.000    0.000    0.000 os.py:746(urandom)
        1    0.000    0.000    0.000    0.000 png.py:1056(_readable)
        1    0.000    0.000    0.000    0.000 png.py:1073(Reader)
        1    0.227    0.227    0.438    0.438 png.py:163(<module>)
      512    0.010    0.000    0.010    0.000 png.py:200(group)
    ...
bash %
```

äÿ■è£ĞéĂŽåÿÿæČĚåĘţæŸŕäżŃäžŐè£Źäÿď äÿłæđĄçńŕäźŃéŮťˇãĂĆæŕŤåęĆä¡ääůšçżŔç§ěéĄŞäżčçăĄè£Ł
åŕźäžŐè£ŹäžŽåĞ¡æŢřçŽĎæĂğèČ¡æţŃèŕŢïijŇåŔŕäżěä¡£çŤĺäÿĂäÿłçőĂå■ŢçŽĎèčĚéěřåŹĺïijŽ

```
# timethis.py

import time
from functools import wraps

def timethis(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.perf_counter()
        r = func(*args, **kwargs)
        end = time.perf_counter()
        print('{}.{} : {}'.format(func.__module__, func.__name__,␣
↪end - start))
        return r
    return wrapper
```

èęĄä¡£çŤĺè£ŹäÿłèčĚéěřåŹĺïijŇåŔŕéIJĂèęĄåřĘåĚűæŤ¿ç¡őåIJĺä¡ăèęĄè£Żèą̈ŇæĂğèČ¡æţŃèŕŢçŽĎåĞ¡æŢ

```
>>> @timethis
... def countdown(n):
...     while n > 0:
...             n -= 1
...
>>> countdown(10000000)
__main__.countdown : 0.803001880645752
>>>
```

èęĄæţŃèŕŢæ§Řäÿłäżčçă ĄåİŮè£ŘèąŇæŮűéŮťˇïijŇä¡ăåŔŕäżěåőŽäźĽäÿĂäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺïijŇ

```
from contextlib import contextmanager

@contextmanager
def timeblock(label):
    start = time.perf_counter()
    try:
        yield
    finally:
        end = time.perf_counter()
        print('{} : {}'.format(label, end - start))
```

äÿŃéİćæŸŕä¡£çŤĺè£ŹäÿłäÿŁäÿŃæŰĞçőąçŘĘåŹĺçŽĎä¿Ńå■ŘïijŽ

```
>>> with timeblock('counting'):
...     n = 10000000
...     while n > 0:
...             n -= 1
...
counting : 1.5551159381866455
>>>
```

åŕźäžŐæţŃèŕŢå¿ĹåřŔçŽĎäžčçăĄçL'Ğæőţè£ŘèąŇæĂğèČ¡ïijŇä¡£çŤÍ timeit æĺąåİŮäijŽå¿ĹæŰźä¿£ïijŇä¿ŃåęĆïijŽ

```
>>> from timeit import timeit
>>> timeit('math.sqrt(2)', 'import math')
0.1432319980012835
>>> timeit('sqrt(2)', 'from math import sqrt')
0.10836604500218527
>>>
```

timeit äijŽæL'ğèąŇçňňäÿĂäÿłåŔĆæŢřäÿ■èŕ■åŘě100äÿĞæňąåźűèőąçőŮè£ŘèąŇæŮűéŮt'ãĂĆ çňňäžŇäÿłåŔĆæŢřæŸŕè£ŘèąŇæţŃèŕŢäźŃåL'■éĚ■ç¡őçŐŕåćČãĂĆåęĆæđIJä¡ăæČşæŤźåŔŸå¿łçŐŕæL'ğèąŇæň åŘŕäžěåČŔäÿŃéİćè£Źæăůèő¿ç¡ő number åŔĆæŢřçŽĎåĂijïijŽ

```
>>> timeit('math.sqrt(2)', 'import math', number=10000000)
1.434852126003534
>>> timeit('sqrt(2)', 'from math import sqrt', number=10000000)
1.0270336690009572
>>>
```

èőÍèőž

å¡ŞæL'ğèąŇæĂğèČ¡æţŃèŕŢçŽĎæŮűåĂŹïijŇéIJĂèęĄæşÍæĎŔçŽĎæŸŕä¡äèŐůåŔŰçŽĎçżŞæđIJéČ¡æŸŕè time.perf_counter() åĞ¡æŢřäijŽåIJÍçżŹåőŽåźşåŘŕäÿŁèŐůåŔŰæIJĂéńŸçš¿åžęçŽĎèőąæŮűåĂijãĂĆ äÿ■è£ĞïijŇåőČäż■çĎűè£ŸæŸŕå§žäžŐæŮűéŠ§æŮűéŮt'ïijŇå¿Ĺåd'ŽåŻăçt'ääijŽå¡šå§■åĹřåőČçŽĎçš¿çąőåžęï åęĆæđIJä¡åårźäžŐæL'ğèąŇæŮűéŮt'æŻt'æĎ§åĚt'èűčïijŇä¡£çŤÍ time.process_time() æİěäžčæŻ£åőČãĂĆä¿ŃåęĆïijŽ

```
from functools import wraps
def timethis(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.process_time()
        r = func(*args, **kwargs)
        end = time.process_time()
        print('{}.{} : {}'.format(func.__module__, func.__name__,␣
↪end - start))
        return r
    return wrapper
```

æIJÄåŘŐïijŇåęĆæđIJä¡ăæČşè£ŻèąŇæŻť æůśåĚěçŽĎæĂğèČ¡åĹĘæđŘïijŇéĆčäźĹä¡ăéIJĂèęĄèŕęçżĘéŸĖ
time ãĂĄtimeit åŠŇåĚűäżŰçŻÿåĚşæĺąåİŮçŽĎæŰĞæąčãĂĆ
è£Źæăůä¡ăåŔŕäżěçŘĘèğčåŠŇåźşåŔřçŻÿåĚşçŽĎåůőåijĆäżěåŔŁäÿĂäžŻåĚűäżŰéŹůéŸśãĂĆ
è£ŸåŔŕäżěåŔĆèĂČ13.13åřŔèŁĆäÿ■çŻÿåĚşçŽĎäÿĂäÿłåĹŻåżžèőąæŮűåŹĺçśżçŽĎä¿Ńå■ŘãĂĆ

14.14 åŁăéĂ§çĺŃåžŔè£ŘèąŇ

éŮőécŸ

ä¡ăçŽĎçĺŃåžŔè£ŘèąŇåd'łæĚćïijŇä¡ăæČşåIJĺäÿ■ä¡£çŤĺåd'■æİĆæŁĂæIJŕæŕŤåęĆCæĽľ'åśŢæĹŰJITçijŰ

èğčåĘşæŰźæąĹ

åĚşäžŐçĺŃåžŔäijŸåŇŰçŽĎçňňäÿĂäÿłåĞĘåĹŹæŸŕâĂIJäÿ■èęĄäijŸåŇŰâĂİïijŇçňňäžŇäÿłåĞĘåĹŹæŸŕ
åęĆæđIJä¡ăçŽĎçĺŃåžŔè£ŘèąŇçijŞæĚćïijŇéęŰåĚĹä¡ăå¿Ůä¡£çŤĺ14.13åřŔèŁĆçŽĎæŁĂæIJŕåĚĹåŕźåőČè£Żèą

éĂŽåÿÿæİěèőšä¡ăäijŽåŔŞçŐřä¡ăå¿ŮçĺŃåžŔåIJĺåřŚæŢřåĞääÿłçČ■çĆźåIJřæŰźèŁśèť źäžĘåd'ğéĞŔæŮűé
ærŤåęĆåĘĚå■ŸçŽĎæŢřæ■őåd'ĎçŘĘå¿łçŐŕãĂĆäÿĂæŮęä¡ăåőŽä¡■åĹrè£ŹäžŻçĆźïijŇä¡ăåřšåŔŕäżěä¡£çŤĺäÿŇ

ä¡£çŤĺåĞ¡æŢř

å¿Ĺåd'ŽçĺŃåžŔåŚŸåĹŽåijĂåğŇäijŽä¡£çŤĺPythonèŕ■èĺĂåĘŹäÿĂäžŻçőĂå■ŢèĎŽæIJñãĂĆ
å¡ŞçijŰåĘŹèĎŽæIJñçŽĎæŮűåĂŹïijŇéĂŽåÿÿäźăæČŕäžĘåĘŹæŕńæŮăçżŞæđĎçŽĎäžčçăĄïijŇæŕŤåęĆïijŽ

```
# somescript.py

import sys
import csv

with open(sys.argv[1]) as f:
     for row in csv.reader(f):

          # Some kind of processing
          pass
```

å¿ĹåřŚæIJL’äžžç§ěéĄŞïijŇåČŕè£Źæăůåő Žäź L’åIJĺåĚĺåśĂèŇČåŻt’çŽĎäžčçăĄè£ŘèąŇètůæİěèęĄæŕŤåő
è£Źçğ■éĂ§åžęåůőåijĆæŸŕçŤśäžŐåśĂéČĺåŔŸéĞŕåŠŇåĚĺåśĂåŔŸéĞŕçŽĎåőđçŐřæŰźåijŕïijĹä¡£çŤĺåśĂéČĺ
åŻäæ■d’ïijŇåęĆæđIJä¡äæČşèől’çÍŃåžŕè£ŘèąŇæŻt’å£ńäžŻïijŇåŕłéIJĂèęĄåřĘèĎŽæIJňèŕ■åŔěæŤ¿åĚěåĞ¡æŢ

```
# somescript.py
import sys
import csv

def main(filename):
    with open(filename) as f:
         for row in csv.reader(f):
             # Some kind of processing
             pass

main(sys.argv[1])
```

éĂ§åžęçŽĎåůőåijĆåŔŰåĘşäžŐåőđéŹĚè£ŘèąŇçŽĎçÍŃåžŕïijŇäÿ■è£Ğæăźæ■őçżŔéłŇïijŇä¡£çŤĺåĞ¡æŢï
30%çŽĎæĂğèČ¡æŔŘå■ĞæŸŕå¿Ĺåÿÿèğ ĄçŽĎãĂĆ

ař¡åŕŕèČ¡åŐżæŐL’åśđæĂğèő£éŮő

æŕŔäÿĂæňąä¡£çŤĺçĆź(.)æŞ■ä¡IJçňęæİěèő£éŮőåśđæĂğçŽĎæŮűåĂŹäijŽåÿęæİěéćİåd’ŰçŽĎåijĂéŤĂãĂ
åőČäijŽèğęåŔŚçL’źåőŽçŽĎæŰźæşŢïijŇærŤåęĆ __getattribute__() åŠŇ
__getattr__() ïijŇè£ŹäžŻæŰźæşŢäijŽè£ŻèąŇå■ŮåĚÿæŞ■ä¡IJæŞ■ä¡IJãĂĆ

éĂŽåÿÿä¡ääŔŕäżěä¡£çŤĺ `from module import name`
è£ŹæăůçŽĎåŕijåĚěå¡ćåijŕïijŇäžěåŔŁä¡£çŤĺçżŚåőŽçŽĎæŰźæşŢãĂĆ
åĄĞèő¿ä¡äæIJL’ąęĆäÿŃçŽĎäžčçăĄçL’Ğæőţïijž

```
import math

def compute_roots(nums):
    result = []
    for n in nums:
        result.append(math.sqrt(n))
    return result

# Test
nums = range(1000000)
for n in range(100):
    r = compute_roots(nums)
```

åIJĺæĹŚäžňæIJžåŹĺäÿŁéIćætŃèŕŢçŽĎæŮűåĂŹïijŇè£ŹäÿłçĺŃåžŔèŁśèť źäžĘåd’ğæęĆ40çğŠãĂĆçŐřåIJĺa
`compute_roots()` åĞ¡æŢřąęĆäÿŃïijž

```
from math import sqrt

def compute_roots(nums):

    result = []
    result_append = result.append
    for n in nums:
```

```
        result_append(sqrt(n))
    return result
```

ä£őæŤźåŘŐçŽĎçL'ĹæIJňè£ŘèąŇæŮűéŮť'åď'ğæęĆæŸŕ29çğŠãĂĆåŤŕäÿĂäÿ■åŘŇäźŃåď'ĎåřśæŸŕæűĹéź
çŤĺ `sqrt()` äžčæŻ£äžĘ `math.sqrt()` ãĂĆ `The result.append()` æŰźæşŢèćńètŇçżŹäÿĂäÿłåśĂéČĺåŔŸéĞŔ `result_append` ïijŇçĎűåŘŐåIJĺåĘĚéČĺå¿łçŐŕäÿ■ä¡£çŤĺåőČãĂĆ

äÿ■è£ĞïijŇè£ŹäžŻæŤźåŔŸåŔłæIJL'åIJĺåď'ğéĞŔéĞ■åď'■äžčçăĄäÿ■æL'■æIJL'æĎŔäźL'ïijŇærŤåęĆå¿łçĆ
åŻäæ■ď'ïijŇè£ŹäžŻäijŸåŇŰäź§åŔłæŸŕåIJĺæ§ŘäžŻçL'źåőŽåIJřæŰźæL'■åžŤèŕĕèćńä¡£çŤĺãĂĆ

çŘĘèğčåśĂéČĺåŔŸéĞŔ

äźŃåL'■æŔŘè£ĞïijŇåśĂéČĺåŔŸéĞŔäijŽærŤåĚĺåśĂåŔŸéĞŔè£ŘèąŇéĂ§åžęå£ńãĂĆ
åŕźäžŐécŚçźĄèő£éŮőçŽĎåŘ■çğřïijŇéĂŽè£ĞářĘè£ŹäžŻåŘ■çğřåŔŸæĹŘåśĂéČĺåŔŸéĞŔåŔŕäžěåŁäéĂ§çĺŃä
ä¿ŃåęĆïijŇçIJŃäÿŃäźŃåL'■åŕźäžŐ `compute_roots()` åĞ¡æŢřè£ŻèąŇä£őæŤźåŘŐçŽĎçL'ĹæIJňïijŽ

```
import math

def compute_roots(nums):
    sqrt = math.sqrt
    result = []
    result_append = result.append
    for n in nums:
        result_append(sqrt(n))
    return result
```

åIJĺè£ŹäÿłçL'ĹæIJňäÿ■ïijŇ`sqrt` äžŐ `match` æĺąåİŮèćńæŃ£åĞžåźűæŤ¿åĚěäžĘäÿĂäÿłåśĂéČĺåŔŸéĞŔ
åęĆædIJä¡äè£ŘèąŇè£ŹäÿłäžčçăĄïijŇåď'ğæęĆèŁśèť'ź25çğŠïijĹåŕźäžŐäźŃåL'■29çğŠåŔĹæŸŕäÿĂäÿłæŤźè£Ż
è£Źäÿłécĺåď'ŰçŽĎåŁäéĂ§åŐ§åŻäæŸŕåŻääÿžåŕźäžŐåśĂéČĺåŔŸéĞŔ `sqrt`
çŽĎæ§ěæL'¿èęĄå£ńäžŐåĚĺåśĂåŔŸéĞŔ `sqrt`

åŕźäžŐçśżäÿ■çŽĎåśđæĂğèő£éŮőäź§åŘŇæăůéĂĆçŤĺäžŐè£ŹäÿłåŐ§çŘĘãĂĆ
éĂŽåÿÿæİěèőšïijŇæ§ěæL'¿æ§ŘäÿłåĂijæŕŤåęĆ `self.name`
äijŽærŤèő£éŮőäÿĂäÿłåśĂéČĺåŔŸéĞŔèęĄæĚćäÿĂäžŻãĂĆ åIJĺåĘĚéČĺå¿łçŐŕäÿ■ïijŇåŔŕäžěåřĘæ§ŘäÿłéIJĂè

```
# Slower
class SomeClass:
    ...
    def method(self):
         for x in s:
             op(self.value)

# Faster
class SomeClass:

    ...
    def method(self):
         value = self.value
         for x in s:
             op(value)
```

éĄ£åĚ■äÿ■å£ĚèęĄçŽĎæŁ¡èśą

äżżä¡ŢæŮűåĂŹå¡Şä¡ää¡£çŤĺéćİåď´ŰçŽĎåď´ĎçŘĘåśĆïijĹæŕŤåęĆèčĚéěřåŹĺãĂĄåśđæĂğèő£éŮőãĂĄæŔ
æŕŤåęĆçIJŃäÿŃåęĆäÿŃçŽĎè£ŹäÿłçśżïijŽ

```
class A:
    def __init__(self, x, y):
        self.x = x
        self.y = y
    @property
    def y(self):
        return self._y
    @y.setter
    def y(self, value):
        self._y = value
```

çŐřåIJĺè£ŻèąŇäÿĂäÿłçőĂå■ŢæţŃèŕŢïijŽ

```
>>> from timeit import timeit
>>> a = A(1,2)
>>> timeit('a.x', 'from __main__ import a')
0.07817923510447145
>>> timeit('a.y', 'from __main__ import a')
0.35766440676525235
>>>
```

åŘŕäżěçIJŃåĹřïijŇèő£éŮőåśđæĂğyçŻÿæŕŤåśđæĂğxèĂŇèĺĂæĚćçŽĎäÿ■æ■ćäÿĂçĆźçĆźïijŇåď´ğæęĆæ
åęĆæđIJä¡ăåIJĺæĎŘæĂğèČ¡çŽĎèŕİïijŇéĆčäźĹåřśéIJĂèęĄéĞ■æŰřåőąèğĘäÿŃåŕźäžŐyçŽĎåśđæĂğèő£éŮőåŻ
åęĆæđIJæšąæIJL’å£ĚèęĄïijŇåřśä¡£çŤĺçőĂå■ŢåśđæĂğåŘğãĂĆ
åęĆæđIJäżĚäżĚæŸŕåŻääÿžåĚűäžŰçijŰçĺŃèŕ■èĺĂéIJĂèęĄä¡£çŤĺgetter/setteråĞ¡æŢŕåřśåŐżä£őæŤźäžčçăĄéč

ä¡£çŤĺåĘĚç¡őçŽĎåőźåŹĺ

åĘĚç¡őçŽĎæŢŕæ■őçśżåđŃæŕŤåęĆå■Ůçňęäÿšã ĂĄåĚČçżĎãĂĄåĹŮèąĺãĂĄéŻĘåŘĹåŠŇå■ŮåĚÿéČ¡æŸí
åęĆæđIJä¡ăæČşèĞłåůśåőđçŐřæŰřçŽĎæŢŕæ■őçżŞæđĎïijĹæŕŤåęĆéŞ¿æŐěåĹŮèąĺãĂĄåźşèąąæăŚç■L’ïijL’ïijŇ
éĆčäźĹèęĄæČşåIJĺæĂğèČ¡äÿŁè¿¿åĹřåĘĚç¡őçŽĎéĂ§ăžęåĞääźŐäÿ■åŕŕèČ¡ïijŇåŻäæ■d’ïijŇè£ŸæŸŕäźŰäźŰ

éĄ£åĚ■åĹŻåżžäÿ■å£ĚèęĄçŽĎæŢŕæ■őçżŞæđĎæĹŰåd’■åĹű

æIJL’æŮűåĂZçÍŃåžŕåŚŸæČşæŸ¿æŚĘäÿŃïijŇæđĎéĂääÿĂäžŻåźűæšąæIJL’å£ĚèęĄçŽĎæŢŕæ■őçżŞæđ

```
values = [x for x in sequence]
squares = [x*x for x in values]
```

äź§èőÿè£ŹéĞŇçŽĎæČşæşŢæŸŕéęŰåĚĹåřĘäÿĂäžŻåĂijæŤűéŻĘåĹřäÿĂäÿłåĹŮèąĺäÿ■ïijŇçĎűåŘŐä¡£çŤ
äÿ■è£ĞïijŇçňñäÿĂäÿłåĹŮèąĺåőŇåĚĺæšąæIJL’å£ĚèęĄïijŇåŕŕäžěçőĂå■ŢçŽĎåČŕäÿŃéĺćè£ŹæăůåĘŹïijŽ

```
squares = [x*x for x in sequence]
```

äÿŐæ■d’çŻÿåĚşïijŇè£ŸèęĄæşĺæĎŘäÿŃéĆčäźŻåŕźPythonçŽĎåĚśäžńæŢŕæ■őæIJžåĹűè£ĞäžŐåĄŕæL’ğ
æIJL’äžŻäžžåźűæšąæIJL’å¿Ĺåě¡çŽĎçŘĘèğčæĹŰä£ąäżżPythonçŽĎåĘĚå■ŸæĺąåđŃïijŇæżěçŤĺ
`copy.deepcopy()` äźŃçśżçŽĎåĞ¡æŢŕãĂĆ éĂŽåÿÿåIJĺè£ŹäźŻäžčçăĄäÿ■æŸŕåŕŕäžěåŐżæŐL’åd’■åĹűæŞ

èőÍèőž

åIJÍäijŸåÑŰäźŃåL’■ïijŇæIJL’å£ĚèęĄåĚĹçăŤçľűäÿŃä¡£çŤÍçŽĎçőŮæşŢãĂĆ éĂL’æŃľäÿĂäÿłåď■æİĆåžęäÿž O(n log n) çŽĎçőŮæşŢèęĄæŕŤä¡ăåŐżèřČæŢť’äÿĂäÿłåď■æİĆåžęäÿž O(n**2) çŽĎçőŮæşŢæL’ĂåÿęæİěçŽĎæĂĝèČ¡æŔŘå■ĞèęĄåď’gå¿Ůåď’ŽãĂĆ

åęĆæđIJä¡ăèğL’å¿Ůä¡ăè£ŸæŸŕå¿Ůè£ŻèąÑäijŸåÑŰïijŇéĆčäźĹèŕůäžŐæŢť’ä¡ŞèĂČèŹŚãĂĆ ä¡IJäÿžäÿĂèĹňåĞĘåĹŹïijÑäÿ■èęĄåŕźçÍŃåžŔçŽĎæŕŘäÿĂäÿłéČĺåĹĘéČ¡åŐżäijŸåÑŰ,åŻăäÿžè£ŹäžŻä£őæŤźä¡ăåžŤèŕěäÿŞæşĺäžŐäijŸåÑŰäžğçŤ§æĂĝèČ¡çŞűécĹçŽĎåIJræŰźïijÑæŕŤåęĆåĘĚéČĺå¿łçŐŕãĂĆ

ä¡ăè£ŸèęĄæşĺæĎŘå¿őåřŘäijŸåÑŰçŽĎçżŞæđIJãĂĆä¿ŃåęĆèĂČèŹŚäÿŃéİćåĹŻåžžäÿĂäÿłå■ŮåĚÿçŽĎă

```
a = {
    'name' : 'AAPL',
    'shares' : 100,
    'price' : 534.22
}

b = dict(name='AAPL', shares=100, price=534.22)
```

åŘŐéİćäÿĂçğ■åĘŹæşŢæŻť’çőĂæť’ĄäÿĂäžŻïijĹä¡ăäÿ■éIJĂèęĄåIJÍåĚşéŤőå■ŮäÿŁè¿ŞåĚěåijŢåŔůïijL’ãĂ äÿ■è£ĞïijÑåęĆæđIJä¡ăåřĘè£Źäÿđ’äÿłäžčçăĄçL’Ğæőţè£ŻèąÑæĂğèČ¡æţŃèŕŢåŕźæŕŤæŮűïijÑäijŽåŔŞçŐřä¡£ç `dict()` çŽĎæŰźåijŔäijŽæĚćäžĘ3åĂ■ãĂĆ çIJÑåĹrè£ŹäÿłïijÑä¡ăæŸŕäÿ■æŸŕæIJL’åĘşåŁĺæŁŁæL’ĂæIJL’ä¡£ `dict()` çŽĎäžčçăĄéČ¡æŻ£æ■ćæĹŘçñäÿĂçğ■ãĂĆ äÿ■åď’§ïijÑèĄłæŸŐçŽĎçĺÑåžŔåŚŸåŔłäijŽåĚşæşĺäžŰ

åęĆæđIJä¡ăçŽĎäijŸåÑŰèęĄæśĆæŕŤè¿ČéńŸïijÑæIJñèŁĆçŽĎè£ŹäžŻçőÅå■ŢæŁĂæIJŕæżąèűşäÿ■äžĘïij ä¿ÑåęĆïijÑPyPyåůèçĺÑæŸŕPythonèğčéĞŁåŹĺçŽĎåŔęåď’ŮäÿĂçğ■åődčŐřïijÑåőČäijŽåĹĘædŘä¡ăçŽĎçĺÑåžł åőČæIJL’æŮűåĂŹèČ¡ædĄåď’gçŽĎæŔŘå■ĞæĂğèČ¡ïijÑéĂŽåÿÿåŔŕäžěæŐěè£ŚCäžčçăĄçŽĎéĂ§åžęãĂĆ äÿ■è£ĞåŔŕæČIJçŽĎæŸŕïijÑåĹřåĘŹè£ŹæIJñäźęä¡■ç¡őïijÑPyPyè£Ÿäÿ■èČ¡åőÑåĚĺæŤŕæŇĄPython3. åŽăæ■ď’ïijÑè£ŹäÿłæŸŕä¡ăåřĘæÍěéIJĂèęĄåŐżçăŤçľűçŽĎãĂĆä¡ăè£ŸåŔŕäžěèĂČèŹŚäÿŃNumbaåůěçĺÑïijÑ NumbaæŸŕäÿĂäÿłåIJĺä¡ăä¡£çŤÍèČĕéřåŹĺæÍěéĂL’æŃľPythonåĞ¡æŢrè£ŻèąÑäijŸåÑŰæŮűçŽĎåŁĺæĂĄçijŰ è£ŹäžŽåĞ¡æŢřäijŽä¡£çŤÍLLVMècńçijŰèŕŚæĹŘæIJñåIJřæIJžåŹĺçăĄãĂĆåőČåŘŇæăůåŔŕäžěædĄåď’gçŽĎæŔ ä¡ĘæŸŕïijÑèů§PyPyäÿĂæăůïijÑåőČåŕźäžŐPython 3çŽĎæŤŕæŇĄçŐřåIJÍè£ŸåĄIJçŢŹåIJÍåőđéłÑéŸűæőţãĂĆ

æIJĂåŘŐæĹŚåijŢçŤÍJohn Ousterhoutèŕť’è£ĞçŽĎèŕÍä¡IJäÿžçżŞåŕ¿ïijŽâĂIJæIJĂåě¡çŽĎæĂğèČ¡äijŸåÑŰçŻť’åĹŕä¡ăçIJ§çŽĎéIJĂèęĄäijŸåÑŰçŽĎæŮűåĂŹåĘ■åŐżèĂČèŹŚåőČãĂĆçąőä£Íä¡ăçÍÑåžŔæ■čçąőçŽĎè£Řè

çňňå■ĄäžŤçńäïijŽCèŕ■èÍĂæL’ľåśŢ

æIJñçńăçİĂçIJijäžŐäżŐPythonèő£éŮőCäžčçăĄçŽĎéŮőécŸãĂĆèőÿåď’ŽPythonåĘĚç¡őåžŞæŸŕçŤÍCåĘŹ èő£éŮőCæŸŕèőľPythonçŽĎåŕźçŐřæIJL’åžŞè£ŻèąÑäžď’äžŠäÿĂäÿłéĞ■èęĄçŽĎçżĎæĹŔéČÍåĹĘãĂĆ è£Źäź§æŸŕäÿĂäÿłå¡Şä¡ăéİćäÿť’äżŐPython 2 åĹř Python 3æL’ľåśŢäžčçăĄçŽĎéŮőécŸãĂĆ èŹ¡çĎűPythonæŔŘä¿ŻäžĘäÿĂäÿłåź£æşŽçŽĎçijŰçÍŃAPIïijÑåőđéŹĚäÿŁæIJL’å¿Ĺåď’ŽæŰźæşŢæİèåď’ĎçŘĘ çŻÿæŕŤèŕŢåŻ¿çżŹåĞžåŕźäžŐæŕŘäÿĂäÿłåŔŕèČ¡çŽĎåůěåĚůæĹŰæŁĂæIJŕçŽĎèŕęçżĘåŔĆèĂČïijÑ æĹŚäźĹéĞĞçŤÍçŽĎæŸŕæŸŕéŻĘäÿ■åIJÍäÿĂäÿłåřŘçL’Ğæőţç ŽĎC++äžčçăĄïijÑäžěåŘŁäÿĂäžŽæIJL’äžčèąÍæ è£Źäÿłçżőæă Ğæ ŸŕæŔŘä¿ŻäÿĂçşžåĹŮçŽĎçijŰçÍŃæÍąæİ£ïijÑæIJL’çżŔéłŃçŽĎçÍŃåžŔåŚŸåŔŕäžěæL’ľåśŢè

è£ŹéĞÑæŸŕæĹŚäžňåřĘåIJÍåď’géČĺåĹĘçğŸçś■äÿ■åůěä¡IJçŽĎäžčçăĄïijŽ

```
/* sample.c */_method
#include <math.h>

/* Compute the greatest common divisor */
int gcd(int x, int y) {
    int g = y;
    while (x > 0) {
        g = x;
        x = y % x;
        y = g;
    }
    return g;
}

/* Test if (x0,y0) is in the Mandelbrot set or not */
int in_mandel(double x0, double y0, int n) {
    double x=0,y=0,xtemp;
    while (n > 0) {
        xtemp = x*x - y*y + x0;
        y = 2*x*y + y0;
        x = xtemp;
        n -= 1;
        if (x*x + y*y > 4) return 0;
    }
    return 1;
}

/* Divide two numbers */
int divide(int a, int b, int *remainder) {
    int quot = a / b;
    *remainder = a % b;
    return quot;
}

/* Average values in an array */
double avg(double *a, int n) {
    int i;
    double total = 0.0;
    for (i = 0; i < n; i++) {
        total += a[i];
    }
    return total / n;
}

/* A C data structure */
typedef struct Point {
    double x,y;
} Point;

/* Function involving a C data structure */
```

```
double distance(Point *p1, Point *p2) {
    return hypot(p1->x - p2->x, p1->y - p2->y);
}
```

è£Źæőţäžčçă Ąå ŇĚåŘńäžĘåď Žçğ■äÿ■åŘŇçŽĎCèŕ■èĺĂçijŰçĺŃçĽźæĂğãĂĆ éęŰåĚĹïijŇè£ŹéĞŇæIJĽå¿Ĺåď ŽåĞ¡æŢřæŕŤåęĆ `gcd()` åŠŇ `is_mandel()` ãĂĆ `divide()` åĞ¡æŢřæŸŕäÿĂäÿłè£ŤåŻđåď Žäÿłå Äijç ŽĎCåĞ¡æŢřä¿Ńå■ŘïijŇåĚűäÿ■æIJĽäÿĂäÿłæŸŕéĂŽè£Č `avg()` åĞ¡æŢřéĂŽè£ĞäÿĂäÿłCæŢřçžĎæĽğèąŇæŢřæ■őèĄźéŽĘæŞ■ä¡IJãĂĆ`Point` åŠŇ `distance()` åĞ¡æŢřæűĽåŔŁåĹřäžĘCçžŞæđĎä¡ŞãĂĆ

åŕźäžŐæŐěäÿŃæİěçŽĎæĽĂæIJĽåřŔèŁĆïijŇåĚĹåĄĞåőŽäÿŁéİćçŽĎäžčçăĄåůšçżŔèćńåĘŹåĚěäžĘäÿĂ çĎűåŘŐåőČäžňçŽĎåőŽäźĽèćńåĘŹåĚěäÿĂäÿłåŘ■åŔńâĂIJsample.hâĂİçŽĎåď ť æŰĞäžűäÿ■ïijŇ åźűäÿŤèćńçijŰèŕŚäÿžäÿĂäÿłäžŞåŔńâĂIJlibsampleâĂİïijŇèČ¡èćńéŞ¿æŐěåĹřåĚűäžŰCèŕ■èĺĂäžčçăĄäÿ■ãĂĆ çijŰèŕŚåŠŇéŞ¿æŐěçŽĎçżĘèŁĆä¿İæ■őçşżçż§çŽĎäÿ■åŘŇèĂŇäÿ■åŘŇïijŇä¡ĘæŸŕè£Źäÿłäÿ■æŸŕæĹŚäžňåĚ åęĆæđIJä¡äèęĄåď ĎçŘĘCäžčçăĄïijŇæĹŚäžňåĄĞåőŽè£ŹäžŽå§žçąĂçŽĎäÿIJèě£ä¡äéČ¡æŐŇæŔąäžĘãĂĆ

Contents:

15.1 ä¡£çŤĺctypesèő£éŮőCäžčçăĄ

éŮőécŸ

ä¡ăæIJĽäÿĂäžŻCåĞ¡æŢřåůšçżŔèćńçijŰèŕŚåĹřåĚśäžńåžŞæĹŰDLLäÿ■ãĂĆä¡ăåÿŇæIJŻåŔŕäžěä¡£çŤĺçžíèĂŇäÿ■çŤĺçijŰåĘŹéćİåď ŰçŽĎCäžčçăĄæĹŰä¡£çŤĺçňňäÿĽæŰźæĽľ åśŢåůěåĚůãĂĆ

èğčåĘşæŰźæąĹ

åŕźäžŐéIJĂèęĄèřČçŤĺCäžčçăĄçŽĎäÿĂäžŻåŕŔçŽĎéŮőécŸïijŇéĂŽåÿÿä¡£çŤĺPythonæăĞåĞĘåžŞäÿ■çŽ `ctypes` æĺąåİŮåŕśèűşåď §äžĘãĂĆ èęĄä¡£çŤĺ `ctypes` ïijŇä¡ăéęŰåĚĹèęĄçąőä£Ťä¡ăèęĄèő£éŮőçŽĎCäžčçăĄ ïijĹåŘŇæăůçŽĎæđűæđĎãĂĄå■Ůåď ğåŕŔãĂĄçijŰèŕŚåŹĺç■ĽïijĽçŽĎæ§ŘäÿłåĚśäžńåžŞäÿ■äžĘãĂĆ äÿžäžĘè£ŻèąŇæIJńèŁĆçŽĎæijŤçď žïijŇåĄĞèő¿ä¡ăæIJĽäÿĂäÿłåĚśäžńåžŞåŘ■å■ŮåŔń `libsample.so` ïijŇéĞŇéİćçŽĎåĘĚåőźåřśæŸŕ15çńääžŃçż■éČĺåĹĘéĆčæăůãĂĆ åŔęåď Űè£ŸåĄĞèő¿è£ŹäÿłåĞ¡æŢřåžŞïijŇä¡ăèęĄåĚĹæđĎåžžäÿĂäÿłåŇĚèČĚåőČçŽĎPythonæĺąåİŮïijŇåęĆäÿŃè `libsample.so` æŰĞäžűèćńæŤ¿ç¡őåĹřä¡■äžŐ `sample.py` æŰĞäžűçŽÿåŘŇçŽĎçŽőå¡Ţäÿ■äžĘãĂĆ

èęĄèő£éŮőè£ŹäÿłåĞ¡æŢřåžŞïijŇä¡ăèęĄåĚĹæđĎåžžäÿĂäÿłåŇĚèČĚåőČçŽĎPythonæĺąåİŮïijŇåęĆäÿŃè

```
# sample.py
import ctypes
import os

# Try to locate the .so file in the same directory as this file
_file = 'libsample.so'
_path = os.path.join(*(os.path.split(__file__)[:-1] + (_file,)))
_mod = ctypes.cdll.LoadLibrary(_path)

# int gcd(int, int)
```

```
gcd = _mod.gcd
gcd.argtypes = (ctypes.c_int, ctypes.c_int)
gcd.restype = ctypes.c_int

# int in_mandel(double, double, int)
in_mandel = _mod.in_mandel
in_mandel.argtypes = (ctypes.c_double, ctypes.c_double, ctypes.c_
↪int)
in_mandel.restype = ctypes.c_int

# int divide(int, int, int *)
_divide = _mod.divide
_divide.argtypes = (ctypes.c_int, ctypes.c_int, ctypes.
↪POINTER(ctypes.c_int))
_divide.restype = ctypes.c_int

def divide(x, y):
    rem = ctypes.c_int()
    quot = _divide(x, y, rem)

    return quot,rem.value

# void avg(double *, int n)
# Define a special type for the 'double *' argument
class DoubleArrayType:
    def from_param(self, param):
        typename = type(param).__name__
        if hasattr(self, 'from_' + typename):
            return getattr(self, 'from_' + typename)(param)
        elif isinstance(param, ctypes.Array):
            return param
        else:
            raise TypeError("Can't convert %s" % typename)

    # Cast from array.array objects
    def from_array(self, param):
        if param.typecode != 'd':
            raise TypeError('must be an array of doubles')
        ptr, _ = param.buffer_info()
        return ctypes.cast(ptr, ctypes.POINTER(ctypes.c_double))

    # Cast from lists/tuples
    def from_list(self, param):
        val = ((ctypes.c_double)*len(param))(*param)
        return val

    from_tuple = from_list

    # Cast from a numpy array
    def from_ndarray(self, param):
```

```
        return param.ctypes.data_as(ctypes.POINTER(ctypes.c_double))

DoubleArray = DoubleArrayType()
_avg = _mod.avg
_avg.argtypes = (DoubleArray, ctypes.c_int)
_avg.restype = ctypes.c_double

def avg(values):
    return _avg(values, len(values))

# struct Point { }
class Point(ctypes.Structure):
    _fields_ = [('x', ctypes.c_double),
                ('y', ctypes.c_double)]

# double distance(Point *, Point *)
distance = _mod.distance
distance.argtypes = (ctypes.POINTER(Point), ctypes.POINTER(Point))
distance.restype = ctypes.c_double
```

åęĆædIJäÿĂåĹĞæ■čåÿÿïijŇä¡ăářśåŘŕäžěåŁăè¡¡åźűä¡£çŤĺéĞŇéİćåőŽäźL’çŽĎCåĞ¡æŢřäžĘãĂĆä¿ŃåęĆï

```
>>> import sample
>>> sample.gcd(35,42)
7
>>> sample.in_mandel(0,0,500)
1
>>> sample.in_mandel(2.0,1.0,500)
0
>>> sample.divide(42,8)
(5, 2)
>>> sample.avg([1,2,3])
2.0
>>> p1 = sample.Point(1,2)
>>> p2 = sample.Point(4,5)
>>> sample.distance(p1,p2)
4.242640687119285
>>>
```

èőÍèőž

æIJňářŔèŁĆæIJL’å¿Ĺåd’ŽåĂija¿ŮæĹŠäžňèŕęçżĘèőÍèőžçŽĎåIJřæŰźãĂĆ éęŰåĚĹæŸŕářźäžŐCåŠŇPythonäżčçăĄäÿĂètůæL’ŞåŇĚçŽĎéŮőécŸïijŇåęĆædIJä¡ăåIJÍä¡£çŤÍ `ctypes` æİěèő£éŮőçijŰèŕŠåŘŐçŽĎCäžčçăĄïijŇ éĆčäźĹéIJĂèęĄçąőä£Íè£ŹäÿłåĚšäžńåžŞæŤ¿åIJÍ `sample.py` æĺąåİŮåŘŇäÿĂäÿłåIJřæŰźãĂĆ äÿĂçğ■åŔŕèČ¡æŸŕářĘçŤ§æĹŘçŽĎ . `so` æŰĞäżűæŤ¿ç¡őåIJÍèęĄä¡£çŤĺåőČçŽĎPythonäżčçăĄåŘŇäÿĂäÿłçŻőå¡ŢäÿŃãĂĆ æĹŠäžňåIJÍ `recipeâĂŤsample.py` äÿ■ä¡£çŤÍ `__file__` åŔŸéĞŔæİěæ§ęçIJŃåőČèćńåőL’èčĚçŽĎä¡■ç¡őïijŇ çĎűåŘŐædĎéĂääÿĂäÿłæŇĞåŘŠåŘŇäÿĂäÿłçŻőå¡Ţäÿ■ç `libsample.so` æŰĞäżűçŽĎèůŕå¿ĎãĂĆ

åęĆæđIJCåĞ¡æŢřåžŞèćńåőL'èčĚåĹřåĚűäžŰåIJřæŰźïijŇéĆčäźĹä¡ăåřśèęĄä£őæŤźçŻÿåžŤçŽĎèůřå¿ĎãĂ
åęĆæđIJCåĞ¡æŢřåžŞåIJĺä¡ăæIJžåŹĺäÿŁèćńåőL'èčĚäÿžäÿĂäÿłæăĞåĞĘåžŞäžĘïijŇ
éĆčäźĹåŔřäžěä¡£çŤĺ `ctypes.util.find_library()` åĞ¡æŢřæİěæ§ěæL'¿ïijŽ

```
>>> from ctypes.util import find_library
>>> find_library('m')
'/usr/lib/libm.dylib'
>>> find_library('pthread')
'/usr/lib/libpthread.dylib'
>>> find_library('sample')
'/usr/local/lib/libsample.so'
>>>
```

äÿĂæŮęä¡ăç§ěéĄŞäžĘCåĞ¡æŢřåžŞçŽĎä¡■ç¡őïijŇéĆčäźĹářšåŔřäžěåČŔäÿŃéİćè£Źæăůä¡£çŤĺ
`ctypes.cdll.LoadLibrary()` æİěåŁăè¡¡åőČïijŇ åĚűäÿ■ `_path`
æŸřæăĞåĞĘåžŞçŽĎåĚĺèůřå¿ĎïijŽ

```
_mod = ctypes.cdll.LoadLibrary(_path)
```

åĞ¡æŢřåžŞèćńåŁăè¡¡åŘŐïijŇä¡äéIJĂèęĄçijŰåĘŹåĞääÿłèŕ■åŔěæİěæŔŘåŔŰçL'źåőŽçŽĎçňęåŔůåźűæŇ
ářšåČŔäÿŃéİćè£Źäÿłäžčçă Ąç L'Ğæőţäÿ ĂæăůïijŽ

```
# int in_mandel(double, double, int)
in_mandel = _mod.in_mandel
in_mandel.argtypes = (ctypes.c_double, ctypes.c_double, ctypes.c_
↪int)
in_mandel.restype = ctypes.c_int
```

åIJĺè£Źæőţäžčçă Ąäÿ■ïijŇ `.argtypes` åšđæĂğæŸŕäÿĂäÿłåĚČçżĎïijŇåŇĚåŘńäžĘæ§ŘäÿłåĞ¡æŢřçŽĎè
èĂŇ `.restype` åřśæŸŕçŻÿåžŤçŽĎè£ŤåŻđçśžåđŃãĂĆ `ctypes`
åőŽäźL'äžĘåď ğéĞŘçŽĎçśžåđŃåŕźèśąïijĹæŕŤåęĆc_double, c_int, c_short, c_floatç■L'ïijL'ïijŇ
äžčèąĺäžĘåŕźåžŤçŽĎCæŢřæ■őçśžåđŃãĂĆåęĆæđIJä¡ăæČşèől' Pythonè Č¡åď §äijăéĂŠæ■čçąőçŽĎåŔĆæŢřçśž
éĆčäźĹè£ŹäžŻçśžåđŃç■¿åŘ■çŽĎçżŠåőŽæŸŕå¿ĹéĞ■èęĄçŽĎäÿĂæ■ěãĂĆåęĆæđIJä¡ăæšąæIJL'è£ŹäźĹåĄŽï
è£ŸåŔŕèČ¡äijŽåŕijèĞt' æŢt' äÿłèğčéĞŁåŹĺè£ŻçĺŃæŇĆæŐL'ãĂĆ
ä¡£çŤĺctypesæIJL'äÿĂäÿłéžżçČęçĆźçŽĎåIJřæŰźæŸŕåŐ§çŤ§çŽĎCäžčçăĄä¡£çŤĺçŽĎæIJŕèŕ■åŔŕèČ¡èů§Pytho
`divide()` åĞ¡æŢřæŸŕäÿĂäÿłå¿Ĺåě¡çŽĎä¿Ńå■ŘïijŇåőĆéĂŽè£ĞäÿĂäÿłåŔĆæŢřéŹď äžěåŘęäÿĂäÿłåŔĆæŢ
ář¡çőąè£ŹæŸŕäÿĂäÿłå¿Ĺåÿÿèğ ĄçŽĎCæŁĂæIJŕïijŇä¡ĘæŸŕåIJĺPythonäÿ■å■ť äÿ■ç§ěéĄŞæĂŐæăůæÿĚæŹŕçŽ
ä¿ŃåęĆïijŇä¡ääÿ■èČ¡åČŔäÿŃéİćè£ŹæăůçőĂå■ŢçŽĎåĄŽïijŽ

```
>>> divide = _mod.divide
>>> divide.argtypes = (ctypes.c_int, ctypes.c_int, ctypes.
↪POINTER(ctypes.c_int))
>>> x = 0
>>> divide(10, 3, x)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ctypes.ArgumentError: argument 3: <class 'TypeError'>: expected LP_
↪c_int
instance instead of int
>>>
```

åřšçőŮè£Źäÿłè Č¡æ■čçąőçŽĎåůěä¡IJïijŇåőČäijŽè£ÌåŔ■Pythonåŕźäž ŐæŢť æŢřçŽĎäÿ■åŔŕæŻť æŤźåŐ§å
åŕźäžŐæűĽ'åŔŁåĹřæŇĞéŠĹçŽĎåŔĆæŢřïijŇä¡ăéĂŽåÿÿéIJĂèęĄåĚĹæđĎåżžäÿĂäÿłçŻÿåžŤçŽĎctypesåŕźèšą

```
>>> x = ctypes.c_int()
>>> divide(10, 3, x)
3
>>> x.value
1
>>>
```

åIJÍè£ŹéĞŇïijŇäÿĂäÿł `ctypes.c_int` åőđä¿ŃèćńåĹŻåżžåźűä¡IJäÿžäÿĂäÿłæŇĞéŠĹèćńäijăè£ŻåŐżãĂ
èů§æŹőéĂŽPythonæŢť å¡ćäÿ■åŘŇçŽĎæŸŕïijŇäÿĂäÿł `c_int`
åŕźèšąæŸŕåŔŕäżěèćńä£őæŤźçŽĎãĂĆ `.value` åšđæĂğåŔŕèćńçŤĺæİěèŐůåŔŰæĹŰæŻť æŤźè£ŹäÿłåĂijãĂĆ

åŕźäžŐéĆčäžŻäÿ■åČŔPythonçŽĎCèřČçŤĺïijŇéĂŽåÿÿåŔŕäżěåĘŹäÿĂäÿłåřŔçŽĎåŇĚèčĚåĞ¡æŢřãĂĆ
è£ŹéĞŇïijŇæĹŠäžňèőľ `divide()` åĞ¡æŢřéĂŽè£ĞåĚČçżĎæİěè£ŤåŽđäÿd'äÿłçżŞæđIJïijŽ

```
# int divide(int, int, int *)
_divide = _mod.divide
_divide.argtypes = (ctypes.c_int, ctypes.c_int, ctypes.
↪POINTER(ctypes.c_int))
_divide.restype = ctypes.c_int

def divide(x, y):
    rem = ctypes.c_int()
    quot = _divide(x,y,rem)
    return quot, rem.value
```

`avg()` åĞ¡æŢřåŔĹæŸŕäÿĂäÿłæŰřçŽĎæŇŚæĹŸãĂĆCäžčçăĄæIJ§æIJŻæŐěåŔŮåĹřäÿĂäÿłæŇĞéŠĹåŠ
ä¡ĘæŸŕïijŇåIJÍPythonäÿ■ïijŇæĹŠäžňå£Ěéążè ĂČèŹŚè£Źäÿłéůőéć ŸïijŽæŢřçŽĎæŸŕåŢěïij§åőČæŸŕäÿĂäÿłåĹ
è£ŸæŸŕ `array` æĺąåİŮäÿ■çŽĎäÿĂäÿłæŢřçŽĎïij§è£ŸæŸŕäÿĂäÿł `numpy`
æŢřçŽĎïij§è£ŸæŸŕèŕt æĽĂæIJL'éČ¡æŸŕïij§ åőđéŹĚäÿŁïijŇäÿĂäÿłPythonâĂIJæŢřçŽĎâĂİæIJL'åd'Žçğ■å¡ćå

`DoubleArrayType` æijŤçd'žäžĘæĂŐæăůåd'ĎçŘĘè£Źçğ■æČĚåĘţãĂĆ
åIJÍè£Źäÿłçśžäÿ■åőŽäźL'äžĘäÿĂäÿłå■ŢäÿłæŰźæşŢ `from_param()` ãĂĆ
è£ŹäÿłæŰźæşŢçŽĎèğŠèĽšæŸŕæŐěåŮůäÿĂäÿłå■ŢäÿłåŔĆæŢřçĎűåŘŐářĘåĚűåŘŚäÿŃè¡ňæ■ćäÿžäÿĂäÿłåŘĺ
ïijĹæIJňä¿Ńäÿ■æŸŕäÿĂäÿł `ctypes.c_double` çŽĎæŇĞéŠĹïijL'ãĂĆ
åIJÍ `from_param()` äÿ■ïijŇä¡ăåŔŕäžěåĄŽäżžä¡Ţä¡ăæČşåĄŽçŽĎäžŃãĂĆ
åŔĆæŢřçŽĎçśžåđŇåŘ■èćńæŔŘåŔŰåĞžæİěåźűèćńçŤĺäžŐåĹĘåŚŠåĹřäÿĂäÿłæŻť åĚůä¡ŞçŽĎæŰźæşŢäÿ■åŐ
ä¿ŃåęĆïijŇåęĆæđIJäÿĂäÿłåĹŮèąĺèćńäijăéĂŠè£ĞæİěïijŇéĆčäźĹ `typename` åřšæŸŕ `list`
ïijŇ çĎűåŘŐ `from_list` æŰźæşŢèćńèřĆçŤĺãĂĆ

åŕźäžŐåĹŮèąĺåŠŇåĚČçżĎïijŇ`from_list` æŰźæşŢåřĘåĚűè¡ňæ■ćäÿžäÿĂäÿł `ctypes`
çŽĎæŢřçŽĎåŕźèśąãĂĆ è£ŹäÿłçIJŃäÿŁåŐżæIJL'çĆźåěĞæĂłïijŇäÿŃéİćæĹŠäžňä¡£çŤĺäÿĂäÿłäžd'äžŠåijŔä¿Ń
`ctypes` æŢřçŽĎïijŽ

```
>>> nums = [1, 2, 3]
>>> a = (ctypes.c_double * len(nums))(*nums)
>>> a
<__main__.c_double_Array_3 object at 0x10069cd40>
>>> a[0]
```

```
1.0
>>> a[1]
2.0
>>> a[2]
3.0
>>>
```

åŕźäžŐæŢřçżĎåŕźèśąïijŇfrom_array() æŔŘåŔŰåžŢåśĆçŽĎåĘĚå■ŸæŇĞéŠĹåźűåřĘåĚűè¡ñæ■ćäÿžctypes æŇĞéŠĹåŕźèśąãĂĆä¿ŃåęĆïijŽ

```
>>> import array
>>> a = array.array('d',[1,2,3])
>>> a
array('d', [1.0, 2.0, 3.0])
>>> ptr_ = a.buffer_info()
>>> ptr
4298687200
>>> ctypes.cast(ptr, ctypes.POINTER(ctypes.c_double))
<__main__.LP_c_double object at 0x10069cd40>
>>>
```

from_ndarray() æijŤçd'žäžĘåŕźäžŐ numpy æŢřçżĎçŽĎè¡ñæ■ćæŞ■ä¡IJãĂĆéĂŽè£ĞåőŽäźL' DoubleArrayType çśžåźűåIJÍ avg() çśžåđŃç■¿åŘ■äÿ■ä¡£çŤĺåőČïijŇéĆčäźĹè£ŹäÿłåĞ¡æŢřåřśèČ¡æŐěåŔŮåd'Žäÿłäÿ■åŘŇçŽĎçśžæŢřçżĎè¿ŞåĚěäžĘïijŽ

```
>>> import sample
>>> sample.avg([1,2,3])
2.0
>>> sample.avg((1,2,3))
2.0
>>> import array
>>> sample.avg(array.array('d',[1,2,3]))
2.0
>>> import numpy
>>> sample.avg(numpy.array([1.0,2.0,3.0]))
2.0
>>>
```

æIJňèŁĆæIJĂåŘŐäÿĂéČĺåĹĘåŘŚä¡ăæijŤçd'žäžĘæĂŐæăůåd'ĎçŘĘäÿĂäÿłçőĂå■ŢçŽĎCçżŞædĎãĂĆåŕźäžŐçżŞædĎä¡ŞïijŇä¡ăåŘłéIJÄèęĄåČŔäÿŇéÍć è£Źæăůçő Ăå■ŢçŽĎåőŽäźL'äÿĂäÿłçśžïijŇåŇĚåŘńçŻÿåžŢç

```
class Point(ctypes.Structure):
    _fields_ = [('x', ctypes.c_double),
                ('y', ctypes.c_double)]
```

äÿĂæŮęçśžèćńåőŽäźL'åŘŐïijŇä¡ăåŕśåŔŕäžěåIJÍçśžåđŃç■¿åŘ■äÿ■æĹŰèĂĚæŸŕéIJÄèęĄåőđä¿ŃåŇŰçżŞ

```
>>> p1 = sample.Point(1,2)
>>> p2 = sample.Point(4,5)
>>> p1.x
1.0
```

```
>>> p1.y
2.0
>>> sample.distance(p1,p2)
4.242640687119285
>>>
```

æIJĂåŘŐäÿĂäžŻåřŔçŽĎæŔŘçď žïijŽåęĆæđIJä¡ăæČşåIJĺPythonäÿ■èő£éŮőäÿĂäžŻåřŔçŽĎCåĞ¡æŢřïijŇ
`ctypes` æŸŕäÿĂäÿłå¿ĹæIJL'çŤĺçŽĎåĞ¡æŢřåžŞãĂĆ år¡çőąåęĆæ■d'ïijŇåęĆæđIJä¡ăæČşèęĄåŐžèő£éŮőäÿĂ
`Swig` (15.9èŁĆäijŽèőšåĹř) æĹŰ CythonïijĹ15.10èŁĆïijL'ãĂĆ

åŕźäžŐåď ğåđŃåžŞçŽĎèő£éŮőæIJL'äÿłäÿżèęĄéŮőécŸïijŇçŤšäžŐctypesåźűäÿ■æŸŕåőŇåĚĺèĞłåŁĺåŇŰï
éĆčäźĹä¡ăåŕšå£ĚéążèŁśèt'źåď ğéĞŔæŮűéŮt'æİěçijŰåĘŹæL'ĂæIJL'çŽĎçśžåđŃç■¿åŘ■ïijŇåŕšåČŔä¿Ńå■Řäÿ
åęĆæđIJåĞ¡æŢřåžŞåď §åď ■æİĆïijŇä¡ăè£Ÿå¿ŮåŐžçijŰåĘŹå¿Ĺåď ŽåřŔçŽĎåŇĚèčĚåĞ¡æŢřåŠŇæŤŕæŇĄçśž
åŔęåď ŰïijŇéŹď éİđä¡ăåůšçżŔåőŇåĚĺçš¿éĂŽäžĘæL'ĂæIJL'åžŢåśĆçŽĎCæŐěåŔčçżĘèŁĆïijŇåŇĚæŃňåĘĚå■
éĂŽåÿÿäÿĂäÿłå¿ĹåřŔçŽĎäžčçăĄçijžéŹůãĂĄèő£éŮőèűŁçŢŇæĹŮåĚűäžŰçśžäijijéŤŹèŕŕåřśèČ¡èőľ PythonçĺŇ

ä¡IJäÿž `ctypes` çŽĎäÿĂäÿłæŻ£äżčïijŇä¡ăè£ŸåŔŕäżěèĂČèŹŚäÿŇCFFIãĂĆCFFIæŔŘä¿ŻäžĘå¿Ĺåď Žç
ä¡ĘæŸŕä¡£çŤĺCèŕ■æşŢåźűæŤŕæŇĄæŻťåď ŽéńŸçžğçŽĎCäżčçăĄçśžåđŃãĂĆ
åĹřåĘŹè£ŹæIJñäźęäÿžæ■ćïijŇCFFIè£ŸæŸŕäÿĂäÿłçŻÿåŕźè¿ČæŰřçŽĎåůěçĺŃïijŇ
ä¡ĘæŸŕåőČçŽĎæţĄèąŇåžęæ■čåIJĺå£néĂ§äÿŁå■ĞãĂĆ çŤŽèĞşè£ŸæIJL'åIJĺèőĺèőžåIJĺPythonåŘĘæİěçŽĎçL'İ

15.2 çőÅå■ŢçŽĎCæL'ľåśŢæĺąåİŮ

éŮőécŸ

ä¡ăæČşäÿ■ä¿İéİăåĚűäżŰåůěåĚůïijŇçŻť'æŐěä¡£çŤĺPythonçŽĎæL'ľåśŢAPIæİěçijŰåĘŹäÿĂäžŻçőÅå■ŢçŽ

èğčåĘşæŰźæąĹ

åŕźäžŐçőÅå■ŢçŽĎCäżčçăĄïijŇædĎåžžäÿĂäÿłèĞłåőŽäźL'æL'ľåśŢæĺąåİŮæŸŕå¿Ĺåőźæ ŸŞçŽĎãĂĆ
ä¡IJäÿžçňňäÿĂæ■ëïijŇä¡ăéIJÅèęĄçąőä£Ïä¡ăçŽĎCäżčçăĄæIJL'äÿĂäÿłæ■čçąőçŽĎåď ť'æŰĞäžűãĂĆä¿ŃåęĆïij

```
/* sample.h */

#include <math.h>

extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

éĂŽåÿÿæİêèőšïijŇè£Źäÿłåď'ťæŰĞäžűèęĄåŕźåžŤäÿĂäÿłåůšçżŔèćńå■ŢçŃňçijŰèŕŚè£ĞçŽĎåžŞãĂĆ
æIJĽäžĘè£ŹäžŻïijŇäÿŃéİćæĹŚäžňæijŤçď'žäÿŃçijŰåĘŹæĽ'ľåśŢåĞ¡æŢřçŽĎäÿĂäÿłçőĂå■Ţä¿Ńå■ŘïijŽ

```
#include "Python.h"
#include "sample.h"

/* int gcd(int, int) */
static PyObject *py_gcd(PyObject *self, PyObject *args) {
  int x, y, result;

  if (!PyArg_ParseTuple(args,"ii", &x, &y)) {
    return NULL;
  }
  result = gcd(x,y);
  return Py_BuildValue("i", result);
}

/* int in_mandel(double, double, int) */
static PyObject *py_in_mandel(PyObject *self, PyObject *args) {
  double x0, y0;
  int n;
  int result;

  if (!PyArg_ParseTuple(args, "ddi", &x0, &y0, &n)) {
    return NULL;
  }
  result = in_mandel(x0,y0,n);
  return Py_BuildValue("i", result);
}

/* int divide(int, int, int *) */
static PyObject *py_divide(PyObject *self, PyObject *args) {
  int a, b, quotient, remainder;
  if (!PyArg_ParseTuple(args, "ii", &a, &b)) {
    return NULL;
  }
  quotient = divide(a,b, &remainder);
  return Py_BuildValue("(ii)", quotient, remainder);
}

/* Module method table */
static PyMethodDef SampleMethods[] = {
  {"gcd",  py_gcd, METH_VARARGS, "Greatest common divisor"},
  {"in_mandel", py_in_mandel, METH_VARARGS, "Mandelbrot test"},
  {"divide", py_divide, METH_VARARGS, "Integer division"},
  { NULL, NULL, 0, NULL}
};

/* Module structure */
static struct PyModuleDef samplemodule = {
  PyModuleDef_HEAD_INIT,
```

```
  "sample",             /* name of module */
  "A sample module",    /* Doc string (may be NULL) */
  -1,                   /* Size of per-interpreter state or -1 */
  SampleMethods         /* Method table */
};

/* Module initialization function */
PyMODINIT_FUNC
PyInit_sample(void) {
  return PyModule_Create(&samplemodule);
}
```

èęAçżŚåőŽè£ŹäÿłæL'ľåśŢæĺąåİŮïijŇåČŔäÿŃéİćè£ŹæăůåĹŽåżžäÿĂäÿł `setup.py` æŰĞäżűïijŽ

```
# setup.py
from distutils.core import setup, Extension

setup(name='sample',
      ext_modules=[
        Extension('sample',
                  ['pysample.c'],
                  include_dirs = ['/some/dir'],
                  define_macros = [('FOO','1')],
                  undef_macros = ['BAR'],
                  library_dirs = ['/usr/local/lib'],
                  libraries = ['sample']
                  )
        ]
)
```

äÿžäžĘæđĎåżžæIJĂçżĹçŽĎåĞ¡æŢřåžŞïijŇåŔłéIJĂçőĂå■ŢçŽĎä¡£çŤĺ `python3 buildlib.py build_ext --inplace` åŚ¡äżď'å■şåŔřïijŽ

```
bash % python3 setup.py build_ext --inplace
running build_ext
building 'sample' extension
gcc -fno-strict-aliasing -DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-
 ↪prototypes
 -I/usr/local/include/python3.3m -c pysample.c
 -o build/temp.macosx-10.6-x86_64-3.3/pysample.o
gcc -bundle -undefined dynamic_lookup
build/temp.macosx-10.6-x86_64-3.3/pysample.o \
 -L/usr/local/lib -lsample -o sample.so
bash %
```

åęĆäÿŁæL'Ăçď'žïijŇåőČäijŽåĹŽåżžäÿĂäÿłåŘ■å■ŮåŔń `sample.so` çŽĎåĚśäžńåžŞãĂĆå¡Şèćńçij ÚèŕŚåŘŐïijŇä¡ăåřśèČ¡åřĘåőČä¡IJäÿžäÿĂäÿłæĺąåİŮåŕijåĚěè£ŽæİěäžĘïijŽ

```
>>> import sample
>>> sample.gcd(35, 42)
7
>>> sample.in_mandel(0, 0, 500)
1
>>> sample.in_mandel(2.0, 1.0, 500)

0
>>> sample.divide(42, 8)
(5, 2)
>>>
```

åęĆædIJä¡ăæŸŕåIJĺWindowsæIJžåŹĺäÿŁéİćåřİèŕŢè£ŹäžŻæ■ěéłď'ïijŇåŔŕèČ¡äijŽéĄĞåĹŕåŘĎçğ■çŐŕáćČ
PythonçŽĎäžŇè£ŻåĽűåĹĘåŔŠéĂŽåÿÿä¡£çŢĺäžĘMicrosoft Visual StudioæİěædĎåżžãĂĆ
äÿžäžĘèől'è£ŹäžŻæL'l'åśŢèČ¡æ■čåÿÿåůěä¡IJïijŇä¡ăéIJĂèęĄä¡£çŢĺåŘŇæăůæĹŰåĚijåőźçŽĎåůěåĚůæİěçijŰè
åŔĆèĂČçŻÿåžŢçŽĎ PythonæŰĞæąč

èőĺèőž

åIJĺåřİèŕŢäżżä¡ŢæL'ŃåĘŹæL'l'åśŢäźŃåL'■ïijŇæIJĂåě¡èČ¡åĚĹåŔĆèĂČäÿŃPythonæŰĞæąčäÿ■çŽĎ
æL'l'åśŢåŠŇåţŇåĚěPythonèğčéĞŁåŹĺ . PythonçŽĎCæL'l'åśŢAPIå¿Ĺåď'ğïijŇåIJĺè£ŹéĞŇæŢt'äÿłåŐżèőšè£řåć
äÿ■è£Ğåŕźäž0æIJĂæäÿå£ČçŽĎéČĺåĹĘè£ŸæŸŕåŔŕäżěèőĺèőžäÿŃçŽĎãĂĆ

éęŰåĚĹïijŇåIJĺæL'l'åśŢæĺąåİŮäÿ■ïijŇä¡ăåĘźçŽĎåĞ¡æŢřéČ¡æŸŕåČŔäÿŃéİćè£ŹæăůçŽĎäÿĂäÿłæŹőéĂ

```
static PyObject *py_func(PyObject *self, PyObject *args) {
  ...
}
```

`PyObject` æŸŕäÿĂäÿłèČ¡èąĺçd'žäżżä¡ŢPythonåŕźèśąçŽĎCæŢřæ■őçśżåđŃãĂĆ
åIJĺäÿĂäÿłéńŸçžğåśĆéİćïijŇäÿĂäÿłæL'l'åśŢåĞ¡æŢřářśæŸŕäÿĂäÿłæŐěåŔŮäÿĂäÿłPythonåŕźèśą
ïijĹåIJĺ PyObject *argsäÿ■ïijL'åĚČçŽĎåźűè£ŤåŽđäÿĂäÿłæŰřPythonåŕźèśąçŽĎCåĞ¡æŢřãĂĆ
åĞ¡æŢřçŽĎ `self` åŔĆæŢřåŕźäžŐçőĂå■ŢçŽĎæL'l'åśŢåĞ¡æŢræšąæIJL'èćńä¡£çŢĺåĹřïijŇ
äÿ■è£ĞåęĆædIJä¡ăæČşåőŽäźL'æŰřçŽĎçśżæĹŰèĂĚæŸŕCäÿ■çŽĎåŕźèśąçśżåđŃçŽĎèŕİářśèČ¡æt'¿äÿŁçŢĺåIJž
éĆčäźĹ `self` ářśèČ¡åijŢçŢĺéĆčäÿłåőđä¿ŃäžĘãĂĆ

`PyArg_ParseTuple()` åĞ¡æŢřèćńçŢĺæİěářĘPythonäÿ■çŽĎåĂijè¡ñæ■ćæĹŘCäÿ■åŕźåžŢèąĺçd'žãĂĆ
åőČæŐěåŔŮäÿĂäÿłæŇĞåőŽè¿ŞåĚěăijåijŔçŽĎæăijåijŔåŇŰå■Ůçñęäÿšä¡IJäÿžè¿ŞåĚěïijŇærŤåęĆâĂIJiâĂİ
åŘŇæăůè£ŸæIJL'å■ŸæŤ¿è¡ñæ■ćåŘŐçżŞædIJçŽĎCåŘŸéĞŔçŽĎåIJřåİĂãĂĆ
åęĆædIJè¿ŞåĚěçŽĎåĂijäÿ■åŇźéĚ■è£ŹäÿłæăijåijŔåŇŰå■ŮçñęäÿšïijŇářśäijŽæŁŻåĞžäÿĂäÿłåijĆåÿÿåźűè£Ťa
éĂŽè£ĞæčĂæ§ěåźűè£ŤåŽđNULLïijŇäÿĂäÿłåŘĹéĂĆçŽĎåijĆåÿÿäijŽåIJĺèŕĈçŤĺäžčçăĄäÿ■èćńæŁŻåĞžãĂĆ

`Py_BuildValue()` åĞ¡æŢřèćńçŢĺæİěæăźæ■őCæŢřæ■őçśżåđŃåĹŻåżžPythonåŕźèśąãĂĆ
åőČåŘŇæăůæŐěåŔŮäÿĂäÿłæăijåijŔåŇŰå■ŮçñęäÿšæİěæŇĞåőŽæIJ§æIJŻçśżåđŃãĂĆ
åIJĺæL'l'åśŢåĞ¡æŢřäÿ■ïijŇåőČèćńçŢĺæİěè£ŤåŽđçżŞædIJçżŹPythonãĂĆ
`Py_BuildValue()` çŽĎäÿĂäÿłçL'źæĂğæŸŕåőČèČ¡æđĎåżžæŻt'åŁåđ'■æİĆçŽĎåŕźèśąçśżåđŃïijŇærŤåęĆ
åIJĺ `py_divide()` äžčçăĄäÿ■ïijŇäÿĂäÿłä¿Ńå■ŘæijŢçđ'žäžĘæĂŐæăůè£ŤåŽđäÿĂäÿłåĚĆçżĎãĂĆäÿ■è£Ğï

```
return Py_BuildValue("i", 34);       // Return an integer
return Py_BuildValue("d", 3.4);      // Return a double
return Py_BuildValue("s", "Hello"); // Null-terminated UTF-8 string
return Py_BuildValue("(ii)", 3, 4); // Tuple (3, 4)
```

åIJÍæĽľåśŢæÍąåİŮåžŢéČÍïijÑä¡ăäijŽåŔŚçŐřäÿĂäÿłåĞ¡æŢřèąÍïijÑæŕŤåęĆæIJňèŁĆäÿ■çŽĎ SampleMethods èąÍãĂĆ è£ŹäÿłèąÍåŔŕäžěåĹŮåĞžCåĞ¡æŢřãĂĄPythonäÿ■ä¡£çŤÍçŽĎåŘ■å■ŮãĂĄæŰĞæa æĽĂæIJĽæÍąåİŮéČ¡éIJĂèęĄæŇĞåőŽè£ŹäÿłèąÍïijÑåŽăäÿžåőČåIJÍæÍąåİŮåĹİåğŃåŇŰæŮűèęĄèćńä¡£çŤÍåĹř

æIJĂåŘŐçŽĎåĞ¡æŢř PyInit_sample() æŸŕæÍąåİŮåĹİåğŃåŇŰåĞ¡æŢřïijÑä¡Ęèŕěæ Íąåİ Ůçňňäÿ Ăæè è£ŹäÿłåĞ¡æŢřçŽĎäÿžèęĄåůěä¡IJæŸŕåIJÍèğčéĞŁåŹÍäÿ■æşÍåĘŇæÍąåİŮåŕźèśąãĂĆ

æIJĂåŘŐäÿĂäÿłèęĄçĆźéIJĂèęĄæŔŘåĞžæİěïijÑä¡£çŤÍCåĞ¡æŢřæİěæĽľåśŢPythonèęĄèĂČèŹŚçŽĎäžl ïijÍåőđéŹĚäÿŁïijÑC APIåŇĚåŘńäžĘèűĚè£Ğ500äÿłåĞ¡æŢřïijĽãĂĆä¡ăåžŤèŕěåřĘæIJňèŁĆå¡ŞåĄŽæŸŕäÿĂäÿ æŽťåďŽéńŸçžğåĘĚåőźïijÑåŔŕäžěçIJŃçIJŃ PyArg_ParseTuple() åŠŇ Py_BuildValue() åĞ¡æŢřçŽĎæŰĞæąčïijÑ çĎűåŘŐè£ŽäÿĂæ■ěæĽľåśŢåijĂãĂĆ

15.3 çijŰåĘŹæĽľåśŢåĞ¡æŢřæŞ■ä¡IJæŢřçžĎ

éŮőécŸ

ä¡ăæČşçijŰåĘŹäÿĂäÿłCæĽľåśŢåĞ¡æŢřæİěæŞ■ä¡IJæŢřçžĎïijÑåŔŕèČ¡æŸŕèćńarrayæÍąåİŮæĹŰçśžäijijÑ äÿ■è£ĞïijÑä¡ăæČşèőľä¡ăçŽĎåĞ¡æŢřæŻťåŁăéĂŽçŤÍïijÑèĂŇäÿ■æŸŕéŠĹåŕźæ§ŘäÿłçĽźåőŽçŽĎåžŞæĽĂçŤ

èğčåĘşæŰźæąĹ

äÿžäžĘèČ¡èőľæŐěåŔŮåŠŇåďĎçŘĘæŢřçžĎåĚůæIJĽåŔŕçğžæď■æĂğïijÑä¡ăéIJĂèęĄä¡£çŤÍåĹř *Buffer Protocol* . äÿŃéİćæŸŕäÿĂäÿłæĽŃåĘŹçŽĎCæĽľåśŢåĞ¡æŢřä¿Ńå■ŘïijÑ çŤÍæİěæŐěåŔŮæŢřçžĎæŢřæ■őåźűèřĆçŤÍæIJňçńäåijĂçŕĞéČÍåĹĘçŽĎ avg(double *buf, int len) åĞ¡æŢřïijŽ

```
/* Call double avg(double *, int) */
static PyObject *py_avg(PyObject *self, PyObject *args) {
  PyObject *bufobj;
  Py_buffer view;
  double result;
  /* Get the passed Python object */
  if (!PyArg_ParseTuple(args, "O", &bufobj)) {
    return NULL;
  }

  /* Attempt to extract buffer information from it */

  if (PyObject_GetBuffer(bufobj, &view,
      PyBUF_ANY_CONTIGUOUS | PyBUF_FORMAT) == -1) {
    return NULL;
  }
```

```
  if (view.ndim != 1) {
    PyErr_SetString(PyExc_TypeError, "Expected a 1-dimensional array
↪");
    PyBuffer_Release(&view);
    return NULL;
  }

  /* Check the type of items in the array */
  if (strcmp(view.format,"d") != 0) {
    PyErr_SetString(PyExc_TypeError, "Expected an array of doubles
↪");
    PyBuffer_Release(&view);
    return NULL;
  }

  /* Pass the raw buffer and size to the C function */
  result = avg(view.buf, view.shape[0]);

  /* Indicate we're done working with the buffer */
  PyBuffer_Release(&view);
  return Py_BuildValue("d", result);
}
```

äÿŃéÍćæĹŚäżňæijŤçd’žäÿŃè£ŹäÿłæĽľ’åśŢåĞ¡æŢřæŸŕåęĆä¡Ţåůěä¡IJçŽĎïjŽ

```
>>> import array
>>> avg(array.array('d',[1,2,3]))
2.0
>>> import numpy
>>> avg(numpy.array([1.0,2.0,3.0]))
2.0
>>> avg([1,2,3])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'list' does not support the buffer interface
>>> avg(b'Hello')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Expected an array of doubles
>>> a = numpy.array([[1.,2.,3.],[4.,5.,6.]])
>>> avg(a[:,2])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: ndarray is not contiguous
>>> sample.avg(a)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Expected a 1-dimensional array
>>> sample.avg(a[0])
```

```
2.0
>>>
```

èőĺèőž

åřĘäÿĂäÿłæŢřçżĎåŕźèśąäijąçżŹCåĞ¡æŢřåŔŕèČ¡æŸŕäÿĂäÿłæĽľåśŢåĞ¡æŢřåĄŽçŽĎæIJĂåÿÿèğĄçŽĎäž
å¿Ĺåď˙ŽPythonåžŤçŤĺçĺŃåžŔïijŇäžŐåŻ¿åČŔåď˙ĎçŘĘåĹřçğŚå■ęèőąçőŮïijŇéČ¡æŸŕå§žäžŐéńŸæĂğèČ¡çŽĎ
éĂŽè£ĞçijŰåĘŹèČ¡æŐěåŔŮåźűæŞ■ä¡IJæŢřçżĎçŽĎäžčçăĄïijŇä¡ăåŔŕäžěçijŰåĘŹå¿Ĺåě¡çŽĎåĚijåőźè£ŹäžŽ
èĂŇäÿ■æŸŕåŔŕłèČ¡åĚijåőźä¡äèĞłåůśçŽĎäžčçăĄãĂĆ

äžčçăĄçŽĎåĚşéŤõçĆźåIJĺäžŐ `PyBuffer_GetBuffer()` åĞ¡æŢřãĂĆ
çżŹåőŽäÿĂäÿłäżżæĎŔçŽĎPythonåŕźèśąïijŇåőČäijŽèŕŢçİĂåŐżèŐůåŔŰåžŢåśĆåĘĚå■Ÿä£ąæĄŕïijŇåőČçőĂå
1. äijăçżŹ `PyBuffer_GetBuffer()` çŽĎçĽźæőŁæăĞå£ŮçżŹåĞžäžĘæĽĂéIJĂçŽĎåĘĚå■ŸçijŞåĘšçśżåč
ä¿ŇåęĆïijŇ`PyBUF_ANY_CONTIGUOUS` èąĺçď˙žæŸŕäÿĂäÿłèĄŢçşżçŽĎåĘĚå■ŸåŇžå§§ãĂĆ

åŕźäžŐæŢřçżĎãĂĄå■ŮèŁĆå■ŮçňęäÿšåŇåĚűäžŰçśżäijijåŕźèśąèĂŇèĺĂïijŇäÿĂäÿł
`Py_buffer` çżŞæđĎä¡ŞåŇĚåŘńäžĘæĽĂæIJĽåžŢåśĆåĘĚå■ŸçŽĎä£ąæĄŕãĂĆ
åőĆåŇĚåŘńäÿĂäÿłæŇĞåŘŚåĘĚå■ŸåIJŕåİĂãĂĄåď˙ğåŕŔãĂĄåĚČçť˙ăåď˙ğåŕŔãĂĄæăijåijŔåŠŇåĚűäžŰçżĘèŁ

```
typedef struct bufferinfo {
    void *buf;              /* Pointer to buffer memory */
    PyObject *obj;          /* Python object that is the owner */
    Py_ssize_t len;         /* Total size in bytes */
    Py_ssize_t itemsize;    /* Size in bytes of a single item */
    int readonly;           /* Read-only access flag */
    int ndim;               /* Number of dimensions */
    char *format;           /* struct code of a single item */
    Py_ssize_t *shape;      /* Array containing dimensions */
    Py_ssize_t *strides;    /* Array containing strides */
    Py_ssize_t *suboffsets; /* Array containing suboffsets */
} Py_buffer;
```

æIJňèŁĆäÿ■ïijŇæĹŚäžňåŔłåĚşæşĺæŐěåŔŮäÿĂäÿłåŔŇçš¿åžęæţőçĆźæŢřæŢřçżĎä¡IJäÿžåŔĆæŢřãĂĆ
èęĄæčĂæ§ěåĚČçť˙ăæŸŕåŘęæŸŕäÿĂäÿłåŔŇçš¿åžęæţőçĆźæŢřïijŇåŔłéIJĂéłŇèŕĄ
`format` åšđæĂğæŸŕäÿ■æŸŕå■Ůçňęäÿš”d”. è£ŹäÿłäźŞæŸŕ `struct`
æĺąåİŮçŤĺæİěçijŰçăĄäžŇè£ŻåĹűæŢřæ■őçŽĎãĂĆ éĂŽåÿÿæİèèőšïijŇ`format`
åŔŕäžěæŸŕäżżä¡ŢåĚijåőź `struct` æĺąåİŮçŽĎæăijåijŔåŇŰå■ŮçňęäÿšïijŇ
åźűäÿŤåęĆæđIJæŢřçżĎåŇĚåŘńäžĘCçżŞæđĎçŽĎèŕİåőČåŔŕäžěåŇĚåŘńåď˙ŽäÿłåĂijãĂĆ
äÿĂæŮęæĹŚäžňåůšçżŔçąőåőŽäžĘåžŢåśĆçŽĎçijŞå■ŸåŇžä£ąæĄŕïijŇéĆčåŔłéIJĂèęĄçőĂå■ŢçŽĎåřĘåőČäij
åőđéŹĚäÿŁïijŇæĹŚäžňäÿ■å£ĚæŃĚå£ĆæŸŕæĂŐæăůçŽĎæŢřçżĎçśżåđŃæĹŰèĂĚåőĆæŸŕèćńäżĂäźĹåžŞåĹŹ
è£ŹäźŞæŸŕäÿžäżĂäźĹè£ŹäÿłåĞ¡æŢřèČ¡åĚijåőź `array` æĺąåİŮäźŞèČ¡åĚijåőź `numpy`
æĺąåİŮäÿ■çŽĎæŢřçżĎäžĘãĂĆ

åIJĺè£ŤåŽđæIJĂçżĹçżŞæđIJäźŃåĽ■ïijŇåžŢåśĆçŽĎçijŞåĘšåŇžèğĘåŻ¿å£Ěéążä¡£çŤĺ
`PyBuffer_Release()` éĞŁæŤ¿æŐĽãĂĆ äźŃæĽĂäżěèęĄè£ŹäÿĂæ■ěæŸŕäÿžäžĘèČ¡æ■čçąőçŽĎçőąçŘĘ

åŘŇæăůïijŇæIJňèŁĆäź§äžĚäžĚåŔŁæŸŕæijŤçď˙žäžĘæŐěåŔŮæŢřçżĎçŽĎäÿĂäÿłårĆçŽĎäžčçăĄçĽĞæő
åęĆæđIJä¡ăçIJ§çŽĎèęĄåď˙ĎçŘĘæŢřçżĎïijŇä¡ăåŔŕèČ¡äijŽçćřåĹřåď˙Žçżť˙æŢræ■őãĂĄåď˙ğæŢræ■őãĂĄäÿ■å
éĆčäźĹåřšå¿ŮåŐżå■ęæŽť˙énŸçžğçŽĎäÿIJèě£äžĘãĂĆä¡äéIJĂèęĄåŔĆèĂČåőŸæŰźæŰĞæąčæİěèŐůåŔŰæŽť

åęĆæđIJä¡ăéIJĂèęAçijŰåĘŹæűL'åŔŁåĹřæŢřçżĎåđ'ĎçŘĘçŽĎåđ'Žäÿ■æL'ľåśŢïijŇéĆčäźĹéĂŽè£ĞCytho

15.4 åIJÍCæL'ľåśŢæĺąåİŮäÿ■æŞ■ä¡IJéŽŘå¡ćæŇĞéŠĹ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłæL'ľåśŢæĺąåİŮéIJĂèęĄåđ'ĎçŘĘCçżŞæđĎä¡Şäÿ■çŽĎæŇĞéŠĹïijŇ ä¡ĘæŸŕä¡ăåŔĹäÿ■æČşæŽt'éIJšçżŞæđĎä¡Şäÿ■äżżä¡ŢåĘĚéČĺçżĘèŁĆçżŹPythonãĂĆ

èğčåĘşæŰźæąĹ

éŽŘå¡ćçżŞæđĎä¡ŞåŔŕäżěå¿Ĺåőźæ ŸŞçŽĎéĂŽè£ĞåřĘåőČäżňåŇĚècĚåIJĺèČűåŻŁåŕźèśąäÿ■æİěåđ'ĎçŘĘ èĂČèŹŚæĹŚäżňä¿Ňå■Řäżčçă Ąäÿ■çŽĎäÿŇåĹŮCäżčçăĄçL'Ğæőţïijž

```
typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

äÿŃéİćæŸŕäÿĂäÿłä¡£çŤĺèČűåŻŁåŇĚècĚPointçżŞæđĎä¡ŞåŠŇ `distance()` åĞ¡æŢřçŽĎæL'ľåśŢäżčçăĄåőđä¿ŃïijŽ

```
/* Destructor function for points */
static void del_Point(PyObject *obj) {
  free(PyCapsule_GetPointer(obj,"Point"));
}

/* Utility functions */
static Point *PyPoint_AsPoint(PyObject *obj) {
  return (Point *) PyCapsule_GetPointer(obj, "Point");
}

static PyObject *PyPoint_FromPoint(Point *p, int must_free) {
  return PyCapsule_New(p, "Point", must_free ? del_Point : NULL);
}

/* Create a new Point object */
static PyObject *py_Point(PyObject *self, PyObject *args) {

  Point *p;
  double x,y;
  if (!PyArg_ParseTuple(args,"dd",&x,&y)) {
    return NULL;
  }
  p = (Point *) malloc(sizeof(Point));
  p->x = x;
  p->y = y;
```

```
  return PyPoint_FromPoint(p, 1);
}

static PyObject *py_distance(PyObject *self, PyObject *args) {
  Point *p1, *p2;
  PyObject *py_p1, *py_p2;
  double result;

  if (!PyArg_ParseTuple(args,"OO",&py_p1, &py_p2)) {
    return NULL;
  }
  if (!(p1 = PyPoint_AsPoint(py_p1))) {
    return NULL;
  }
  if (!(p2 = PyPoint_AsPoint(py_p2))) {
    return NULL;
  }
  result = distance(p1,p2);
  return Py_BuildValue("d", result);
}
```

åIJÍPythonäÿ■åŔŕäžěåČŔäÿŃéÍćè£ŹæăůæÌěä¡£çŤÍè£ŹäžŻåĞ¡æŢřïijŽ

```
>>> import sample
>>> p1 = sample.Point(2,3)
>>> p2 = sample.Point(4,5)
>>> p1
<capsule object "Point" at 0x1004ea330>
>>> p2
<capsule object "Point" at 0x1005d1db0>
>>> sample.distance(p1,p2)
2.8284271247461903
>>>
```

èőÍèőž

èČűåŻŁåŠŇCæŇĞéŠĹçśžäijijãĂĆåIJÍåĘĚéČÍïijŇåőČäžňèŐůåŔŰäÿĂäÿłéĂŽçŤÍæŇĞéŠĹåŠŇäÿĂäÿłåŔ
PyCapsule_New() åĞ¡æŢřå¿ĹåőźæŸ§çŽĎèćńåĹŻåžžãĂĆ
åŔęåď ŰïijŇäÿĂäÿłåŔŕéĂL'çŽĎæđŘæđĎåĞ¡æŢřèČ¡èćńçżŚåőŽåĹřèČűåŻŁäÿŁïijŇçŤÍæÌěåIJÍèČűåŻŁåŕźèśąè

èęĄæŔŘåŔŰèČűåŻŁäÿ■çŽĎæŇĞéŠĹïijŇåŔŕä¡£çŤÍ PyCapsule_GetPointer()
åĞ¡æŢřåźűæŇĞåőŽåŘ■çğřãĂĆ åęĆæđIJæŔŘä¿ŻçŽĎåŘ■çğřåŠŇèČűåŻŁäÿ■åŇźéĚ■æĹŰåĚűäžŰéŤŹèŕŕåĞž

æIJňèŁĆäÿ■ïijŇäÿĂåŕźåůěåĚůåĞ¡æŢřâĂŤâĂŤ PyPoint_FromPoint()
åŠŇ PyPoint_AsPoint() èćńçŤÍæÌěåĹŻåžžåŠŇäžŐèČűåŻŁåŕźèśąäÿ■æŔŘåŔŰ-
Pointåőđä¿ŃãĂĆ åIJÍäżżä¡ŢæĽ'ľåśŢåĞ¡æŢřäÿ■ïijŇæĹŚäžňäijŽä¡£çŤÍè£ŹäžŻåĞ¡æŢřèĂŇäÿ■æŸŕçŻť'æŐěä¡
è£Źçğ■èő¿èőąä¡£å¿ŮæĹŚäžňåŔŕäžěå¿ĹåőźæŸ§çŽĎåžŤåŕźářĘæÌěåŕźPointåžŢäÿŃçŽĎåŇĚèčĚçŽĎæŻť'æŤź
ä¿ŃåęĆïijŇåęĆæđIJä¡ăåEşåőŽä¡£çŤÍåŘęåď ŰäÿĂäÿłèČűåŻŁäžĘïijŃéČčäźĹåŘłéIJÅèęĄæŻť'æŤźè£Źäÿď äÿłå

åŕźäžŐèČűåŻŁåŕźèśąäÿĂäÿłéŽ¿çĆźåIJÍäžŐåđČåIJ¿åŻđæŤűåŠŇåĘĚå■Ÿçőąçŗ ĘãĂĆ

PyPoint_FromPoint() åĞ¡æŢřæŐěåŔŮäÿĂäÿł must_free åŔĆæŢřïijŇ çŤĺæİěæŇĞåőŽå¡ŞèČűåŻŁèćńéŤĂæŕĄæŮűåžŢåśĆPoint * çżŞædĎä¡ŞæŸŕåŘęåžŤèŕěèćńåŻđæŤűãĂĆ åIJĺæ§ŘäžŻCäžčçăĄäÿ■ïijŇå¡ŠåśđéůőécŸéĂŽåÿÿå¿ĹéŽ¿èćńåđ'İ çĺŇåžŔåŚŸåŔŕäžěä¡£çŤĺ extra åŔĆæŢřæİěæŐğåĹűïijŇèĂŇäÿ■æŸŕå■ŢæŰźéİćçŽĎåĘşåőŽåđČåIJ¿åŻđæŤ èęĄæşĺæĎŔçŽĎæŸŕåŠŇçŐřæIJL'èČűåŻŁæIJL'åĚşçŽĎædŘædĎåŹĺèČ¡ä¡£çŤĺ PyCapsule_SetDestructor() åĞ¡æŢřæİěæŻť'æŤźãĂĆ

åŕźäžŐæűL'åŘŁåĹřçżŞædĎä¡ŞçŽĎCäžčçăĄèĂŇèĺĂïijŇä¡£çŤĺèČűåŻŁæŸŕäÿĂäÿłærŕŤè¿ČåŘĹçŘĘçŽĎè ä¿ŃåęĆïijŇæIJL'æŮűåĂŹä¡ååźűäÿ■åĚşå£ČæŽť'éIJšçżŞædĎä¡ŞçŽĎåĘĚéČĺä£ąæĄŕæĹŮèĂĚårĘåĚűè¡ñæ■ćæ éĂŽè£Ğä¡£çŤĺèČűåŻŁïijŇä¡ååŘŕäžěåIJĺåőČäÿŁéİćæŤ¿äÿĂäÿłè¡žéĞŔçžğçŽĎåŇĚèčĚåŹĺïijŇçĎűåŘŐårĘåőČ

15.5 äžŐæL'ľåśŢæĺąåİŮäÿ■åőŽäźL'åŠŇårijåĞžCçŽĎAPI

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłCæL'ľåśŢæĺąåİŮïijŇåIJĺåĘĚéČĺåőŽäźL'äžĘå¿Ĺåď'ŽæIJL'çŤĺçŽĎåĞ¡æŢřïijŇä¡ăæČşåřĘå APIä¿ŻåĚűäžŰåIJřæŰźä¡£çŤĺãĂĆ ä¡ăæČşåIJĺåĚűäžŰæL'ľåśŢæĺąåİŮäÿ■ä¡£çŤĺè£ŹäžŻåĞ¡æŢřïijŇä¡ĘæŸŕäÿ åźűäÿŤéĂŽè£ĞCçijŰèŕŠåŹĺ/éŞ¿æŐěåŹĺæİěåĄŽçIJŃäÿŁåŐžçL'źåĹńåď'■æİĆïijĹæĹŰèĂĚäÿ■åŔŕèČ¡åĄŽåĹř

èğčåĘşæŰźæąĹ

æIJňèŁĆäÿżèęĄéŮőécŸæŸŕåęĆä¡Ţåď'ĎçŘĘ15.4åřŔèŁĆäÿ■æŔŘåĹřçŽĎPointåŕźèśąãĂĆäžŤçżĘåŻđäÿ/

```
/* Destructor function for points */
static void del_Point(PyObject *obj) {

  free(PyCapsule_GetPointer(obj,"Point"));
}

/* Utility functions */
static Point *PyPoint_AsPoint(PyObject *obj) {
  return (Point *) PyCapsule_GetPointer(obj, "Point");
}

static PyObject *PyPoint_FromPoint(Point *p, int must_free) {
  return PyCapsule_New(p, "Point", must_free ? del_Point : NULL);
}
```

çŐřåIJĺçŽĎéŮőécŸæŸŕæĂŐæăůårĘ PyPoint_AsPoint() åŠŇ Point_FromPoint() åĞ¡æŢřä¡IJäÿžAPIåŕijåĞžïijŇ è£ŹæăůåĚűäžŰæL'ľåśŢæĺąåİŮèČ¡ä¡£çŤĺåźűéŞ¿æŐěåőČäžňïijŇærŕŤåęĆåęĆædIJä¡ăæIJL'åĚűäžŰæL'ľåśŢäź§

èęĄèğčåĘşè£ŹäÿłéŮőécŸïijŇéęŰåĚĹèęĄäÿž sample æL'ľåśŢåĘŹäÿłæŰřçŽĎåď'ť'æŰĞäžűåŘ■åŔń pysample.h ïijŇåęĆäÿŃïijŽ

```
/* pysample.h */
#include "Python.h"
#include "sample.h"
```

```
#ifdef __cplusplus
extern "C" {
#endif

/* Public API Table */
typedef struct {
  Point *(*aspoint)(PyObject *);
  PyObject *(*frompoint)(Point *, int);
} _PointAPIMethods;

#ifndef PYSAMPLE_MODULE
/* Method table in external module */
static _PointAPIMethods *_point_api = 0;

/* Import the API table from sample */
static int import_sample(void) {
  _point_api = (_PointAPIMethods *) PyCapsule_Import("sample._point_
↪api",0);
  return (_point_api != NULL) ? 1 : 0;
}

/* Macros to implement the programming interface */
#define PyPoint_AsPoint(obj) (_point_api->aspoint)(obj)
#define PyPoint_FromPoint(obj) (_point_api->frompoint)(obj)
#endif

#ifdef __cplusplus
}
#endif
```

è£ŹéĞŇæIJĂéĞ■èęĄçŽĎéČĺåĹĘæŸŕåĞ¡æŢřæŇĞéŠĹèąĺ `_PointAPIMethods` . åőČäijŽåIJĺåŕijåĞžæĺąåİŮæŮűèćńåĹĺåğŃåŇŰïijŇçĎűåŘŐåŕijåĚěæĺąåİŮæŮűèćńæ§ěæĽ¿åĹřãĂĆä£őæŤźåŐ§åğŃçŽĎæĽ'ľåśŢæĺąåİŮæİěåąńåĚĚèąĺæäijåźűåřĘåőČåČŔäÿŃéİćè£ŹæăůåŕijåĞžïijŽ

```
/* pysample.c */

#include "Python.h"
#define PYSAMPLE_MODULE
#include "pysample.h"

...
/* Destructor function for points */
static void del_Point(PyObject *obj) {
  printf("Deleting point\n");
  free(PyCapsule_GetPointer(obj,"Point"));
}

/* Utility functions */
static Point *PyPoint_AsPoint(PyObject *obj) {
  return (Point *) PyCapsule_GetPointer(obj, "Point");
```

```
}

static PyObject *PyPoint_FromPoint(Point *p, int free) {
  return PyCapsule_New(p, "Point", free ? del_Point : NULL);
}

static _PointAPIMethods _point_api = {
  PyPoint_AsPoint,
  PyPoint_FromPoint
};
...

/* Module initialization function */
PyMODINIT_FUNC
PyInit_sample(void) {
  PyObject *m;
  PyObject *py_point_api;

  m = PyModule_Create(&samplemodule);
  if (m == NULL)
    return NULL;

  /* Add the Point C API functions */
  py_point_api = PyCapsule_New((void *) &_point_api, "sample._point_
↪api", NULL);
  if (py_point_api) {
    PyModule_AddObject(m, "_point_api", py_point_api);
  }
  return m;
}
```

æIJĂåŘŐïijŇäÿŃéİćæŸŕäÿĂäÿłæŰřçŽĎæĽľåśŢæĺąåİŮä¿Ńå■ŘïijŇçŤĺæİěåŁäè¡¡åźűä¡£çŤĺè£ŹäžŻAPIå

```
/* ptexample.c */

/* Include the header associated with the other module */
#include "pysample.h"

/* An extension function that uses the exported API */
static PyObject *print_point(PyObject *self, PyObject *args) {
  PyObject *obj;
  Point *p;
  if (!PyArg_ParseTuple(args,"O", &obj)) {
    return NULL;
  }

  /* Note: This is defined in a different module */
  p = PyPoint_AsPoint(obj);
  if (!p) {
    return NULL;
```

```
  }
  printf("%f %f\n", p->x, p->y);
  return Py_BuildValue("");
}

static PyMethodDef PtExampleMethods[] = {
  {"print_point", print_point, METH_VARARGS, "output a point"},
  { NULL, NULL, 0, NULL}
};

static struct PyModuleDef ptexamplemodule = {
  PyModuleDef_HEAD_INIT,
  "ptexample",           /* name of module */
  "A module that imports an API",  /* Doc string (may be NULL) */
  -1,                 /* Size of per-interpreter state or -1 */
  PtExampleMethods       /* Method table */
};

/* Module initialization function */
PyMODINIT_FUNC
PyInit_ptexample(void) {
  PyObject *m;

  m = PyModule_Create(&ptexamplemodule);
  if (m == NULL)
    return NULL;

  /* Import sample, loading its API functions */
  if (!import_sample()) {
    return NULL;
  }

  return m;
}
```

çijŰèŕŚè£ŹäÿłæŰřæĺąåİŮæŮűïijÑä¡ăçŤŽèĞşäÿ■éIJÄèęĄåŐżèĂČèŹŚæĂŐæăůåřĘåĞ¡æŢřåžŞæĹŰäżčçà ä¿ŇåęĆïijÑä¡ääŔŕäżěåČŔäÿŇéİćè£ŹæăůåĹŻåžžäÿĂäÿłçőĂå■ŢçŽĎ `setup.py` æŰĞäżűïijŽ

```
# setup.py
from distutils.core import setup, Extension

setup(name='ptexample',
      ext_modules=[
        Extension('ptexample',
                  ['ptexample.c'],
                  include_dirs = [],  # May need pysample.h␣
↪directory
                  )
        ]
)
```

åęĆædIJäÿĂåĹĞæ■čåÿÿïijÑä¡ăäijŽåŔŚçŐřä¡ăçŽĎæŰřæĽľåśŢåĞ¡æŢřèČ¡åŠÑåőŽäźĽåIJĺåĚűäżŰæĺąåİ
APIåĞ¡æŢřäÿĂèţůè£ŘèąŇçŽĎå¿Ĺåě¡ãĂĆ

```
>>> import sample
>>> p1 = sample.Point(2,3)
>>> p1
<capsule object "Point *" at 0x1004ea330>
>>> import ptexample
>>> ptexample.print_point(p1)
2.000000 3.000000
>>>
```

èőĺèőž

æIJňèŁĆå§žäžŐäÿĂäÿłåĽ■æŔŘåřśæŸŕïijŇèČűåŻŁåŕźèśąèČ¡èŐůåŔŰäżżä¡Ţä¡ăæČşèęĄçŽĎåŕźèśąçŽĎ
è£ŹæăůçŽĎèŕİïijŇåőŽäźĽæĺąåİŮäijŽåąńåĚĚäÿĂäÿłåĞ¡æŢřæŇĞéŠĹçŽĎçżŞæđĎä¡ŞïijŇå
ä¿ŃåęĆ `sample._point_api`.

åĚűäżŰæĺąåİŮèČ¡åď§åIJĺåŕijåĚěæŮűèŐůåŔŰåĹřè£ŹäÿłåśđæĂğåźűæŔŘåŔŰåžŢåśĆçŽĎæŇĞéŠĹãĂĆ
äžŃåőđäÿŁïijŇPythonæŔŘä¿ŻäžĘ `PyCapsule_Import()`
åůěåĚůåĞ¡æŢřïijŇäÿžäžĘåőŇæĹŘæĽĂæIJL'çŽĎæ■ěélď'ãĂĆ
ä¡ăåŘłéIJĂæŔŘä¿ŻåśđæĂğçŽĎåŘ■å■Ůå■şåŕŕïijĹæŕŤåęĆsample._point_apiïijL'ïijŇçĎűåŘŐäžŰåřśäijŽäÿĂ

åIJĺåřĘèćńåŕijåĞžåĞ¡æŢřåŔŸäÿžåĚűäżŰæĺąåİŮäÿ■æŹőéĂŽåĞ¡æŢřæŮűïijŇæIJL'äÿĂäžŻCçijŰçÍNéŹů
åIJÍ `pysample.h` æŰĞäżűäÿ■ïijŇäÿĂäÿł `_point_api`
æŇĞéŠĹèćńçŤÍæİěæŇĞåŘŚåIJĺåŕijåĞžæĺąåİŮäÿ■èćńåĹİåğŃåŇŰçŽĎæŰźæşŢèąĺãĂĆ
äÿĂäÿłçŻÿåĚşçŽĎåĞ¡æŢř `import_sample()` èćńçŤÍæİěæŇĞåŘŚèČűåŽŁåŕijåĚěźűåĹİåğŃåŇŰè£Źäÿłæ
è£ŹäÿłåĞ¡æŢřå£Ěéążåİĺäżżä¡ŢåĞ¡æŢřèćńä¡£çŤĺäźŃåĽ■èćńèřĆçŤÍãĂĆéĂŽåÿÿæİèèőšïijŇåőČäijŽåIJÍæĺąåİ
æIJĂåŘŐïijŇCçŽĎéćĎåď'ĎçŘĘåőŘèćńåőŽäźĽïijŇèćńçŤÍæİěéĂŽè£ĞæŰźæşŢèąĺåŐżåĹĘåŔŚè£ŹäžŻAPIåĞ
çŤÍæĹůåŘłéIJĂèęĄä¡£çŤÍè£ŹäžŻåŐ§åğŃåĞ¡æŢřåŘ■çğřå■şåŕŕïijŇäÿ■éIJĂèęĄéĂŽè£ĞåőŘåŐżäžĘèğčåĚűä

æIJĂåŘŐïijŇè£ŸæIJL'äÿĂäÿłéĞ■èęĄçŽĎåŐ§åŽäèől'ä¡ăåŐżä¡£çŤÍè£ŹäÿłæŁĂæIJŕæİěéŞ¿æŐěæĺąåİŮâ
åęĆædIJä¡ääÿ■æČşä¡£çŤÍæIJňæIJžçŽĎæŁĂæIJŕïijŇéĆčä¡ăåřśå£Ěéążä¡£çŤÍåĚśäžńåžŞçŽĎéńŸçžğçL'źæĂğå
ä¿ŃåęĆïijŇåřĘäÿĂäÿłæŹőéĂŽçŽĎAPIåĞ¡æŢřæŤ¿åĚěäÿĂäÿłåĚśäžńåžŞåźűçąőä£ÍæL'ĂæIJL'æL'ľåśŢæĺąåİŮ
è£Źçğ■æŰźæşŢçąőåőđåŘŕèąŇïijŇä¡ĘæŸŕåőĆçŽÿåŕźçźĄçŘŘïijŇçL'źåĹńæŸŕåIJĺåď'ğåđŃçşżçžş§äÿ■ãĂĆ
æIJňèŁĆæijŤçd'žäžĘåęĆä¡ŢéĂŽè£ĞPythonçŽĎæŹőéĂŽåŕijåĚěæIJžåĹűåŠŃäžĚäżĚåĞääÿłèČűåŻŁèřĆçŤÍæİ
åŕźäžŐæĺąåİŮçŽĎçijŰèŕŚïijŇä¡ăåŘłéIJĂèęĄåőŽäźĽåd'ť'æŰĞäžűïijŇèĂŇäÿ■éIJĂèęĄèĂČèŹŠåĞ¡æŢřåžŞçŽ

æŻť'åd'ŽåĚşäžŐåĹl'çŤÍC APIæİěædĎéĂäæĽľåśŢæĺąåİŮçŽĎä£qæĄŕåŘŕäžěåŘĆèĂČ
PythonçŽĎæŰĞæąč

15.6 äžŐCèŕ■èÍÄäÿ■èřĈçŤÍPythonäžčçăĄ

éŮőécŸ

ä¡ăæČşåIJÍCäÿ■åőĽåĚÍçŽĎæĽğèąŇæ§ŘäÿłPythonèřĈçŤÍåźűè£ŤåŻđçżŞædIJçżŹCãĂĆ
ä¿ŃåęĆïijŇä¡ăæČşåIJÍCèŕ■èÍÄäÿ■ä¡£çŤÍæ§ŘäÿłPythonåĞ¡æŢřä¡IJäÿžäÿĂäÿłåŻđèřČãĂĆ

èğčåĘşæŰźæąĹ

åIJÍCèŕ■èÍĂäÿ■èřČçŤÍPythonéÍđåÿÿçőĂå■ŢïijŇäÿ■è£Ğèő¿èőąåĹřäÿĂäžŻåřŔçł■éŮÍãĂĆäÿŃéÍćçŽĎCäżčçăĄåŚŁèŕL'ä¡ăæĂŐæăůåőL'åĚÍçŽĎèřČçŤÍïijŽ

```
#include <Python.h>

/* Execute func(x,y) in the Python interpreter.  The
   arguments and return result of the function must
   be Python floats */

double call_func(PyObject *func, double x, double y) {
  PyObject *args;
  PyObject *kwargs;
  PyObject *result = 0;
  double retval;

  /* Make sure we own the GIL */
  PyGILState_STATE state = PyGILState_Ensure();

  /* Verify that func is a proper callable */
  if (!PyCallable_Check(func)) {
    fprintf(stderr,"call_func: expected a callable\n");
    goto fail;
  }
  /* Build arguments */
  args = Py_BuildValue("(dd)", x, y);
  kwargs = NULL;

  /* Call the function */
  result = PyObject_Call(func, args, kwargs);
  Py_DECREF(args);
  Py_XDECREF(kwargs);

  /* Check for Python exceptions (if any) */
  if (PyErr_Occurred()) {
    PyErr_Print();
    goto fail;
  }

  /* Verify the result is a float object */
  if (!PyFloat_Check(result)) {
    fprintf(stderr,"call_func: callable didn't return a float\n");
    goto fail;
  }

  /* Create the return value */
  retval = PyFloat_AsDouble(result);
  Py_DECREF(result);

  /* Restore previous GIL state and return */
```

```
  PyGILState_Release(state);
  return retval;

fail:
  Py_XDECREF(result);
  PyGILState_Release(state);
  abort();   // Change to something more appropriate
}
```

èęĄä¡£çŤĺè£ŹäÿłåĞ¡æŢřïijŇä¡ăéIJĂèęĄèŐůåŔŰäijăéĂŠè£ĞæİěçŽĎæ§Řäÿłåůšå■ŸåIJĺPythonèřČçŤĺçŽ
æIJL'å¿Ĺåd'Žçğ■æŰźæşŢåŔŕäżěèől'ä¡ăè£ŹæăůåĄŽïijŇ æŕŤåęĆářĘäÿĂäÿłåŔŕèřČçŤĺårźèśąäijăçżŹäÿĂäÿłæL'

äÿŃéİćæŸŕäÿĂäÿłçőĂå■Ţä¿Ńå■ŘçŤĺæİěæŐl'éěřäžŐäÿĂäÿłåţŇåĚěçŽĎPythonèğčéĞŁåŹĺäÿ■èřČçŤĺäÿ

```
#include <Python.h>

/* Definition of call_func() same as above */
...

/* Load a symbol from a module */
PyObject *import_name(const char *modname, const char *symbol) {
  PyObject *u_name, *module;
  u_name = PyUnicode_FromString(modname);
  module = PyImport_Import(u_name);
  Py_DECREF(u_name);
  return PyObject_GetAttrString(module, symbol);
}

/* Simple embedding example */
int main() {
  PyObject *pow_func;
  double x;

  Py_Initialize();
  /* Get a reference to the math.pow function */
  pow_func = import_name("math","pow");

  /* Call it using our call_func() code */
  for (x = 0.0; x < 10.0; x += 0.1) {
    printf("%0.2f %0.2f\n", x, call_func(pow_func,x,2.0));
  }
  /* Done */
  Py_DECREF(pow_func);
  Py_Finalize();
  return 0;
}
```

èęĄæđĎåżžä¿Ńå■ŘäżčçăĄïijŇä¡ăéIJĂèęĄçijŰèŕŚCåźűårĘåőČéŞ¿æŐěåĹřPythonèğčéĞŁåŹĺãĂĆ
äÿŃéİćçŽĎMakefileåŔŕäżěæŢŹä¡ăæĂŐæăůåĄŽïijĹäÿ■è£ĞåIJĺä¡ăæIJžåŹĺäÿŁéİćéIJĂèęĄäÿĂäžŻéĚ■ç¡őïijL'

```
all::
        cc -g embed.c -I/usr/local/include/python3.3m \
          -L/usr/local/lib/python3.3/config-3.3m -lpython3.3m
```

çijŰèŕŠåźűè£ŘèąŇäijŽäžğçŤ§çśżäijijäÿŃéİćçŽĎè¿ŞåĞžïijŽ

```
0.00 0.00
0.10 0.01
0.20 0.04
0.30 0.09
0.40 0.16
...
```

äÿŃéİćæŸŕäÿĂäÿłçĺ■å¿őäÿ■åŘŇçŽĎä¿Ńå■ŘïijŇåśŢçd'žäžĘäÿĂäÿłæL'l'åśŢåĞ¡æŢřïijŇ åőČæŐěåŔŮäÿĂäÿłåŔŕèřČçŤĺåŕźèśąåŠŇåĚűäžŰåŔĆæŢřïijŇåźűårĘåőČäžñäijäéĂŠçżŹ `call_func()` æİěåĄŽæţŃèŕŢïijŽ

```
/* Extension function for testing the C-Python callback */
PyObject *py_call_func(PyObject *self, PyObject *args) {
  PyObject *func;

  double x, y, result;
  if (!PyArg_ParseTuple(args,"Odd", &func,&x,&y)) {
    return NULL;
  }
  result = call_func(func, x, y);
  return Py_BuildValue("d", result);
}
```

ä¡£çŤĺè£ŹäÿłæL'l'åśŢåĞ¡æŢřïijŇä¡ăèęĄåČŔäÿŃéİćè£ŹæăůæţŃèŕŢåőČïijŽ

```
>>> import sample
>>> def add(x,y):
...     return x+y
...
>>> sample.call_func(add,3,4)
7.0
>>>
```

èőĺèőž

åęĆædIJä¡ăåIJĺCèŕ■èĺĂäÿ■èřČçŤĺPythonïijŇèęĄèőřä¡ŔæIJĂéĞ■èęĄçŽĎæŸŕCèŕ■èĺĂäijŽæŸŕäÿżä¡ŞãĂ äźş§åřśæŸŕèŕťïijŇCèŕ■èĺĂèť§èť čædĎéĂăåŔĆæŢřãĂĄèřČçŤĺPythonåĞ¡æŢřãĂĄæčĂæ§ěåijĆåÿÿãĂĄæčĂæ

ä¡IJäÿžçňňäÿĂæ■ěïijŇä¡ăå£ĚéąžåĕĺæIJL'äÿĂäÿłèąĺçd'žä¡ăåřĘèęĄèřČçŤĺçŽĎPythonåŔŕèřČçŤĺåŕźèśąãĂ è£ŹåŔŕäžěæŸŕäÿĂäÿłåĞ¡æŢřãĂĄçśžãĂĄæŰźæşŢãĂĄåĘĚç¡őæŰźæşŢæĹŰåĚűäžŰäżżæĎŔåőđçŐřäžĘ `__call__()` æŞ■ä¡IJçŽĎäÿIJèě£ãĂĆ äÿžäžĘçąőä£ÍæŸŕåŔŕèřČçŤĺçŽĎïijŇåŔŕäžěåČŔäÿŃéİćçŽĎäžčçăĄè `PyCallable_Check()` åĄŽæčĂæ§ěijŽ

```
double call_func(PyObject *func, double x, double y) {
  ...
  /* Verify that func is a proper callable */
  if (!PyCallable_Check(func)) {
    fprintf(stderr,"call_func: expected a callable\n");
    goto fail;
  }
  ...
```

åIJÍCäžčçăĄéĞŇåď ĎçŘĘéŤŹèŕŕä¡ăéIJĂèęĄæăijåď ŰçŽĎåřŔå£ČãĂĆäÿĂèĹňæİěèőšïijŇä¡ăäÿ■èČ¡äžĚä
éŤŹèŕŕåžŤèŕěä¡£çŤÍCäžčçăĄæŰźåijŔæİěèćńåď ĎçŘĘãĂĆåIJÍè£ŹéĞŇïijŇæĹŚäžňæĽŞçőŮåřĘåŕźéŤŹèŕŕçŽĎ
abort() çŽĎéŤŹèŕŕåď ĎçŘĘåŹÍãĂĆ åőČäijŽçžŞæİ§æŐĽæŢť äÿłçÍŃåžŔïijŇåIJÍçIJ§åőđçŐŕåćČäÿŃéİćä¡ăå
ä¡ăèęĄèőŕä¡ŔçŽĎæŸŕåIJÍè£ŹéĞŇCæŸŕäÿżèğŠïijŇåŻăæ■ď åźűæšąæIJL'èů§æŁŻåĞžåijĆåÿÿçŻÿåŕźåžŤçŽĎæ
éŤŹèŕŕåď ĎçŘĘæŸŕä¡ăåIJÍçijŰçÍŃæŮűå£ĚéążèęĄèĂČèŹŚçŽĎäžŃæČĚãĂĆ

èřČçŤÍäÿĂäÿłåĞ¡æŢřçŻÿåŕźæİěèőšå¿ĹçőĂå■ŢâĂŤâĂŤåŔłéIJĂèęĄä¡£çŤÍ
PyObject_Call() ïijŇ äijăäÿĂäÿłåŔŕèŕČçŤÍåŕźèśąçżŹåőČãĂĄäÿĂäÿłåŔĆæŢřåĚČçżĎåŠŇäÿĂäÿłåŔŕéĂ
èęĄæđĎåżžåŔĆæŢřåĚČçżĎæĹŰå■ŮåĚÿïijŇä¡ăåŔŕäžěä¡£çŤÍ Py_BuildValue()
,åęĆäÿŃïijŽ

```
double call_func(PyObject *func, double x, double y) {
  PyObject *args;
  PyObject *kwargs;

  ...
  /* Build arguments */
  args = Py_BuildValue("(dd)", x, y);
  kwargs = NULL;

  /* Call the function */
  result = PyObject_Call(func, args, kwargs);
  Py_DECREF(args);
  Py_XDECREF(kwargs);
  ...
```

åęĆæđIJæšąæIJL'åĚşéŤőå■ŮåŔĆæŢřïijŇä¡ăåŔŕäžěäijăéĂŠNULLãĂĆå¡Şä¡ăèęĄèřČçŤÍåĞ¡æŢřæŮűïijŇ
éIJĂèęĄçąőä£Ìä¡£çŤÍäžĘ Py_DECREF() æĹŰèĂĚ Py_XDECREF() æÿĚçŘĘåŔĆæŢřãĂĆ
çňňäžŇäÿłåĞ¡æŢřçŻÿåŕźåőL'åĚÍçĆźïijŇåŻääÿžåőČåĚĄèőÿäijăéĂŠNULLæŇĞéŠĹïijÍçŻť'æŐěå£¡çŢěåőČïijÍ
è£Źäź§æŸŕäÿžäžĂäźĹæĹŚäžňä¡£çŤÍåőČæÌěæÿĚçŘĘåŔŕéĂĽ'çŽĎåĚşéŤőå■ŮåŔĆæŢřãĂĆ

èřČçŤÍäÿĞPythonåĞ¡æŢřäźŃåŘŐïijŇä¡ăå£Ěéążæč Ăæ§ěæŸŕåŘęæIJL'åijĆåÿÿåŔŚçŤ§ãĂĆ
PyErr_Occurred() åĞ¡æŢřåŔŕèćńçŤÍæİěåĄŽè£Źäžűäž ŃãĂĆ
åŕźåŕźäžŐåijĆåÿÿçŽĎåď ĎçŘĘåŕšæIJL'çĆźžżçČęäžĘïijŇçŤśäžŐæŸŕçŤÍCèŕ■èÍÃåĘŹçŽĎïijŇä¡ăæšąæIJL'åČ
åŻăæ■ď'ïijŇä¡ăå£ĚéążèęĄèő¿ç¡őäÿĂäÿłåijĆåÿÿçŁűæĂĄçĂąïijŇæĽ'Şå■ŕåijĆåÿÿä£ąæĄŕæĹŰåĚűäžŰçŻÿåžŤ
åIJÍè£ŹéĞŇïijŇæĹŚäžňéĂĽ'æŃľäžĘçőĂå■ŢçŽĎ abort()
æİěåď ĎçŘĘãĂĆåŘęåď ŰïijŇäijăçż§CçÍŃåžŔåŚŸåŔŕèČ¡äijŽçŻť'æŐěěőľçÍŃåžŔåěŤæžČãĂĆ

```
...
/* Check for Python exceptions (if any) */
if (PyErr_Occurred()) {
```

```
  PyErr_Print();
  goto fail;
}
...
fail:
  PyGILState_Release(state);
  abort();
```

äżŐèřČçŤÍPythonåĞ¡æŢřçŽĎè£ŤåŻđåĂijäÿ■æŔŘåŔŰä£ąæĄŕéĂŽåÿÿèęĄè£Żèą
èęĄè£ŹæăůåĄŽçŽĎèŕĬïijŇä¡ăå£Ěéążä¡£çŤÍPythonåŕźèśąåśĆäÿ■çŽĎåĞ¡æŢřãĂĆ
åIJÍè£ŹéĞŇæĹŚäżňä¡£çŤÍäžĘ `PyFloat_Check()` åŠŇ `PyFloat_AsDouble()`
æİěæčĂæ§ěåŠŇæŔŘåŔŰPythonæţőçĆźæŢřãĂĆ

æIJĂåŘŐäÿĂäÿłéŮőécŸæŸŕåŕźäžŐPythonåĚÍåśĂéŤĄçŽĎçőąçŘĘãĂĆ
åIJÍCèŕ■èÍĂäÿ■èő£éŮőPythonçŽĎæŮűůåĂŹïijŇä¡ăéIJĂèęĄçąőä£ĺGILèćńæ■čçąőçŽĎèŐůåŘŰåŠŇéĞŁæŤ¿äž
äÿ■çĎűçŽĎèŕĬïijŇåŘŕèČ¡äijŽåŕijèĞť èğčéĞŁåŹÍè£ŤåŻđéŤŹèŕŕæŢřæ■őæĹŰèĂĚçŻť æŐěåěŤæžČãĂĆ
èřČçŤÍ `PyGILState_Ensure()` åŠŇ `PyGILState_Release()`
åŘŕäžěçąőä£ÍäÿĂåĹĞéČ¡èČ¡æ■čåÿÿãĂĆ

```
double call_func(PyObject *func, double x, double y) {
  ...
  double retval;

  /* Make sure we own the GIL */
  PyGILState_STATE state = PyGILState_Ensure();
  ...
  /* Code that uses Python C API functions */
  ...
  /* Restore previous GIL state and return */
  PyGILState_Release(state);
  return retval;

fail:
  PyGILState_Release(state);
  abort();
}
```

äÿĂæŮęè£ŤåŻđijŇ`PyGILState_Ensure()` åŘŕäžěçąőä£ÍèřČçŤÍçž£çÍŃçŃňå■ăPythonèğčéĞŁåŹÍãĂ
åřśçőŮCäżčçăĄè£ŘèąŇäžŐåŔęåď ŰäÿĂäÿłèğčéĞŁåŹÍäÿ■ç§ěéĄŞçŽĎçž£çÍŃäź§æšąäžŇãĂĆ
è£ŹæŮűůåĂŹïijŇCäżčçăĄåŘŕäžěèĞłçŤśçŽĎä¡£çŤÍäżżä¡ŢåőČæČşèęĄçŽĎPython C-API
åĞ¡æŢřãĂĆ èřČçŤÍæĹŘåŁ§åŘŐïijŇPyGILState_Release()èćńçŤÍæÍěèőšèğčéĞŁåŹÍæĄćåď ■åĽřåŐ§ågŇçŁűa

èęĄæşÍæĎŔçŽĎæŸŕæŕŔäÿĂäÿł `PyGILState_Ensure()`
èřČçŤÍå£ĚéążèůçİĂäÿĂäÿłåŇźéĚ■çŽĎ `PyGILState_Release()`
èřČçŤÍâĂŤâĂŤå■şä¿£æIJL'éŤŹèŕŕåŔŚçŤ§ãĂĆ åIJÍè£ŹéĞŇïijŇæĹŚäżňä¡£çŤÍäÿĂäÿł `goto`
èŕ■åŘěçIJŇäÿŁåŐżæŸŕäÿłåŘŕæĂŢçŽĎèő¿ëőąïijŇ ä¡ĘæŸŕåőđéŹĚäÿŁæĹŚäżňä¡£çŤÍåőČæİèèőšæŐğåĹűæİČè
åIJÍ `fail:` æăĞç■¿åŘŐéİćçŽĎäžčçăĄåŠŇPythonçŽĎ `fianl:`
åİŮçŽĎçŤÍéĂŤæŸŕäÿĂæăůçŽĎãĂĆ

åęĆæđIJä¡ăä¡£çŤÍæL'ĂæIJL'è£ŹäžŻçžęåőŽæİěçijŰåĘŹCäżčçăĄïijŇåŇĚæŃňåŕźGILçŽĎçőąçŘĘãĂĄåij
ä¡ääijŽåŔŚçŐŕäžŐCèŕ■èÍĂäÿ■èřČçŤÍPythonèğčéĞŁåŹÍæŸŕåŘŕéİăçŽĎâĂŤâĂŤåŕśçőŮåĘ■åď ■æİĆçŽĎçÍŃåž

15.7 äžŐCæĽľåśŢäÿ■éĞŁæŤ¿åĚĺåśĂéŤĄ

éŮőécŸ

ä¡ăæČşèőľCæĽľåśŢäžčçăĄåŠŇPythonèğčéĞŁåŹĺäÿ■çŽĎåĚűäžŰè£ŻçÍŃäÿĂèţůæ■čçąőçŽĎæĽğèąŇïij
éĆčäźĹä¡ăåřśéIJĂèęĄåŐžéĞŁæŤ¿åźűéĞ■æŰřèŐůåŔŰåĚĺåśĂèğčéĞŁåŹĺéŤĄïijĹGILïijĽãĂĆ

èğčåĘşæŰźæąĹ

åIJÍCæĽľåśŢäžčçăĄäÿ■ïijŇGILåŔŕäžěéĂŽè£ĞåIJĺäžčçăĄäÿ■æŔŠåĚěäÿŃéİćè£ŹæăůçŽĎåőŔæİěéĞŁæ

```
#include "Python.h"
...

PyObject *pyfunc(PyObject *self, PyObject *args) {
   ...
   Py_BEGIN_ALLOW_THREADS
   // Threaded C code.  Must not use Python API functions
   ...
   Py_END_ALLOW_THREADS
   ...
   return result;
}
```

èőÍèőž

åŘłæIJL'å¡Şä¡ăçąőä£ÍæšąæIJL'Python C APIåĞ¡æŢřåIJÍCäÿ■æĽğèąŇçŽĎæŮűåĂŹä¡ăæĽ■èČ¡åőĽ'åĚĺçŽ
GILéIJĂèęĄèćńéĞŁæŤ¿çŽĎåÿÿèğĄçŽĎåIJžæŹŕæŸŕåIJÍèőąçőŮårĘéŹĘåđŃäžčçăĄäÿ■éIJĂèęĄåIJÍCæŢřçżĎ
æĹŰèĂĚæŸŕêęĄæĽğèąŇéŸżåąđçŽĎI/OæŞ■ä¡IJæŮűïijĹæŕŤåęĆåIJĺäÿĂäÿłæŰĞäžűæŔŘè£řçňęäÿŁèŕżåŔŰa

å¡ŞGILèćńéĞŁæŤ¿åŘŐïijŇåĚűäžŰPythonçž£çÍŃæĽ■èćńåĚĄèőÿåIJÍèğčéĞŁåŹĺäÿ■æĽğèąŇãĂĆ
`Py_END_ALLOW_THREADS` åőŔäijŽéŸżåąđæĽğèąŇçŻťåĹřèřČçŤĺçž£çÍŃéĞ■æŰřèŐůåŔŰäžĘGILãĂĆ

15.8 CåŠŇPythonäÿ■çŽĎçž£çÍŃæůúçŤÍ

éŮőécŸ

ä¡ăæIJL'äÿĂäÿłçÍŃåžŔéIJĂèęĄæůůåŘĹä¡£çŤÍCãĂĄPythonåŠŇçž£çÍŃïijŇ
æIJL'äžŻçž£çÍŃæŸŕåIJÍCäÿ■åĹŻåžžçŽĎïijŇèűĚåĞžäžĘPythonèğčéĞŁåŹÍçŽĎæŐğåĹűèŇČåŻť ãĂĆ
åźűäÿŤäÿĂäžŻçž£çÍŃè£Ÿä¡£çŤĺäžĘPython C APIäÿ■çŽĎåĞ¡æŢřãĂĆ

èğčåĘşæŰźæąĹ

åęĆædIJä¡ăæČşåřĘCãĂĄPythonåŠŇçž£çÍŃæůůåŘĹåIJÍäÿĂèţůïijŇä¡ăéIJĂèęĄçąőä£Íæ■čçąőçŽĎåĹİåğŇ
èęĄæČşè£ŹæăůåĄŽïijŇåŔŕäžěåřĘäÿŃåĹŮäžčçăĄæŤ¿åĹřä¡ăçŽĎCäžčçăĄäÿ■åźűçąőä£ÍåőČåIJÍäžżä¡Ţçž£çÍŃ

```
#include <Python.h>
  ...
  if (!PyEval_ThreadsInitialized()) {
    PyEval_InitThreads();
  }
  ...
```

åŕźäžŐäżżä¡ŢèřČçŤÍPythonåŕźèšąæĹŰPython C APIçŽĎCäžčçăĄïijŇçąőä£Ïä¡ăéęŰåĚĹåůšçżŔæ■čçąőåI
è£ŹåŔŕäžěçŤÍ `PyGILState_Ensure()` åŠŇ `PyGILState_Release()` æÏěåĄŽåĹřïijŇåęĆäÿŇæĽĂçđ'žïijŽ

```
...
/* Make sure we own the GIL */
PyGILState_STATE state = PyGILState_Ensure();

/* Use functions in the interpreter */
...
/* Restore previous GIL state and return */
PyGILState_Release(state);
...
```

æŕŔæňąèřČçŤÍ `PyGILState_Ensure()` éČ¡èęAçŻÿåžŢçŽĎèřČçŤÍ `PyGILState_Release()`.

èőÍèőž

åIJÍæűĽ'åŔŁåĹřCåŠŇPythonçŽĎéńŸçžğçÍŃåžŔäÿ■ïijŇå¿Ĺåđ'ŽäžŃæČĚäÿĂètůåĄŽæŸŕå¿Ĺåÿÿèğ
åŔŕèČ¡æŸŕåŕźCãĂĄPythonãĂĄCçž£çÍŃãĂĄPythonçž£çÍŃçŽĎæůůåŘĹä¡£çŤÍãĂĆ
åŔłèęAä¡ăçąőä£Ïèğčé ĞŁåŹÍèćńæ■čçąőçŽĎåĹÌåğŃåŇŰïijŇåźűäÿŢæűĽ'åŔŁåĹřèğčéĞŁåŹÍçŽĎCäžčçăĄæĽ'ğ

èęĄæşÍæĎŘçŽĎæŸŕèřČçŤÍ `PyGILState_Ensure()`
åźűäÿ■äijŽçńŇåĹżæŁćå■ăæĹŰäÿ■æŰ■èğčéĞŁåŹÍãĂĆ åęĆæđIJæIJL'åĚűäžŰäžčçăĄæ■čåIJÍæL'ğèąŇïijŇè£Ź
åIJÍåĘĚéČÍïijŇèğčéĞŁåŹÍäijŽæL'ğèąŇåŠÍæIJ§æĂğçŽĎçž£çÍŃåĹĞæ■ćïijŇåŻăæ■d'åęĆæđIJåĚűäžŰçž£çÍŃå
èřČçŤÍèĂĚæIJĂçżĹè£ŸæŸŕåŔŕäžěè£ŘèąŇçŽĎïijÍår¡çőąåŔŕèČ¡èęĄåĚĹç■L'äÿĂäijŽïijL'ãĂĆ

15.9 çŤÍWSIGåŇĚèčĚCäžčçăĄ

éŮőécŸ

ä¡ăæČşèől'ä¡ăåĘŹçŽĎCäžčçăĄä¡IJäÿžäÿĂäÿłCæL'l'åśŢæÍąåİŮæÏěèő£éŰőïijŇæČşéĂŽè£Ğä¡£çŤÍ
SwigåŇĚèčĚçŤ§æĹŘåŹÍ æÏěåőŇæĹŘãĂĆ

èğčåĘşæŰźæąĹ

SwigéĂŽè£Ğèğčæđ ŘCåđ't'æŰĞäžűåźűèĞłåŁÍåĹŻåžžæL'l'åśŢäžčçăĄæÏěæŞ■ä¡IJãĂĆ
èęĄä¡£çŤÍåőČïijŇä¡ăåĚĹèęĄæIJL'äÿĂäÿłCåđ't'æŰĞäžűãĂĆä¿ŃåęĆïijŇæĹŚäžñçđ'žä¿ŃçŽĎåđ't'æŰĞäžűåę

```
/* sample.h */

#include <math.h>
extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

äÿĂæŮęä¡ăæIJL’äžĘè£Źäÿłåđ’ť’æŰĞäžűïijŇäÿŃäÿĂæ■ěářśæŸŕçijŰåĘŹäÿĂäÿłSwig”æŐěåŔč”æŰĞäžűæŇL’çĚğçžęåőŽïijŇè£ŹäžŻæŰĞäžűäžě”.i”åŘŐçijĂåźűäÿŤçśżäijijäÿŃéÍćè£ŹæăůïijŽ

```
// sample.i - Swig interface
%module sample
%{
#include "sample.h"
%}

/* Customizations */
%extend Point {
    /* Constructor for Point objects */
    Point(double x, double y) {
        Point *p = (Point *) malloc(sizeof(Point));
        p->x = x;
        p->y = y;
        return p;
   };
};

/* Map int *remainder as an output argument */
%include typemaps.i
%apply int *OUTPUT { int * remainder };

/* Map the argument pattern (double *a, int n) to arrays */
%typemap(in) (double *a, int n)(Py_buffer view) {
  view.obj = NULL;
  if (PyObject_GetBuffer($input, &view, PyBUF_ANY_CONTIGUOUS |␣
↪PyBUF_FORMAT) == -1) {
    SWIG_fail;
  }
  if (strcmp(view.format,"d") != 0) {
    PyErr_SetString(PyExc_TypeError, "Expected an array of doubles
↪");
    SWIG_fail;
  }
```

```
  $1 = (double *) view.buf;
  $2 = view.len / sizeof(double);
}

%typemap(freearg) (double *a, int n) {
  if (view$argnum.obj) {
    PyBuffer_Release(&view$argnum);
  }
}

/* C declarations to be included in the extension module */

extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

äÿĂæŮęä¡ăåĘŹåě¡äžĘæŐěåŘčæŰĞäżűĭijŇářśåŔŕäžěåIJÍåŚ¡äžd'èąŇåůěåĚůäÿ■èřČçŤÍSwigäžĘĭijŽ

```
bash % swig -python -py3 sample.i
bash %
```

swigçŽĎè¿ŞåĞžářśæŸŕäÿd'äÿłæŰĞäżűĭijŇsample_wrap.cåŠŇsample.pyãĂĆ åŘŐéÍćçŽĎæŰĞäżűåřśæŸŕçŤÍæĹůéIJĂèęĄåŕijåĚěçŽĎãĂĆ èĂŇsample_wrap.cæŰĞäżűæŸŕéIJĂèęĄęćńçijŰèŕŚåĹřåŘ■åŔń `_sample` çŽĎæŤŕæŇĄæĺąåİŮçŽĎCäžčçăĄãĂĆ è£ŹäÿłåŔŕäžěéĂŽè£Ğèů§æŹőéĂŽæĽ'å śŢæĺąåİŮäÿĂæăůçŽĎæŁĂæ ä¿ŇåęĆĭijŇä¡ăåĹŻåžžäžĘäÿĂäÿłåęĆäÿŇæĽ'Äçd'žçŽĎ `setup.py` æŰĞäżűĭijŽ

```
# setup.py
from distutils.core import setup, Extension

setup(name='sample',
      py_modules=['sample.py'],
      ext_modules=[
        Extension('_sample',
                  ['sample_wrap.c'],
                  include_dirs = [],
                  define_macros = [],

                  undef_macros = [],
                  library_dirs = [],
                  libraries = ['sample']
                  )
        ]
```

```
)
```

èęAçijŰèŕŚåŠŇæţŃèŕŢïijŇåIJĺsetup.pyäÿŁæL’ğèąŇpython3ïijŇåęĆäÿŃïijŽ

```
bash % python3 setup.py build_ext --inplace
running build_ext
building '_sample' extension
gcc -fno-strict-aliasing -DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-
 ↪prototypes
-I/usr/local/include/python3.3m -c sample_wrap.c
 -o build/temp.macosx-10.6-x86_64-3.3/sample_wrap.o
sample_wrap.c: In function âĂŸSWIG_InitializeModuleâĂŹ:
sample_wrap.c:3589: warning: statement with no effect
gcc -bundle -undefined dynamic_lookup build/temp.macosx-10.6-x86_64-
 ↪3.3/sample.o
 build/temp.macosx-10.6-x86_64-3.3/sample_wrap.o -o _sample.so -
 ↪lsample
bash %
```

åęĆædIJäÿĂåĹĞæ■čåÿÿçŽĎèŕİïijŇä¡ăäijŽåŔŚçŐřä¡ăåřšåŔŕäžěå¿ĹæŰźä¿£çŽĎä¡£çŤĺçŤ§æĹŘçŽĎCæĽ

```
>>> import sample
>>> sample.gcd(42,8)
2
>>> sample.divide(42,8)
[5, 2]
>>> p1 = sample.Point(2,3)
>>> p2 = sample.Point(4,5)
>>> sample.distance(p1,p2)
2.8284271247461903
>>> p1.x
2.0
>>> p1.y
3.0
>>> import array
>>> a = array.array('d',[1,2,3])
>>> sample.avg(a)
2.0
>>>
```

èőĺèőž

SwigæŸŕPythonnåŐĘåŔšäÿ■ædĎåžžæL’ľåśŢæĺąåİŮçŽĎæIJĂåŔd’èĂĄçŽĎåůěåĚůäźŃäÿĂãĂĆ SwigèČ¡èĞłåŁĺåŇŰå¿Ĺåd’ŽåŇĚèčĚçŤ§æĹŘåŹĺçŽĎåd’ĎçŘĘãĂĆ

æL’ĂæIJL’SwigæŐěåŘčéČ¡äžěçśžäijijäÿŃéİćè£Źæăůç ŽĎäÿžåijĂåd’t’ïijŽ

```
%module sample
%{
```

```
#include "sample.h"
%}
```

è£ŹäÿłäżĕäżĕåŕłæŸŕåčřæŸŐäžĘæĽľåśŢæĺąåİŮçŽĎåŘ■çğřåźűæŇĞåőŽäžĘCåď'ťæŰĞäżűïijŇ
äÿžäžĘèČ¡èőľçijŰèŕŚéĂŽè£Ğå£Ěéążèę Ąå ŇĚåŘńè£ŹäžŽåď'ťæŰĞäżűïijĹä¡■äžŐ %{ åŠŇ %}
çŽĎäżčçăĄïijĽïijŇ åřĘåőČäżňäźŇéŮť'åď'■åĹűçšŸèť'ťåĹřè¿ŞåĞžäżčçăĄäÿ■ïijŇè£Źäź§æŸŕä¡äèęĄæŢ¿ç¡őæ

SwigæŐĕåŔčçŽĎåžŢäÿŇéČĺåĹĘæŸŕäÿĂäÿłCåčřæŸŐåĹŮèąĺïijŇä¡ăéIJĂèęĄåIJĺæĽľåśŢäÿ■åŇĚåŘńåő
è£ŹéĂŽåÿÿäżŐåď'ťæŰĞäżűäÿ■èćńåď'■åĹűãĂĆåIJĺæĹŚäžňçŽĎä¿Ńå■Řäÿ■ïijŇæĹŚäžňäžĔäżĔåČŔäÿŇéİćè£

```
%module sample
%{
#include "sample.h"
%}
...
extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

æIJL'äÿĂçĆźéIJĂèęĄåijžěřČçŽĎæŸŕè£ŹäžŽåčřæŸŐäijŽåŚŁèŕĽ'Swigä¡ăæČşèęĄåIJĺPythonæĺąåİŮäÿ■å
éĂŽåÿÿä¡ăéIJĂèęĄçijŰè¿Śè£ŹäÿłåčřæŸŐåĹŮèąĺæĹŰçŽÿåžŢçŽĎä£őæŢźäÿŇåőČãĂĆ
ä¿ŇåęĆïijŇåęĆædIJä¡ääÿ■æČşæ§ŘäžŽåčřæŸŐèćńåŇĚåŘńè£ŻæİěïijŇä¡ăèęĄårĘåőČäżŐåčřæŸŐåĹŮèąĺäÿ■

ä¡£çŢĺSwigæIJĂåď'■æİĆçŽĎåIJřæŰźæŸŕåőČèČ¡çżŹCäżčçăĄæŔŘä¿Żåď'ğéĞŔçŽĎèĞłåőŽäźL'æŞ■ä¡IJ
è£ŹäÿłäÿżéćŸåď'łåď'ğïijŇè£ŹéĞŇæŮăæşŢåśŢåijĂïijŇä¡ĘæŸŕæĹŚäžňåIJĺæIJńèŁĆè£ŸåL'ľåśŢçď'žäžĘäÿĂäż

çňňäÿĂäÿłèĞłåőŽäźL'æŸŕ `%extend` æŇĞäżď'åĚĄèőÿæŰźæşŢèćńéŹĎåŁäåĹřåůšå■ŸåIJĺçŽĎçżŞædĎä
æĹŚä¿Ńå■Řäÿ■ïijŇè£ŹäÿłèćńçŢĺæİěæůżåŁääÿĂäÿłPointçżŞædĎä¡ŞçŽĎædĎéĂäåŹĺæŰźæşŢãĂĆ
åőČåŔŕäżěèőľä¡äåČŔäÿŇéİćè£ŹæăůäĄ£çŢĺè£ŹäÿłçżŞædĎä¡ŞïijŽ

```
>>> p1 = sample.Point(2,3)
>>>
```

åęĆædIJçŢěè£ĞçŽĎèŕİïijŇPointåŕźèśąåŕśå£ĚéążäžěæŻť'åŁäåď'■æİĆçŽĎæŰźåijŔæİěèćńåĹŻåżžïijŽ

```
>>> # Usage if %extend Point is omitted
>>> p1 = sample.Point()
>>> p1.x = 2.0
>>> p1.y = 3
```

çňňäžŇäÿłèĞłåőŽäźL'æűL'åŔŁåĹŕåŕź `typemaps.i` åžŞçŽĎåijŢåĚěåŠŇ
`%apply` æŇĞäżď'ïijŇ åőČäijŽæŇĞçď'žSwigåŔĆæŢřç■¿åŘ■ `int *remainder`
èęĄèćńå¡ŞåĄŽæŸŕè¿ŞåĞžåĂijãĂĆ è£ŹäÿłåőđéŹĚäÿŁæŸŕäÿĂäÿłæĺąåijŔåŇźéĚ■èğĎåĹŹãĂĆ
åIJĺæŐěäÿŇæİěçŽĎæL'ĂæIJL'åčřæŸŐäÿ■ïijŇäżżä¡ŢæŮűåĂŹåŔłèęĄçćŕäÿŁ `int`

`*remainder` ïijŇäżŰåřśäijŽèćńä¡IJäÿžè¿ŞåĞžãĂĆ è£ŹäÿłèĞłåőŽäźL'æŰźæşŢåŔŕäžěèől' `divide()` åĞ¡æŢřè£ŤåŻđäÿd'äÿłåĂijãĂĆ

```
>>> sample.divide(42,8)
[5, 2]
>>>
```

æIJĂåŘŐäÿĂäÿłæűL'åŔŁåĹř `%typemap` æŇĞäżd'çŽĎèĞłåőŽäźL'åŔŕèČ¡æŸŕè£ŹéĞŇåśŢçd'žçŽĎæIJĂ äÿĂäÿłtypemapåřśæŸŕäÿĂäÿłåIJÍè¿ŞåĚëäÿ■çL'źåőŽåŔĆæŢřæĺąåijŔçŽĎèğĎåĹŹãĂĆ åIJÍæIJňèŁĆäÿ■ïijŇäÿĂäÿłtypemapèćńåőŽäźL'äÿžåŇźéĚ■åŔĆæŢřæĺąåijŔ `(double *a, int n)`. åIJÍtypemapåĘĚéČÍæŸŕäÿĂäÿłCäżčçăĄçL'Ğæőţ̈ijŇåőČåŚèŕL'SwigæĂŐæăůåřĘäÿĂäÿłPythonår æIJňèĽĆäżčçăĄä¡£çŤÍäžĘPythonçŽĎçijŞå■Ÿå■ŔèőőåőžåŇźéĚ■äżżä¡ŢçIJŃäÿŁåŐżçśżäijijåŔŇçš¿åžęæŢřçż ïijĹæŕŤåęĆNumPyæŢřçżĎãĂĄarrayæĺąåİŮåĹŻåžžçŽĎæŢřçżĎç■L'ïijL'ïijŇæŻt'åd'ŽèŕůåŔĆèĂČ15.3åřŔèŁĆ

åIJÍtypemapäżčçăĄåĘĚéČÍïijŇ$1åŠŇ$2è£ŹæăůçŽĎåŔŸéĞŔæŻ£æ■ćäijŽèŐůåŔŰtypemapæĺąåijŔçŽĎC ïijĹæŕŤåęĆ$1æŸăåřĎäÿž `double *a` ïijL'ãĂĆ$inputæŇĞåŘśäÿĂäÿłä¡IJäÿžè¿ŞåĚěçŽĎ `PyObject *` åŔĆæŢřïijŇ èĂŇ `$argnum` åřśäżčèąĺåŔĆæŢřçŽĎäÿłæŢřãĂĆ

çijŰåĘŹåŠŇçŘĘèğčtypemapsæŸŕä¡£çŤÍSwigæIJĂå§žæIJňçŽĎåL'■æŔŘãĂĆ äÿ■äžĚæŸŕèŕť'äžčçăĄæŻt'çědçğŸïijŇèĂŇäÿŤä¡ăéIJĂèęĄçŘĘèğčPython C APIåŠŇSwigåŠŇåőČäžd'äžŞçŽĎæŰźåijŔãĂĆ SwigæŰĞæąčæIJL'æŻt'åd'Žè£ŹæŰźéİćçŽĎçżĘèŁĆïijŇåŔŕäż

äÿ■è£ĞïijŇåęĆædIJä¡ăæIJL'åd'ğéĞŔçŽĎCäżčçăĄéIJĂèęĄèćńæŽt'éIJšäÿžæL'l'åśŢæĺąåİŮãĂĆ SwigæŸŕäÿĂäÿłéİđåÿÿåijžåd'ğçŽĎåůěåĚůãĂĆåĚşéŤőçĆźåIJĺäžŐSwigæŸŕäÿĂäÿłåd'ĎçŘĘCåčřæŸŐçŽĎçij éĂŽè£Ğåijžåd'ğçŽĎæĺąåijŔåŇźéĚ■åŠŇèĞłåőŽäźL'çżĎäžűïijŇåŔŕäżčèől'ä¡ăæŻt'æŤźåčřæŸŐæŇĞåőŽåŠŇçś æŻt'åd'Žä£ąæĄŕèŕůåŐżæ§ěéŸĚ Swigç¡ŚçńŹ ïijŇ è£ŸæIJL' çL'źåőŽäžŐPythonçŽĎçŽÿåĚşæŰĞæąč

15.10 çŤÍCythonåŇĚèčĚCäżčçăĄ

éŮőécŸ

ä¡ăæČşä¡£çŤÍCythonæİěåĹŻåžžäÿĂäÿłPythonæL'l'åśŢæĺąåİŮïijŇçŤÍæİěåŇĚèčĚæ§Řäÿłåůšå■ŸåIJÍçŽĎ

èğčåĘşæŰźæąĹ

ä¡£çŤÍCythonæđĎåžžäÿĂäÿłæL'l'åśŢæĺąåİŮçIJŃäÿŁåŐżå¿ĹæL'ŃåĘŹæL'l'åśŢæIJL'äžŽçśżäijijïijŇ åŽäääÿžä¡äéIJĂèęĄåĹŻåžžå¿Ĺåd'ŽåŇĚèčĚåĞ¡æŢřãĂĆäÿ■è£ĞïijŇèů§åL'■éİćäÿ■åŘŇçŽĎæŸŕïijŇä¡ääÿ■éIJĂ

ä¡IJäÿžåĞĘåd'ĞïijŇåĄĞèő¿æIJňçńääżŃçż■éČĺåĹĘçŽĎçd'žä¿Ńäžčçă ĄåůšçżŕèćńçijŰèŕŚåĹřæ§Řäÿłåŕ `libsample` çŽĎCåĞ¡æŢřåžŞäÿ■äžĘãĂĆ éęŰåĚĹåĹŻåžžäÿĂäÿłåŘ■åŕń `csample.pxd` çŽĎæŰĞäżűïijŇåęĆäÿŃæL'Ăçd'žïijŽ

```
# csample.pxd
#
# Declarations of "external" C functions and structures

cdef extern from "sample.h":
    int gcd(int, int)
```

```
    bint in_mandel(double, double, int)
    int divide(int, int, int *)
    double avg(double *, int) nogil

    ctypedef struct Point:
         double x
         double y

    double distance(Point *, Point *)
```

è£Źäÿ łæŰĞäżűåIJÍCythonäÿ■çŽĎä¡IJçŤĺåřśèů§CçŽĎåď ť æŰĞäżűäÿĂæăůãĂĆ åĹİåğŃåčřæŸŐ `cdef extern from "sample.h"` æŇĞåőŽäžĘæĽĂå■ęçŽĎCåď ť æŰĞäżűãĂĆ æŐěäÿŃæİěçŽĎåčřæŸŐéČ¡æŸŕæİèèĞłäžŐéĆčäÿłåď ť æŰĞäżűãĂĆæŰĞäżűåŘ■æŸŕ `csample.pxd` ïijŇèĂŇäÿ■æŸŕ `sample.pxd` âĂŤâĂŤè£ŹçĆźå¿ĹéĞ■èęĄãĂĆ

äÿŃäÿĂæ■ěïijŇåĹŻåžžäÿĂäÿłåŘ■äÿž `sample.pyx` çŽĎéŮőécŸãĂĆ èŕěæŰĞäżűäijŽåőŽäźĽåŇĚèčĚåŹĺïijŇçŤĺæİěæąěæŐěPythonèğčéĞŁåŹĺåĹř `csample.pxd` äÿ■åčřæŸŐçŽĎCäžčçăĄãĂĆ

```
# sample.pyx

# Import the low-level C declarations
cimport csample

# Import some functionality from Python and the C stdlib
from cpython.pycapsule cimport *

from libc.stdlib cimport malloc, free

# Wrappers
def gcd(unsigned int x, unsigned int y):
    return csample.gcd(x, y)

def in_mandel(x, y, unsigned int n):
    return csample.in_mandel(x, y, n)

def divide(x, y):
    cdef int rem
    quot = csample.divide(x, y, &rem)
    return quot, rem

def avg(double[:] a):
    cdef:
        int sz
        double result

    sz = a.size
    with nogil:
        result = csample.avg(<double *> &a[0], sz)
    return result
```

```
# Destructor for cleaning up Point objects
cdef del_Point(object obj):
    pt = <csample.Point *> PyCapsule_GetPointer(obj,"Point")
    free(<void *> pt)

# Create a Point object and return as a capsule
def Point(double x,double y):
    cdef csample.Point *p
    p = <csample.Point *> malloc(sizeof(csample.Point))
    if p == NULL:
        raise MemoryError("No memory to make a Point")
    p.x = x
    p.y = y
    return PyCapsule_New(<void *>p,"Point",<PyCapsule_Destructor>
 ↪del_Point)

def distance(p1, p2):
    pt1 = <csample.Point *> PyCapsule_GetPointer(p1,"Point")
    pt2 = <csample.Point *> PyCapsule_GetPointer(p2,"Point")
    return csample.distance(pt1,pt2)
```

èŕěæŰĞäžűæŻťåď ŽçŽĎçżĘèŁĆéČĺåĹĘäijŽåIJĺèőĺèőžéČĺåĹĘèŕęçżĘåśŢåijĂãĂĆ æIJĂåŘŐïijŇäÿžäžĘæđĎåžžæĽľåśŢæĺąåİŮïijŇåČŔäÿŇéİćè£ŹæăůåĹŽåžžäÿĂäÿł `setup.py` æŰĞäžűïijŽ

```
from distutils.core import setup
from distutils.extension import Extension
from Cython.Distutils import build_ext

ext_modules = [
    Extension('sample',

              ['sample.pyx'],
              libraries=['sample'],
              library_dirs=['.'])]
setup(
  name = 'Sample extension module',
  cmdclass = {'build_ext': build_ext},
  ext_modules = ext_modules
)
```

èęĄæđĎåžžæĹŚäžňæţŇèŕŢçŽĎçŻőæăĞæĺąåİŮïijŇåČŔäÿŇéİćè£ŹæăůåĄŽïijŽ

```
bash % python3 setup.py build_ext --inplace
running build_ext
cythoning sample.pyx to sample.c
building 'sample' extension
gcc -fno-strict-aliasing -DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-
 ↪prototypes
 -I/usr/local/include/python3.3m -c sample.c
```

```
 -o build/temp.macosx-10.6-x86_64-3.3/sample.o
gcc -bundle -undefined dynamic_lookup build/temp.macosx-10.6-x86_64-
 ↪3.3/sample.o
  -L. -lsample -o sample.so
bash %
```

åęĆædIJäÿĂåĹĞéąžåĹl'çŽĎèŕĬïijŇä¡ååžŤèŕěæIJL'äžĘäÿĂäÿłæL'l'åśŢæĺąåİŮ `sample.so` ïijŇåŔŕåIJĺäÿŃéİćä¿Ńå■Řäÿ■ä¡£çŤĺïijŽ

```
>>> import sample
>>> sample.gcd(42,10)
2
>>> sample.in_mandel(1,1,400)
False
>>> sample.in_mandel(0,0,400)
True
>>> sample.divide(42,10)
(4, 2)
>>> import array
>>> a = array.array('d',[1,2,3])
>>> sample.avg(a)
2.0
>>> p1 = sample.Point(2,3)
>>> p2 = sample.Point(4,5)
>>> p1
<capsule object "Point" at 0x1005d1e70>
>>> p2
<capsule object "Point" at 0x1005d1ea0>
>>> sample.distance(p1,p2)
2.8284271247461903
>>>
```

èőĺèőž

æIJňèŁĆåŇĚåŘńäžĘå¿Ĺåd'ŽåL'■éİćæL'Ăèőšç ŽĎéńŸçžğçL'źæĂğïijŇåŇĚæŃňæŢřçżĎæŞ■ä¡IJãĂĄåŇĚæŕŔäÿĂéČĺåĹĘéČ¡äijŽéĂŘäÿłèćńèőšè£řåĹřïijŇä¡ĘæŸŕæĹŚäžňæIJĂåě¡èČ¡åd'■äźääÿĂäÿŃåL'■éİćåĞäářŔèŁåIJĺéąűåśĆïijŇä¡£çŤĺCythonæŸŕå§žäžŐCäźŃäÿŁãĂĆ.pxdæŰĞäżűäžĚäżĚåŔłåŇĚåŘńCåőŽäźL'ïijĹçśżäijij.hæ.pyxæŰĞäżűåŇĚåŘńäžĘåőđçŐřïijĹçśżäijij.cæŰĞäżűïijL'ãĂĆ`cimport` èŕ■åŘèèćńCythonçŤĺæĭěåŕijåĚě.pxdæŰĞäżűäÿ■çŽĎåőŽäźL'ãĂĆ åőČèů§ä¡£çŤĺæŹőéĂŽçŽĎåŁäè¡¡PythonæĺąåİŮçŽĎåŕijåĚèèŕ■åŘèæŸŕäÿ■åŘŇçŽĎãĂĆ

åř¡çőą.*pxd* æŰĞäżűåŇĚåŘńäžĘåőŽäźL'ïijŇä¡ĘåőČäżňåźűäÿ■æŸŕçŤĺæĭèèĞłåŁĺåĹŻåżžæL'l'åśŢäżčçăĄçåŻăæ■d'ïijŇä¡ăè£ŸæŸŕèęĄåĘŹåŇĚèčĚåĞ¡æŢřãĂĆä¿ŃåęĆïijŇåřśçőŮ `csample.pxd` æŰĞäżűåčřæŸŐäžĘ `int gcd(int, int)` åĞ¡æŢřïijŇ ä¡ääž■çĎűéIJĂèęĄåIJĺ `sample.pyx` äÿ■äÿžåőČåĘŹäÿĂäÿłåŇĚèčĚåĞ¡æŢřãĂĆä¿ŃåęĆïijŽ

```
cimport csample
```

```
def gcd(unsigned int x, unsigned int y):
    return csample.gcd(x,y)
```

åŕźäžŐçőĂå■ŢçŽĎåĞ¡æŢřïijŇä¡ääźűäÿ■éIJĂèęĄåŐżåĄŽåď'łåď'ŽçŽĎæŮűãĂĆ
CythonäijŽçŤ§æĹŘåŇĚèčĔäžčçăĄæİěæ■čçąőçŽĎè¡ñæ■ćåŔĆæŢřåŠŇè£ŤåŻđåĂijãĂĆ
çżŚåőŽåĹřåśđæĂğäÿŁçŽĎCæŢřæ■őçśžåđŇæŸŕåŔŕéĂĽ'çŽĎãĂĆäÿ■è£ĞïijŇåęĆæđIJä¡ääŇĚåŘńäžĘåőČäžñ
ä¿ŇåęĆïijŇåęĆæđIJæIJL'äžžä¡£çŤĺèt'§æŢřæİěèřČçŤĺè£ŹäÿłåĞ¡æŢřïijŇäijŽæŁŻåĞžäÿĂäÿłåijĆåÿÿïijŽ

```
>>> sample.gcd(-10,2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "sample.pyx", line 7, in sample.gcd (sample.c:1284)
    def gcd(unsigned int x,unsigned int y):
OverflowError: can't convert negative value to unsigned int
>>>
```

åęĆæđIJä¡ăæČşåŕźåŇĚèčĔåĞ¡æŢřåĄŽåŘęåď'ŰçŽĎæčÄæ§ěïijŇåŘłéIJĂèęĄä¡£çŤĺåŘęåď'ŰçŽĎåŇĚèčĚ

```
def gcd(unsigned int x, unsigned int y):
    if x <= 0:
        raise ValueError("x must be > 0")
    if y <= 0:
        raise ValueError("y must be > 0")
    return csample.gcd(x,y)
```

åIJĺcsample.pxdæŰĞäžűäÿ■çŽĎ‘‘in_mandel()‘‘ åčřæŸŐæIJL'äÿłå¿ĹæIJL'èűčä¡ĘæŸŕæŕŤè¿ČéŽ¿çŘĘèğ
åIJĺè£ŹäÿłæŰĞäžűäÿ■ïijŇåĞ¡æŢřèćńåčřæŸŐäÿžçĎűåŘŐäÿĂäÿłbintèĂŇäÿ■æŸŕäÿĂäÿłintãĂĆ
åőČäijŽèől'åĞ¡æŢřåĹŻåžžäÿĂäÿłæ■čçąőçŽĎBooleanåĂijèĂŇäÿ■æŸŕçőĂå■ŢçŽĎæŢt'æŢřãĂĆ
åŽăæ■d'ïijŇè£ŤåŽđåĂij0èąĺçd'žFalseèĂŇ1èąĺçd'žTrueãĂĆ

åIJÍCythonåŇĚèčĔåŹĺäÿ■ïijŇä¡ääŔŕäžěéĂĽ'æŃl'åčřæŸŐCæŢřæ■őçśžåđŇïijŇäź§åŔŕäžěä¡£çŤĺæĽĂæIJ
åŕźäžŐ `divide()` çŽĎåŇĚèčĔåŹĺåśŢçd'žäžĘè£ŹæăůäÿĂäÿłä¿Ňå■ŘïijŇåŘŇæŮűè£ŸæIJL'åęĆä¡ŢåŐžåd'Ď

```
def divide(x,y):
    cdef int rem
    quot = csample.divide(x,y,&rem)
    return quot, rem
```

åIJĺè£ŹéĞŇïijŇ`rem`åŘŸéĞŔèćńæŸ¿çd'žçŽĎåčřæŸŐäÿžäÿĂäÿłCæŢt'ådŃåŘŸéĞŔãĂĆ
å¡ŞåőČèćńäijăåĚě `divide()` åĞ¡æŢřçŽĎæŮűåĂŹïijŇ`&rem`
åĹŻåžžäÿĂäÿłèů§CäÿĂæăůçŽĎæŇĞåŘŚåőČçŽĎæŇĞéŠĹãĂĆ `avg()`
åĞ¡æŢřçŽĎäžčçăĄæijŤçd'žäžĘCythonæŻt'éńŸçžğçŽĎçL'źæĂğãĂĆ
éęŰåĚĹ `def avg(double[:] a)` åčřæŸŐäžĘ `avg()`
æŐěåŘŮäÿĂäÿłäÿĂçžt'çŽĎåŘŇçš¿ăžęåĘĚå■ŸèğĘåŻ¿ãĂĆ æIJĂæČŁåěĞçŽĎéČĺåĹĘæŸŕè£ŤåŽđçŽĎçżŞæđ

```
>>> import array
>>> a = array.array('d',[1,2,3])
>>> import numpy
>>> b = numpy.array([1., 2., 3.])
>>> import sample
```

```
>>> sample.avg(a)
2.0
>>> sample.avg(b)
2.0
>>>
```

åIJÍæ■d'åÑĚèčĚåŹÍäÿ■ïijÑa.size0 åŠÑ &a[0] åĹĘåĹńåijŢçŤÍæŢřçżĎåĚČçt'ääÿłæŢřåŠÑåžŢåśĆæÑèŕ■æşŢ <double *> &a[0] æŢŹä¡äæĂŐæăůåřĘæÑĞéŠĹè¡ňæ■ćäÿžäÿ■åŘÑçŽĎçśžåđŃãĂĆ åL'■æŔŘæŸŕCäÿ■çŽĎ avg() æŐěåŔŮäÿĂäÿłæ■čçąőçśžåđŃçŽĎæÑĞéŠĹãĂĆ åŔĆèĂČäÿŃäÿĂèŁĆåĚşäžŐCythonåĘĚå■ŸèğĘåŻ¿çŽĎæŻt'énŸçžğèőšè£řãĂĆ

éŹđ'äžĘåđ'ĎçŘĘéĂŽåÿÿçŽĎæŢřçżĎåđ'ŰïijÑavg() çŽĎè£Źäÿłä¿Ńå■Řè£ŸåśŢçđ'žäžĘåęĆä¡Ţåđ'ĎçŘèŕ■åŔě with nogil: åčřæŸŐäžĘäÿĂäÿłäÿ■éIJĂèęĄGILåřśèČ¡æL'ğèąŇçŽĎäżčçăĄåİŮãĂĆ åIJÍè£ŹäÿłåİŮäÿ■ïijÑäÿ■èČ¡æIJL'äżżä¡ŢçŽĎæŹőéĂŽPythonåŕźèśąâĂŤâĂŤåŔłèČ¡ä¡£çŤÍèćńåčřæŸŐäÿž cdef çŽĎåŕźèśąåŠÑåĞ¡æŢřãĂĆ åŘęåđ'ŰïijÑåđ'ŰéČÍåĞ¡æŢřå£Ëéążç Őřåőđç ŽĎåčřæŸŐåőČäžňèČ¡äÿ■ä¿İè¢åŻäæ■d'ïijÑåIJÍcsample.pxdæŰĞäżűäÿ■ïijÑavg() èćńåčřæŸŐäÿž double avg(double *, int) nogil.

åŕźPointçżŞæđĎä¡ŞçŽĎåđ'ĎçŘĘæŸŕäÿĂäÿłæŃŚæÍŸãĂĆæIJňèŁĆä¡£çŤÍèČűåŻŁåŕźèśąåřĘPointåŕźèśąåèęĄè£ŹæăůåĄŽçŽĎèŕİïijÑåžŢåśĆCythonäżčçăĄçÍ■å¿őæIJL'çĆźåđ'■æİĆãĂĆ éęŰåĚĹïijÑäÿŃéİćçŽĎåŕijåĚèèćńçŤÍæİèåijŢåĚěCåĞ¡æŢřåžŞåŠÑPython C APIäÿ■åőŽäźL'çŽĎåĞ¡æŢřïijŽ

```
from cpython.pycapsule cimport *
from libc.stdlib cimport malloc, free
```

åĞ¡æŢř del_Point() åŠÑ Point() ä¡£çŤÍè£ŹäÿłåŁ§èČ¡æİèåĹŽåžžäÿĂäÿłèČűåŻŁåŕźèśąïijÑåőČäijŽåŃĚèčĚäÿĂäÿł Point * æÑĞéŠĹãĂĆcdef del_Point() åřĘ del_Point() åčřæŸŐäÿžäÿĂäÿłåĞ¡æŢřïijÑ åŘłèČ¡éĂŽè£ĞCythonèő£éŮőïijÑèĂÑäÿ■èČ¡äžŐPythonäÿ■èő£éŮőãĂĆ åŽäæ■d'ïijÑè£ŹäÿłåĞ¡æŢřåŕźåđ'ŰéČÍæŸŕäÿ■åŘŕèğĄçŽĎâĂŤâĂŤåőČèćńçŤÍæİèå¡ŞåĄŽäÿĂäÿłåŽđèřČåĞ¡æŢřåĞ¡æŢřèŘČçŤÍæŕŤåęĆ PyCapsule_New() ãĂĄPyCapsule_GetPointer() çŻt'æŐěæİèèĞłPython C APIåźűäÿŤäžěåŘÑæäůçŽĎæŰźåijŘèćńä¡£çŤÍãĂĆ

distance åĞ¡æŢřäžŐ Point() åĹŽåžžçŽĎèČűåŻŁåŕźèśąäÿ■æŔŘåŘŰæÑĞéŠĹãĂĆ è£ŹéĞÑèęĄæşÍæĎŔçŽĎæŸŕä¡ääÿ■éIJĂèęĄæÑĚå£ĈåijĆåÿÿåd'ĎçŘĘãĂĆ åęĆædIJäÿĂäÿłéŤŹèŕŕçŽĎåŕźèśąèćńäijàè£ŻæİěïijÑPyCapsule_GetPointer() äijŽæŁŽåĞžäÿĂäÿłåijĆåÿÿïijÑ ä¡ĘæŸŕCythonåůšçżŔç§ěéĄŞæĂŐäźĹæ§ěæL'¿åĹřåőČïijÑåźűåřĘåőČäžŐ distance() äijàéĂŠåĞžåŐžãĂĆ

åd'ĎçŘĘPointçżŞæđĎä¡ŞäÿĂäÿłçijžçĆźæŸŕåőČçŽĎåőđçŐřæŸŕäÿ■åŔŕèğĄçŽĎãĂĆ ä¡ääÿ■èČ¡èő£éŮőäżżä¡ŢåśđæĂğæİěæ§ěçIJŃåőČçŽĎåĘĚéČĺãĂĆ è£ŹéĞÑæIJL'åŘęåd'ŰäÿĂçğ■æŰźæşŢåŐżåŃĚèčĚåőČïijÑåřśæŸŕåőŽäźL'äÿĂäÿłæL'l'åśŢçśžåđŃïijÑåęĆäÿŃ

```
# sample.pyx

cimport csample
from libc.stdlib cimport malloc, free
...

cdef class Point:
    cdef csample.Point *_c_point
```

```
    def __cinit__(self, double x, double y):
        self._c_point = <csample.Point *> malloc(sizeof(csample.
↪Point))
        self._c_point.x = x
        self._c_point.y = y

    def __dealloc__(self):
        free(self._c_point)

    property x:
        def __get__(self):
            return self._c_point.x
        def __set__(self, value):
            self._c_point.x = value

    property y:
        def __get__(self):
            return self._c_point.y
        def __set__(self, value):
            self._c_point.y = value

def distance(Point p1, Point p2):
    return csample.distance(p1._c_point, p2._c_point)
```

åIJÍè£ŹéĞŇïijŇcdifçśż Point åřĘPointåčřæŸŐäÿžäÿĂäÿłæĽľåśŢçśżåđŃãĂĆ çśżåśđæĂğ cdef csample.Point *_c_point åčřæŸŐäžĘäÿĂäÿłåőđä¿ŃåŔŸéĞŔïijŇ æŃěæIJĽäÿĂäÿłæŇĞåŘŠåžŢåśĆPointçżŞæđĎä¡ŞçŽĎæŇĞéŠĹãĂĆ __cinit__() åŠŇ __dealloc__() æŰźæşŢéĂŽè£Ğ malloc() åŠŇ free() åĹŻåžžåźűéŤĂæŕĄåžŢåśĆCçżŞæđĎä¡ŞãĂĆ xåŠŇyåśđæĂğçŽĎåčřæŸŐèőľä¡ăèŐůåŔŰåŠŇèő¿ç¡őåžŢåśĆçżŞæđĎä¡ŞçŽĎåśđæĂğåĂijãĂĆ distance() çŽĎåŇĚèčĚåŹÍè£ŸåŔŕäżěèćńä£őæŤźïijŇä¡£å¿ŮåőČèČ¡æŐěåŔŮ Point æĽľåśŢçśżåđŃåőđä¿Ńä¡IJäÿžåŔĆæŢřïijŇ èĂŇäijăéĂŠåžŢåśĆæŇĞéŠĹçżŹĆåĞ¡æŢřãĂĆ

åĄŽäžĘè£ŹäÿłæŤźåŔŸåŘŐïijŇä¡ăäijŽåŔŚçŐřæŞ■ä¡IJPointåŕźèśąåřśæŸ¿å¿ŮæŻť'åŁäèĞłçĎűäžĘïijŽ

```
>>> import sample
>>> p1 = sample.Point(2,3)
>>> p2 = sample.Point(4,5)
>>> p1
<sample.Point object at 0x100447288>
>>> p2
<sample.Point object at 0x1004472a0>
>>> p1.x
2.0
>>> p1.y
3.0
>>> sample.distance(p1,p2)
2.8284271247461903
>>>
```

æIJňèŁĆåůšçżŔæijŤçď'žäžĘå¿Ĺåď'ŽCythonçŽĎæäÿå£ČçĽźæĂğïijŇä¡ăåŔŕäżěäżěæ■d'äÿžå§žåĞĘæİěæ

äÿ■è£ĞïijÑä¡ăæIJĂåě¡åĚĹåŐžéŸĚèŕżäÿŃåőŸæŰźæŰĞæąčæİěäžĘèğčæŻť'åď'Žä£ąæĄŕãĂĆ

æŐěäÿŃæİěåĞăèŁĆè£ŸäijŽçżğçż■æijŤçď'žäÿĂäžŻCythonçŽĎåĚűäžŰçL'źæĂğãĂĆ

15.11 çŤÍCythonåĘŹéńŸæĂğèČ¡çŽĎæŢřçżĎæŞ■ä¡IJ

éŮőécŸ

ä¡ăèęĄåĘŹéńŸæĂğèČ¡çŽĎæŞ■ä¡IJæİěèĞłNumPyäźŃçśżçŽĎæŢřçżĎèőąçőŮåĞ¡æŢřãĂĆ ä¡ăåůšçżŔç§ěéĄŞäžĘCythonè£ŹæăůçŽĎåůěåĚůäijŽèől'åőČåŔŸå¿ŮçőĂå■ŢïijÑä¡ĘæŸŕåźűäÿ■çąőåőŽèřěæĂ

èğčåĘşæŰźæąĹ

ä¡IJäÿžäÿĂäÿłä¿Ńå■ŘïijÑäÿŃéİćçŽĎäžčçăĄæijŤçď'žäžĘäÿĂäÿłCythonåĞ¡æŢřïijÑçŤÍæİěä£őæŢt'äÿĂä

```
# sample.pyx (Cython)

cimport cython

@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip(double[:] a, double min, double max, double[:] out):
    '''
    Clip the values in a to be between min and max. Result in out
    '''
    if min > max:
        raise ValueError("min must be <= max")
    if a.shape[0] != out.shape[0]:
        raise ValueError("input and output arrays must be the same␣
↪size")
    for i in range(a.shape[0]):
        if a[i] < min:
            out[i] = min
        elif a[i] > max:
            out[i] = max
        else:
            out[i] = a[i]
```

èęĄçijŰèŕŚåŠŇædĎåžžè£ŹäÿłæL'l'åśŢïijÑä¡ăéIJĂèęĄäÿĂäÿłåČŔäÿŃéİćè£ŹæăůçŽĎ `setup.py` æŰĞäžű ïijĹä¡£çŤÍ `python3 setup.py build_ext --inplace` æİěædĎåžžåőČïijL'ïijŽ

```
from distutils.core import setup
from distutils.extension import Extension
from Cython.Distutils import build_ext

ext_modules = [
    Extension('sample',
```

```
                ['sample.pyx'])
]

setup(
  name = 'Sample app',
  cmdclass = {'build_ext': build_ext},
  ext_modules = ext_modules
)
```

ä¡ääijŽåŔŚçŐřçżŞæđIJåĞ¡æŢřçąőåőđåŕźæŢřçżĎè£ŻèąŇçŽĎä£őæ■čïijŇåźűäÿŤåŔŕäżěéĂĆçŤĺäžŐåď’Žç

```
>>> # array module example
>>> import sample
>>> import array
>>> a = array.array('d',[1,-3,4,7,2,0])
>>> a

array('d', [1.0, -3.0, 4.0, 7.0, 2.0, 0.0])
>>> sample.clip(a,1,4,a)
>>> a
array('d', [1.0, 1.0, 4.0, 4.0, 2.0, 1.0])

>>> # numpy example
>>> import numpy
>>> b = numpy.random.uniform(-10,10,size=1000000)
>>> b
array([-9.55546017,  7.45599334,  0.69248932, ...,  0.69583148,
       -3.86290931,  2.37266888])
>>> c = numpy.zeros_like(b)
>>> c
array([ 0.,  0.,  0., ...,  0.,  0.,  0.])
>>> sample.clip(b,-5,5,c)
>>> c
array([-5.        ,  5.        ,  0.69248932, ...,  0.69583148,
       -3.86290931,  2.37266888])
>>> min(c)
-5.0
>>> max(c)
5.0
>>>
```

ä¡äè£ŸäijŽåŔŚçŐřè£ŘèąŇçŤ§æĹŘçżŞæđIJéİđåÿÿçŽĎå£ñãĂĆ äÿŃéİćæĹŚäżňåřĘæIJňä¿ŃåŠŇnumpyäÿ■çŽĎåůšå■ŸåIJĺçŽĎ `clip()` åĞ¡æŢřåĄŽäÿĂäÿłæĂğèČ¡åŕźæŕŤïijŽ

```
>>> timeit('numpy.clip(b,-5,5,c)','from __main__ import b,c,numpy',
↪number=1000)
8.093049556000551
>>> timeit('sample.clip(b,-5,5,c)','from __main__ import b,c,sample
↪',
```

```
...           number=1000)
3.760528204000366
>>>
```

æ■čåęĆä¡ăçIJŃåĹřçŽĎïijŇåőČèęĄå£ńå¿Ĺåď˙ŽâĂŤâĂŤè£ŹæŸŕäÿĂäÿłå¿ĹæIJL˙èűčçŽĎçżŞæđIJïijŇåŻă

èőÍèőž

æIJňèŁĆåĹl˙çŤĺäžĘCythonçśžåđŃçŽĎåĘĚå■ŸèğĘåŻ¿ïijŇæđĄåď˙ğçŽĎçőĂåŇŰäžĘæŢřçżĎçŽĎæŞ■ä¡I
`cpdef clip()` åčřæŸŐäžĘ `clip()` åŘŇæŮűäÿžCçžğåĹńåĞ¡æŢřäžěåŔŁPythonçžğåĹńåĞ¡æŢřãĂĆ
åIJÍCythonäÿ■ïijŇè£ŹäÿłæŸŕå¿ĹéĞ■èęĄçŽĎïijŇåŻääÿžåőČèąĺçď˙žæ■ď˙åĞ¡æŢřèřČçŤĺèęĄæŕŤåĚűäžŰCytho
ïijĹæŕŤåęĆä¡ăæČşåIJĺåŔęåď˙ŰäÿĂäÿłäÿ■åŘŇçŽĎCythonåĞ¡æŢřäÿ■èřČçŤĺclip()ïijL˙ãĂĆ

çśžåđŃåŔĆæŢř `double[:] a` åŠŇ `double[:] out` åčřæŸŐè£ŹäžŻåŔĆæŢřäÿžäÿĂçżť˙çŽĎåŔŇçš¿åžęæŢřçżĎãĂĆ
ä¡IJäÿžè¿ŞåĚëïijŇåőČäžňäijŽèő£éŰőäżžä¡Ţåőđç0řäžĘåĘĚå■ŸèğĘåŻ¿æŐěåŔčçŽĎæŢřçżĎåŕźèśąïijŇè£ŹäÿŁ
3118æIJL˙èŕęçżĘåőŽäźL˙ãĂĆ åŇĚæŃňäžĘNumPyäÿ■çŽĎæŢřçżĎåŠŇåĘĚç¡őçŽĎarrayåžŞãĂĆ

å¡Şä¡ăçijŰåEŹçŤ§æĹŘçżŞæđIJäÿžæŢřçżĎçŽĎäżčçăĄæŮűïijŇä¡ăåžŤèŕěéĄţå¿łäÿŁéİćçď˙žä¿ŃéĆčæăůè
åőČäijŽåřĘåĹŻåżžè¿ŞåĞžæŢřçżĎçŽĎèt˙čäżžçżŹèřČçŤĺèĂĚïijŇäÿ■éIJĂèęĄç§ěéĄŞä¡ăæŞ■ä¡IJçŽĎæŢřçżĎç
ïijĹåőČäżĚäżĚåĄĞè0¿æŢřçżĎåůšçżŔåĞĘåď˙Ğåě¡äžĘïijŇåŔłéIJĂèęĄåĄŽäÿĂäžŻåřŘçŽĎæčĂæ§ěæŕŤåęĆçą
åIJĺåČŔNumPyäźŃçśžçŽĎåžŞäÿ■ïijŇä¡£çŤÍ `numpy.zeros()` æĹŰ `numpy.zeros_like()` åĹŻåżžè¿ŞåĞžæŢřçżĎçŻÿåŕźèĂŇèÍĂæŕŤè¿Čåőźæ˙ŸŞãĂĆåŔęåď˙ŰïijŇèęĄåĹŻåżžæIJłåĹİ
ä¡ääŔŕäžěä¡£çŤÍ `numpy.empty()` æĹŰ `numpy.empty_like()` .
åęĆæđIJä¡ăæČşèęĘçŽŰæŢřçżĎåĘĚåőźä¡IJäÿžçżŞæđIJçŽĎèŕÍéĂĽ˙æŃľ˙è£Źäÿđ˙äÿłäijŽæŕŤè¿Čå£ńçĆźãĂĆ

ä¡ää¡ăçŽĎåĞ¡æŢřåőđçŐřäÿ■ïijŇä¡ääŔłéIJĂèęĄçőĂå■ŢçŽĎéĂŽè£ĞäÿŃæăĞè£ŘçőŮåŠŇæŢřçżĎæ§ěæ
CythonäijŽèt˙§èt˙čäÿžä¡ăçŤ§æĹŘéńŸæŢĹçŽĎäżčçăĄãĂĆ

`clip()` åőŽäźL˙äźŃåL˙■çŽĎäÿđ˙äÿłèčĚéěřåŹĺåŔŕäžěäijŸåŃŰäÿŃæĂğèČ¡ãĂĆ
`@cython.boundscheck(False)` çIJĄåŐżäžĘæL˙ĂæIJL˙çŽĎæŢřçżĎèűĹçŢŇçŽĎæŮűåĂŹåŔŕäžěä¡£çŤĺåőČãĂĆ
`@cython.wraparound(False)` æűĹéŹđ˙äžĘçŻÿåŕźæŢřçżĎår¿éČĺçŽĎèt˙§æŢřäÿŃæĂğçŽĎåđ˙ĎçŘĘïijĺ
åijŢåĚèè£Źäÿđ˙äÿłèčĚéěřåŹĺåŔŕäžěæđĄåđ˙ğçŽĎæŔŘå■ĞæĂğèČ¡ïijĹæţŃèŕŢè£Źäÿłä¿Ńå■ŘçŽĎæŮűåĂŹåđ

äžżä¡ŢæŮűåĂŹåđ˙ĎçŘĘæŢřçżĎæŮűïijŇçăŤçl˙ûåźűæŤźåŮĎåžŢåśĆçőŮæşŢåŘŇæäůåŔŕäžěæđĄåđ˙ğçŽ
ä¿ŃåęĆïijŇèĂČèŹŠåŕź `clip()` åĞ¡æŢřçŽĎåęĆäÿŃä£őæ■čïijŇä¡£çŤĺæĮąäžűèąĺè¿¿åijŔïijŽ

```
@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip(double[:] a, double min, double max, double[:] out):
    if min > max:
        raise ValueError("min must be <= max")
    if a.shape[0] != out.shape[0]:
        raise ValueError("input and output arrays must be the same␣
↪size")
    for i in range(a.shape[0]):
        out[i] = (a[i] if a[i] < max else max) if a[i] > min else␣
↪min
```

åőđéŹĚæţŃèŕŢçżŞæđIJæŸŕïijÑè£Źäÿłçľ́ĹæIJňçŽĎäžčçăĄè£ŘèąŇéĂ§åžęèęĄå£ń50%äžëäÿŁïijĹ2.44çğ
`timeit()` æţŃèŕŢçŽĎ3.76çğŠïijĽ̃ãĂĆ

åĹŕè£ŹéĞŇäÿžæ■ćïijÑä¡ååŔŕèČ¡æČşç§ěéĄŞè£Źçğ■äžčçăĄæĂŐäźĹèČ¡èů§æĽ̃ŃåĘŹCèŕ■èĺĂPKåŚćïij§
ä¿ŃåęĆïijÑä¡ååŔŕèČ¡åĘŹäžĘåęĆäÿŃçŽĎCåĞ¡æŢŕåźűä¡£çŤĺåĽ̃■éİćåĞäèŁĆçŽĎæŁĂæIJŕæİěæĽ̃ŃåĘŹæĽ̃ľ

```
void clip(double *a, int n, double min, double max, double *out) {
  double x;
  for (; n >= 0; n--, a++, out++) {
    x = *a;

    *out = x > max ? max : (x < min ? min : x);
  }
}
```

æĹŚäžňæšąæIJĽ̃åśŢçd'žè£ŹäÿłçŽĎæĽ̃ľ̃åśŢäžčçăĄïijÑä¡ĘæŸŕèŕŢélŇäźŃåŘŐïijÑæĹŚäžňåŘŚçŐřäÿĂäj
æIJĂåžŢäÿŃçŽĎäÿĂèąŇæŕŤä¡ăæČşèśąçŽĎè£ŘèąŇçŽĎå£ńå¿Ĺåd'ŽãĂĆ

ä¡ååŔŕäžěåŕźåőđä¿Ńäžčçă Ąæđ Ďåžžåd'ŽäÿłæĽ̃ľ̃åśŢãĂĆ åŕźäžŐæ§ŘäžŻæŢŕçżĎæŞ■ä¡IJïijÑæIJĂåě¡èęĄ
èęĄè£ŹæăůåĄŽçŽĎèŕÍïijÑéIJĂèęĄä£őæŤźäžčçăĄïijÑä¡£çŤÍ `with nogil:` èŕ■åŘěïijŽ

```
@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip(double[:] a, double min, double max, double[:] out):
    if min > max:
        raise ValueError("min must be <= max")
    if a.shape[0] != out.shape[0]:
        raise ValueError("input and output arrays must be the same␣
↪size")
    with nogil:
        for i in range(a.shape[0]):
            out[i] = (a[i] if a[i] < max else max) if a[i] > min␣
↪else min
```

åęĆæđIJä¡ăæČşåĘŹäÿĂäÿłæŞ■ä¡IJäžÑçżť'æŢŕçżĎçŽĎçĽ̃ĹæIJňïijÑäÿÑéİćæŸŕåŔŕäžěåŔĆèĂČäÿÑïijŽ

```
@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip2d(double[:,:] a, double min, double max, double[:,:]␣
↪out):
    if min > max:
        raise ValueError("min must be <= max")
    for n in range(a.ndim):
        if a.shape[n] != out.shape[n]:
            raise TypeError("a and out have different shapes")
    for i in range(a.shape[0]):
        for j in range(a.shape[1]):
            if a[i,j] < min:
                out[i,j] = min
            elif a[i,j] > max:
                out[i,j] = max
            else:
```

```
                out[i,j] = a[i,j]
```

åÿŇæIJŻèřżèĂĚäÿ■èęĄå£ŸäžĘæIJňèŁĆæĽĂæIJĽäżčçăĄéČ¡äÿ■äijŽçżŚåőŽåĹřæ§ŘäÿłçĽźåőŽæŢřçżĎ
è£ŹæăůäżčçăĄåřśæŻťæIJĽçĄţæťżæĂğãĂĆ äÿ■è£ĞïijŇèęĄæşĺæĎŘçŽĎæŸŕåęĆæđIJåđĎçŘĘæŢřçżĎèęĄæ
è£ŹäžŻåĘĚåőźåůšçżŔèűĚåĞžæIJňèŁĆèŇČåŻťïijŇæŻť åđŽä£ąæĄŕèřůåŘĆèĂČ PEP
3118 ïijŇ åŘŇæŮű CythonæŰĞæąčäÿ■åĚşäžŐâĂIJçśżåđŃåĘĚå■ŸèğĘåŻ¿âĂİ
çŕĞäź§åĂijå¿ŮäÿĂèŕżãĂĆ

15.12 åřĘåĞ¡æŢřæŇĞéŠĹè¡ňæ■ćäÿžåŘŕèřČçŤĺåŕźèśą

éŮőécŸ

ä¡ăåůšçżŔèŐůå¿ŮäžĘäÿĂäÿłèćńçijŰèřŚåĞ¡æŢřçŽĎåĘĚå■ŸåIJřåİĂïijŇæČşåřĘåőČè¡ňæ■ćæĹŘäÿĂäÿłP
è£ŹæăůçŽĎèŕİä¡ăåřśåŘŕäžěåřĘåőČä¡IJäÿžäÿĂäÿłæĽľåśŢåĞ¡æŢřä¡£çŤĺäžĘãĂĆ

èğčåĘşæŰźæąĹ

ctypes æĺąåİŮåŘŕèćńçŤĺæİěåĹŻåžžåŇĚèčĚäżżæĎŘåĘĚå■ŸåIJřåİĂçŽĎPythonåŘŕèřČçŤĺåŕźèśąãĂĆ
äÿŇéİćçŽĎä¿Ňå■ŘæijŤçđ žäžĘæĂŐæăůèŐůåŘŰCåĞ¡æŢřçŽĎåŐ§ågŇãĂĄåžŢåśĆåIJřåİĂïijŇäžěåŘŁåęĆä¡

```
>>> import ctypes
>>> lib = ctypes.cdll.LoadLibrary(None)
>>> # Get the address of sin() from the C math library
>>> addr = ctypes.cast(lib.sin, ctypes.c_void_p).value
>>> addr
140735505915760

>>> # Turn the address into a callable function
>>> functype = ctypes.CFUNCTYPE(ctypes.c_double, ctypes.c_double)
>>> func = functype(addr)
>>> func
<CFunctionType object at 0x1006816d0>

>>> # Call the resulting function
>>> func(2)
0.9092974268256817
>>> func(0)
0.0
>>>
```

èőÍèőž

èęĄæđĎåžžäÿĂäÿłåŘŕèřČçŤĺåŕźèśąïijŇä¡ăéęŰåĚĹéIJĂèęĄåĹŻåžžäÿĂäÿł
CFUNCTYPE åőđä¿ŇãĂĆ CFUNCTYPE() çŽĎçňňäÿĂäÿłåŔĆæŢřæŸŕè£ŤåŻđçśżåđŃãĂĆ
æŐëäÿŇæİěçŽĎåŔĆæŢřæŸŕåŔĆæŢřçśżåđŃãĂĆäÿĂæŮęä¡ăåőŽäźĽäžĘåĞ¡æŢřçśżåđŃïijŇä¡ăåřśèČ¡åřĘåőČ

çŤ§æĹŘçŽĎåŕźèśąèćńå¡ŞåĄŽæŹőéĂŽçŽĎåŔŕéĂŽè£Ğ ctypes
èő£éŮőçŽĎåĞ¡æŢřæİěä¡£çŤĺãĂĆ

æIJňèŁĆçIJŃäÿŁåŐżåŔŕèČ¡æIJL'çĆźçěđçğŸïijŇåĄŘåžŢåśĆäÿĂçĆźãĂĆ
ä¡ĘæŸŕïijŇä¡ĘæŸŕåőČèćńáź£æşŻä¡£çŤĺäžŐåŘĎçğ■énŸçžğäžčçäĄçŤ§æĹŘæŁĂæIJŕæŕŤåęĆå■şæŮűçijŰèŕŚ

ä¿ŃåęĆïijŇäÿŃéİćæŸŕäÿĂäÿłä¡£çŤĺ `llvmpy` æĽľåśŢçŽĎçőĂå■Ţä¿Ńå■ŘïijŇçŤĺæİěæđĎåżžäÿĂäÿłåřI
åźűåřĘåĚűè¡ñæ■ćäÿžäÿĂäÿłPythonåŔŕèřČçŤĺåŕźèśąãĂĆ

```
>>> from llvm.core import Module, Function, Type, Builder
>>> mod = Module.new('example')
>>> f = Function.new(mod,Type.function(Type.double(), \
                     [Type.double(), Type.double()], False), 'foo')
>>> block = f.append_basic_block('entry')
>>> builder = Builder.new(block)
>>> x2 = builder.fmul(f.args[0],f.args[0])
>>> y2 = builder.fmul(f.args[1],f.args[1])
>>> r = builder.fadd(x2,y2)
>>> builder.ret(r)
<llvm.core.Instruction object at 0x10078e990>
>>> from llvm.ee import ExecutionEngine
>>> engine = ExecutionEngine.new(mod)
>>> ptr = engine.get_pointer_to_function(f)
>>> ptr
4325863440
>>> foo = ctypes.CFUNCTYPE(ctypes.c_double, ctypes.c_double, ctypes.
↪c_double)(ptr)

>>> # Call the resulting function
>>> foo(2,3)
13.0
>>> foo(4,5)
41.0
>>> foo(1,2)
5.0
>>>
```

åźűäÿ■æŸŕèŕťåIJĺè£ŹäÿłåśĆéİççŁŕäžĘäżżä¡ŢéŤŹèŕŕåřśäijŽåŕijèĞťPythonèğčéĞŁåŹĺæŇĆæŐĽãĂĆ
èęĄèőřå¿ŮçŽĎæŸŕä¡ăæŸŕåIJĺçŻťæÓěèů§æIJžåŹĺçžğåĹńçŽĎåĘĚå■ŸåIJřåİĂåŠŇæIJňåIJřæIJžåŹĺçăĄæĽŞä

15.13 äijăéĂŠNULLçżŞář¿çŽĎå■ŮçňęäÿšçżŹCåĞ¡æŢřážŞ

éŮőécŸ

ä¡ăèęĄåĘŹäÿĂäÿłæĽľåśŢæĺąåİŮïijŇéIJĂèęĄäijăéĂŠäÿĂäÿłNULLçżŞář¿çŽĎå■ŮçňęäÿšçżŹCåĞ¡æŢřå
äÿ■è£ĞïijŇä¡ăäÿ■æŸŕå¿ĹçąőåőŽæĂŐæăůä¡£çŤĺPythonçŽĎUnicodeå■ŮçňęäÿšåŐżåőđçŐřåőĈãĂĆ

èğčåĘşæŰźæąĹ

èőÿåď ŽCåĞ¡æŢřåžŞåŇĚåŘńäÿĂäžŻæŞ■ä¡IJNULLçżŞář¿çŽĎå■ŮçňęäÿšïijŇèćńåčřæŸŐçśżåđŃäÿž `char *`. èĂČèŹŚåęĆäÿŃçŽĎCåĞ¡æŢřïijŇæĹŚäžňçŤĺæİěåĄŽæijŤçď žåŠŇætŃèŕŢçŤĺçŽĎïijŽ

```
void print_chars(char *s) {
    while (*s) {
        printf("%2x ", (unsigned char) *s);

        s++;
    }
    printf("\n");
}
```

æ■ď åĞ¡æŢřäijŽæĽŞå■řèćńäijàè£Żæİěå■ŮçňęäÿšçŽĎæŕŘäÿłå■ŮçňęçŽĎå■ĄåĚ■è£ŻåĹűèąÍçď žïijŇè£Ź

```
print_chars("Hello");   // Outputs: 48 65 6c 6c 6f
```

åŕźäžŐåIJÍPythonäÿ■èřČçŤĺè£ŹæăůçŽĎCåĞ¡æŢřijŇä¡ăæIJĽåĞăçğ■éĂĽæŃľãĂĆ éęŰåĚĹïijŇä¡ăåŘŕäžěéĂŽè£ĞèřČçŤĺ `PyArg_ParseTuple()` åźűæŇĞåőŽâĂİyâĂIJè¡ňæ■ćçăĄæİěéŹŘåĹűåőČåŘłèČ¡æŞ■ä¡IJå■ŮèŁĆïijŇåęĆäÿŃïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s;

  if (!PyArg_ParseTuple(args, "y", &s)) {
    return NULL;
  }
  print_chars(s);
  Py_RETURN_NONE;
}
```

çżŞædIJåĞ¡æŢřçŽĎä¡£çŤĺæŰźæşŢåęĆäÿŃãĂĆäžŤçżĘèğĆår§åţŇåĚěäžĘNULLå■ŮèŁĆçŽĎå■Ůçňęäÿš

```
>>> print_chars(b'Hello World')
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>> print_chars(b'Hello\x00World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be bytes without null bytes, not bytes
>>> print_chars('Hello World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' does not support the buffer interface
>>>
```

åęĆædIJä¡ăæČşäijăéĂŠUnicodeå■ŮçňęäÿšïijŇåIJÍ `PyArg_ParseTuple()` äÿ■ä¡£çŤĺâĂİsâĂIJæăijåijŔçăĄïijŇåęĆäÿŃïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s;
```

```
  if (!PyArg_ParseTuple(args, "s", &s)) {
    return NULL;
  }
  print_chars(s);
  Py_RETURN_NONE;
}
```

å¡Šèćńä¡£çŤÍçŽĎæŮűůåĂŹïijŇåőČäijŽèĞłåŁÍåřĘæĽĂæIJĽå■Ůçňęäÿšè¡ňæ■ćäÿžäżěNULLçżŞář¿çŽĎU
8çijŰçăĄãĂĆä¿ŃåęĆïijŽ

```
>>> print_chars('Hello World')
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>> print_chars('Spicy Jalape\u00f1o')  # Note: UTF-8 encoding
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> print_chars('Hello\x00World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be str without null characters, not str
>>> print_chars(b'Hello World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be str, not bytes
>>>
```

åęĆædIJåŽääÿžæ§ŘäžŽåŐ§åŽäïijŇä¡ăèęAçŻťæŐěä¡£çŤÍ
`PyObject *` èĂŇäÿ■èČ¡ä¡£çŤÍ `PyArg_ParseTuple()` ïijŇ
äÿŃéÍćçŽĎä¿Ńå■ŘåŘŚä¡åáśŢçd'žäžĘæĂŐæăůäżŐå■ŮèŁĆåŠŇå■Ůçňęäÿšářźèśąäÿ■æčÄæ§ěåŠŇæŔŘåŔŰäÿ
`char *` åijŢçŤÍïijŽ

```
/* Some Python Object (obtained somehow) */
PyObject *obj;

/* Conversion from bytes */
{
   char *s;
   s = PyBytes_AsString(o);
   if (!s) {
      return NULL;    /* TypeError already raised */
   }
   print_chars(s);
}

/* Conversion to UTF-8 bytes from a string */
{
   PyObject *bytes;
   char *s;
   if (!PyUnicode_Check(obj)) {
       PyErr_SetString(PyExc_TypeError, "Expected string");
       return NULL;
```

```
    }
    bytes = PyUnicode_AsUTF8String(obj);
    s = PyBytes_AsString(bytes);
    print_chars(s);
    Py_DECREF(bytes);
}
```

åL’■éÍćäÿď’çğ■è¡ňæ■ćéČ¡åŔŕäžěçąőä£ÍæŸŕfNULLçżŞåř¿çŽĎæŢřæ■őïijŇ
ä¡ĘæŸŕåőČäžňåźűäÿ■æčĂæ§ěå■Ůçňęäÿšäÿ■éŮt’æŸŕåŘęåţŇåĚěäžĘNULLå■ŮèŁĆãĂĆ
åŻăæ■d’ïijŇåęĆæđIJè£Źäÿłå¿ĹéĞ■èęĄçŽĎèŕÍïijŇéĆčä¡ăéIJĂèęĄèĞłåůśåŐžåĄŽæčĂæ§ěäžĘãĂĆ

èőÍèőž

åęĆæđIJåŔŕèČ¡çŽĎèŕÍïijŇä¡ăåžŤèŕěéĄ£åĚ■åŐżåĘŹäÿĂäžŻä¿ÌèţŰäžŐNULLçżŞåř¿çŽĎå■ŮçňęäÿšïijŇä
æIJĂåě¡çżŞåŘĹä¡£çŤÍäÿĂäÿłæŇĞéŠĹåŠŇéŢ£åžęåĂijæÌěåď’ĎçŘĘå■ŮçňęäÿšãĂĆ
äÿ■è£ĞïijŇæIJL’æŮűåĂŹä¡ää£ĚéążåŐżåď’ĎçŘĘCèŕ■èÍĂéĄŮçŢŹäżčçăĄæŮűåřśæšąå¿ŮéĂL’æŃľäžĘãĂĆ

åř¡çőąå¿Ĺåőźæ ŸŞä¡£çŤÍïijŇä¡ĘæŸŕå¿Ĺåőźæ ŸŞå£¡ègĘçŽĎäÿĂäÿłéŮőécŸæŸŕåIJÍ
`PyArg_ParseTuple()` äÿ■ä¡£çŤÍâĂIJsâĂÌæăijåijŔåŇŰçăĄäijŽæIJL’åĘĚå■Ÿæ■§èĂŮãĂĆ
ä¡Ęä¡ăéIJĂèęĄä¡£çŤÍè£Źçğ■è¡ňæ■ćçŽĎæŮűåĂŹïijŇäÿĂäÿłUTF-
8å■ŮçňęäÿšèćńåĹŽåžžåźűæřÿäźĚéŹĎåŁăåIJÍåŐ§ågŇå■ŮçňęäÿšåŕźèśąäÿŁéÍćãĂĆ
åęĆæđIJåŐ§ågŇå■Ůçňęäÿšå ŇĚåŘńéÌđASCIIå■Ůçňę ŽĎèŕÍïijŇåŕšäijŽåŕijèĞť å■Ůçňęäÿšç ŽĎåřžåŕÿåćđåĹřäÿ

åęĆæđIJäj¡ăåIJÍäźŐè£ŹäÿłåĘĚå■ŸçŽĎæ■§èĂŮïijŇä¡ăæIJĂåě¡éĞ■åĘŹä¡ăçŽĎCæL’ľåśŢäžčçăĄïijŇèőľå
`PyUnicode_AsUTF8String()` åĞ¡æŢřãĂĆåęĆäÿŃïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  PyObject *o, *bytes;
  char *s;

  if (!PyArg_ParseTuple(args, "U", &o)) {
    return NULL;
  }
  bytes = PyUnicode_AsUTF8String(o);
  s = PyBytes_AsString(bytes);
  print_chars(s);
  Py_DECREF(bytes);
  Py_RETURN_NONE;
}
```

éĂŽè£Ğè£ŹäÿłäŁőæŤźïijÑäÿĂäÿłUTF-8çijŰçăAçŽĎå■Ůçňęäÿšæăźæ■őéIJĂèęĄèćńåĹŻåžžïijÑçĎűåŘČ

```
>>> import sys
>>> s = 'Spicy Jalape\u00f1o'
>>> sys.getsizeof(s)
87
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> sys.getsizeof(s)
87
>>>
```

åęĆædIJä¡ăèŕŢçİĂäijăéĂŠNULLçżŞår¿å■ŮçňęäÿšçżŹctypesåŇĚèčĚè£ĞçŽĎåĞ¡æŢřïijÑ
èęĄæşÍæĎŔçŽĎæŸŕctypesåŔłèČ¡åĚĄèőÿäijăéĂŠå■ŮèŁĆïijÑåźűäÿŤåőČäÿ■äijŽæčĂæ§ëäÿ■éŮť åţÑåĚěçŽĎ

```
>>> import ctypes
>>> lib = ctypes.cdll.LoadLibrary("./libsample.so")
>>> print_chars = lib.print_chars
>>> print_chars.argtypes = (ctypes.c_char_p,)
>>> print_chars(b'Hello World')
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>> print_chars(b'Hello\x00World')
48 65 6c 6c 6f
>>> print_chars('Hello World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ctypes.ArgumentError: argument 1: <class 'TypeError'>: wrong type
>>>
```

åęĆædIJä¡ăæČşäijăéĂŠå■Ůçňęäÿšè ĂÑäÿ■æŸŕå■ŮèŁĆïijÑä¡ăéIJĂèęĄåĚĹæĽğèąÑæĽŃåŁĺçŽĎUTF-8çijŰçăĄãĂĆä¿ŃåęĆïijŽ

```
>>> print_chars('Hello World'.encode('utf-8'))
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>>
```

åŕźäžŐåĚűäżŰæĽ'ľåśŢåůěåĚůïijĹæŕŤåęĆSwigãĂĄCythonïijL'ïijÑ
åIJĺä¡ää¡£çŤĺåőČäžñäijăéĂŠå■ŮçňęäÿšçżŹCäżčçăĄæŮűèęĄåĚĹåě¡åě¡å■ęäźăçŻÿåžŤçŽĎäÿIJèě£äžĘãĂĆ

15.14 äijăéĂŠUnicodeå■ŮçňęäÿšçżŹCåĞ¡æŢřåžŞ

éŮőécŸ

ä¡ăèęĄåĘŹäÿĂäÿłæĽ'ľåśŢæĺąåİŮïijÑéIJĂèęĄåřĘäÿĂäÿłPythonå■ŮçňęäÿšäijăéĂŠçżŹCçŽĎæ§Řäÿłåžş

èğčåĘşæŰźæąĹ

è£ŹéĞŇæĹŚäžňéIJĂèęĄèĂČèŹŚå¿Ĺåď˙ŽçŽĎéŮőécŸïijŇä¡ĘæŸŕæIJĂäÿżèęĄçŽĎéŮőécŸæŸŕçŐřå■Ÿç
åŻăæ■ď˙ïijŇä¡ăçŽĎæŇŚæĹŸæŸŕářĘPythonå■Ůçňęäÿšè¡ňæ■ćäÿžäÿĂäÿłèČ¡èćńCçŘĘèğčçŽĎå¡ćåijŔãĂĆ

äÿžäžĘæijŤçď˙žçŽĎçŻőçŽĎïijŇäÿŃéIćæIJĽäÿď˙äÿłCåĞ¡æŢřïijŇçŤĺæİěæŞ■ä¡IJå■ŮçňęäÿšæŢřæ■őåźűè
äÿĂäÿłä¡£çŤĺå¡ćåijŔäÿž `char *, int` å¡ćåijŔçŽĎå■ŮèŁĆïijŇ
èĂŇåŘęäÿĂäÿłä¡£çŤĺå¡ćåijŔäÿž `wchar_t *, int` çŽĎåő¡å■Ůçňęå¡ćåijŔïijŽ

```
void print_chars(char *s, int len) {
  int n = 0;

  while (n < len) {
    printf("%2x ", (unsigned char) s[n]);
    n++;
  }
  printf("\n");
}

void print_wchars(wchar_t *s, int len) {
  int n = 0;
  while (n < len) {
    printf("%x ", s[n]);
    n++;
  }
  printf("\n");
}
```

åŕźäžŐéİćåŘŚå■ŮèŁĆçŽĎåĞ¡æŢř `print_chars()` ïijŇä¡ăéIJĂèęĄåřĘPythonå■Ůçňęäÿšè¡ňæ■ćäÿžäÿ
8. äÿŃéİćæŸŕäÿĂäÿłè£ŹæăůçŽĎæĽľåśŢåĞ¡æŢřä¿Ńå■ŘïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s;
  Py_ssize_t  len;

  if (!PyArg_ParseTuple(args, "s#", &s, &len)) {
    return NULL;
  }
  print_chars(s, len);
  Py_RETURN_NONE;
}
```

åŕźäžŐéĆčäžŻéIJĂèęĄåď˙ĎçŘĘæIJžåŹĺæIJňåIJř `wchar_t`
çśżåđŃçŽĎåžŞåĞ¡æŢřïijŇä¡ăåŔŕäżěåČŔäÿŃéİćè£ŹæăůçijŰåĘŹæĽľåśŢäžčçăĄïijŽ

```
static PyObject *py_print_wchars(PyObject *self, PyObject *args) {
  wchar_t *s;
  Py_ssize_t  len;

  if (!PyArg_ParseTuple(args, "u#", &s, &len)) {
    return NULL;
```

```
  }
  print_wchars(s,len);
  Py_RETURN_NONE;
}
```

äÿŃéÍćæŸŕäÿĂäÿłäžd'äžŠäijŽèŕÍæÍěæijŤçd'žè£ŹäÿłåĞ¡æŢřæŸŕåęĆä¡Ţåůěä¡IJçŽĎïijŽ

```
>>> s = 'Spicy Jalape\u00f1o'
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> print_wchars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 f1 6f
>>>
```

äžŤçżĘèğĆåŕ§è£ŹäÿłéÍćåŘŠå■ŮèŁĆçŽĎåĞ¡æŢř print_chars() æŸŕæĂŐæăůæŐěåŔŮUTF-8çijŰçăĄæŢřæ■őçŽĎïijŇ äžěåŔŁ print_wchars() æŸŕæĂŐæăůæŐěåŔŮUnicodeçijŰçăĄåĂijçŽĎ

èőÍèőž

åIJÍçżğçż■æIJňèŁĆäźŃåL'■ïijŇä¡ăåžŤèŕěéęŰåĚĹå■ęäźăä¡ăèő£éŮőçŽĎCåĞ¡æŢřåžŞçŽĎçL'źå¿ĄãĂĆ åŕźäžŐå¿Ĺåď'ŽCåĞ¡æŢřåžŞïijŇéĂŽåÿÿäijăéĂŠå■ŮèŁĆèĂŇäÿ■æŸŕå■ŮçňęäÿšäijŽæŕŤè¿Čåě¡äžŽãĂĆèęĄè£

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s;
  Py_ssize_t  len;

  /* accepts bytes, bytearray, or other byte-like object */
  if (!PyArg_ParseTuple(args, "y#", &s, &len)) {
    return NULL;
  }
  print_chars(s, len);
  Py_RETURN_NONE;
}
```

åęĆæđIJä¡ăäż■çĎűè£ŸæŸŕæČşèęĄäijăéĂŠå■ŮçňęäÿšïijŇ ä¡ăéIJĂèęĄç§ěéĄŞPython 3åŔŕä¡£çŤĺäÿĂäÿłåŘĹéĂĆçŽĎå■Ůçňęäÿšèąĺçd'žïijŇ åőČåźűäÿ■çŻť'æŐěæŸăářĎåĹřä¡£çŤĺæăĞåĞĘçśžåđŃ char * æĹŰ wchar_t * ïijĹæŻť'åď'ŽçżĘèŁĆåŔĆèĂČPEP 393ïijL'çŽĎCåĞ¡æŢřåžŞãĂĆ åŻăæ■d'ïijŇèęĄåIJÍCäÿ■èąĺçd'žè£Źäÿłå■ŮçňęäÿšæŢřæ■őïijŇäÿĂäžŻè¡ñæ■ćè£ŸæŸŕå£Ěéążèę ĄçŽĎãĂĆ åIJÍ PyArg_ParseTuple() äÿ■ä¡£çŤĺ"s#" åŠŇ"u#"æăijåijŔåŇŰçĄåŔŕäžěåőL'åĚĺçŽĎæL'ğèąŇè£Źæăůç

äÿ■è£Ğè£Źçğ■è¡ñæ■ćæIJL'äÿłçijžçĆźåřśæŸŕåőČåŔŕèČ¡äijŽåŕijèĞť'åŐ§ågŃå■ŮçňęäÿšåŕźèśąçŽĎåřžåŕÿ äÿĂæŮęè¡ñæ■ćè£ĞåŘŐïijŇäijŽæIJL'äÿĂäÿłè¡ñæ■ćæŢřæ■őçŽĎåď'■åĹűéŹĎåŁăăĹřåŐ§ågŃå■Ůçňęäÿšåŕźèśą ä¡ăåŔŕäžěèğĆåŕ§äÿŃè£Źçğ■æŢĹædIJïijŽ

```
>>> import sys
>>> s = 'Spicy Jalape\u00f1o'
>>> sys.getsizeof(s)
```

```
87
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> sys.getsizeof(s)
103
>>> print_wchars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 f1 6f
>>> sys.getsizeof(s)
163
>>>
```

åŕźäžŐåřŠéĞŔçŽĎå■ŮçňęäÿšåŕźèśąïijŇåŔŕèČ¡æšąäżĂäźĹå¡śåŞ■ïijŇ
ä¡ĘæŸŕåęĆæđIJä¡ăéIJĂèęĄåIJÍæĽľ åśŢäÿ■åď ĎçŘĘåď ğéĞŔçŽĎæŰĞæIJñïijŇä¡ăåŔŕèČ¡æČşéĄ£åĚ■è£ŹäÿłäÿŇéİćæŸŕäÿĂäÿłä£őèőćçĽĹæIJñåŔŕäżěéĄ£åĚ■è£ŹçğåĘĚå■Ÿæ■§èĂŮïijŽ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  PyObject *obj, *bytes;
  char *s;
  Py_ssize_t   len;

  if (!PyArg_ParseTuple(args, "U", &obj)) {
    return NULL;
  }
  bytes = PyUnicode_AsUTF8String(obj);
  PyBytes_AsStringAndSize(bytes, &s, &len);
  print_chars(s, len);
  Py_DECREF(bytes);
  Py_RETURN_NONE;
}
```

èĂŇåŕź `wchar_t` çŽĎåď ĎçŘĘæŮűæČşèęĄéĄ£åĚ■åĘĚå■Ÿæ■§èĂŮåřśæŻť åŁăéŽ¿åŁđäžĘãĂĆ
åIJĺåĘĚéČĺïijŇPythonä¡£çŤĺæIJĂéńŸæŢĹçŽĎèąÍçđ'žæİěå■ŸåĆĺå■Ůçňęäÿšã ĂĆ
ä¿ŃåęĆïijŇåŔłåŇĚåŔńASCIIçŽĎå■Ůçňęäÿšèćńå■ŸåĆĺäÿžå■ŮèŁĆæŢřçżĎïijŇ
èĂŇåŇĚåŔńèŇČåŻť äżŐU+0000åĹřU+FFFFçŽĎå■Ůçňę ŽĎå■Ůçňęäÿšä¡£çŤĺåŔŇå■ŮèŁĆèąÍçđ'žã ĂĆ
çŤśäžŐåŕźäžŐæŢřæ■őçŽĎèąÍçđ'žå¡ćåijŔäÿ■æŸŕå■ŢäÿĂçŽĎïijŇä¡ăäÿ■èČ¡åřĘåĘĚéČĺæŢřçżĎè¡ñæ■ćäÿž
`wchar_t *` çĎűåŘŐæIJ§æIJŻåőČèČ¡æ■čçąőçŽĎåůëä¡IJãĂĆ ä¡ăåžŤèŕěåĹŻåżžäÿĂäÿł
`wchar_t` æŢřçżĎåźűåŘŚåĚűäÿ■åď ■åĹűæŰĞæIJñãĂĆ `PyArg_ParseTuple()`
çŽĎ"u#"æăijåijŔçăĄåŔŕäżěåÿőåŁľä¡ăéńŸæŢĹçŽĎåőŇæĹŘåőČïijĹåőČåřĘåď ■åĹűçżŞæđIJéŹĎåŁăăĹřå■Ůç

åęĆæđIJä¡ăæČşéĄ£åĚ■éŢ£æŮűéŮťåĘĚå■Ÿæ■§èĂŮïijŇä¡ăåŤŕäÿĂçŽĎéĂĽæŃľåŕśæŸŕåď ■åĹűUnicod
åřĘåőČäijăéĂŠçżŹCåĞ¡æŢřïijŇçĎűåŘŐåŽđæŤűè£ŹäÿłæŢřçżĎçŽĎåĘĚå■ŸãĂĆäÿŃéİćæŸŕäÿĂäÿłåŔŕèČ¡çŽ

```
static PyObject *py_print_wchars(PyObject *self, PyObject *args) {
  PyObject *obj;
  wchar_t *s;
  Py_ssize_t len;

  if (!PyArg_ParseTuple(args, "U", &obj)) {
    return NULL;
  }
```

```
  if ((s = PyUnicode_AsWideCharString(obj, &len)) == NULL) {
    return NULL;
  }
  print_wchars(s, len);
  PyMem_Free(s);
  Py_RETURN_NONE;
}
```

åIJÍè£ŹäýłåőđçŐřäÿ■ïijŇPyUnicode_AsWideCharString()
åĹŻåžžäÿĂäýłäýt'æŮűçŽĎwchar_tçijŞåĘšåźűåď■åĹűæŢřæ■őè£ŻåŐžãĂĆ
è£Źäýłçij§åĘšèćńäijăéĂŠçżŹCçĎűåŘŐèćńéĞŁæŤ¿æŐL'ãĂĆ ä¡ĘæŸŕæĹŚåĘŹè£ŹæIJňäźęçŽĎæŮűåĂŹïijŇè

åęĆæđIJä¡ăç§ěéĄŞCåĞ¡æŢřåžŞéIJĂèęĄçŽĎå■ŮèŁĆçijŰçăĄåźűäÿ■æŸŕUTF-8ïijŇ
ä¡ååŔŕäžěåijžåĹűPythonä¡£çŤĺæL'l'åśŢçăĄæÌěæL'ğèąŇæ■čçąőçŽĎè¡ňæ■ćïijŇåřśåČŔäÿŇéÌćè£Źæăůïijž

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  char *s = 0;
  int   len;
  if (!PyArg_ParseTuple(args, "es#", "encoding-name", &s, &len)) {
    return NULL;
  }
  print_chars(s, len);
  PyMem_Free(s);
  Py_RETURN_NONE;
}
```

æIJĂåŘŐïijŇåęĆæđIJä¡ăæČşçŻt'æŐěåď'ĎçŘĘUnicodeå■ŮçňęäÿšïijŇäÿŇéÌćçŽĎæŸŕä¿Ńå■ŘïijŇæijŢç

```
static PyObject *py_print_wchars(PyObject *self, PyObject *args) {
  PyObject *obj;
  int n, len;
  int kind;
  void *data;

  if (!PyArg_ParseTuple(args, "U", &obj)) {
    return NULL;
  }
  if (PyUnicode_READY(obj) < 0) {
    return NULL;
  }

  len = PyUnicode_GET_LENGTH(obj);
  kind = PyUnicode_KIND(obj);
  data = PyUnicode_DATA(obj);

  for (n = 0; n < len; n++) {
    Py_UCS4 ch = PyUnicode_READ(kind, data, n);
    printf("%x ", ch);
  }
  printf("\n");
  Py_RETURN_NONE;
```

```
}
```

åIJÍè£Źäÿłäżčçă Ąäÿ■ïijŇPyUnicode_KIND() åŠŇ PyUnicode_DATA() è£Źäÿď äÿłåőŘåŠŇUnicodeçŽĎåŘŕåŘŸåő¡åžęå■ŸåĆÍæIJĽåĚşïijŇè£ŹäÿłåIJÍPEP 393äÿ■æIJĽæŘŔè£řăĂĆ kind åŘŸéĞŘçijŰçăĄåžŢåśĆå■ŸåĆÍïijĹ8ä¡■ãĂĄ16ä¡■æĹŰ32ä¡■ïijĽäžěåŘŁæŇ åIJÍåőđéŹĚæČĚåĘţäÿ■ïijŇä¡ăåźűäÿ■éIJĂèęĄç§ěéĄŞäżżä¡Ţèů§è£ŹäžŻåĂijæIJĽåĚşçŽĎäÿIJèě£ïijŇ åŘłéIJĂèęĄåIJÍæŘŘåŘŰå■ŮçňęçŽĎæŮűåĂŹářĘåőČäžňäijăçżŹ PyUnicode_READ() åőŘãĂĆ

è£ŸæIJĽæIJĂåŘŐåĞăåŘěïijŽå¡ŞäžŐPythonäijăéĂŠUnicodeå■ŮçňęäÿšçżŹCçŽĎæŮűåĂŹïijŇä¡ăåžŤèŕè åęĆædIJæIJĽUTF-8åŠŇåő¡å■Ůçňęäÿď çğ■éĂĽæŃľïijŇèřůéĂĽæŃľUTF-8. åŕźUTF-8çŽĎæŤŕæŇĄæŻťåŁăæŹőéĄ■äÿĂäžŻïijŇäź§äÿ■åőźæŸŞçŁŕéŤŹïijŇèğčéĞŁåŹĺäź§èČ¡æŤŕæŇĄçŽĎæŻťåě æIJĂåŘŐïijŇçąőä£Ła¡ăäżŤçżĘéŸĚèřżäžĘ åĚşäžŐåď ĎçŘĘUnicodeçŽĎçŻÿåĚşæŰĞæąč

15.15 Cå■Ůçňęäÿšè¡ňæ■ćäÿžPythonå■Ůçňęäÿš

éŮőécŸ

æĂŐæăůåřĘCäÿ■çŽĎå■Ůçňęäÿšè¡ňæ■ćäÿžPythonå■ŮèŁĆæĹŰäÿĂäÿłå■Ůçňęäÿšåŕźèšąïij§

èğčåĘşæŰźæąĹ

Cå■Ůçňęäÿšä¡£çŤĺäÿĂåŕź char * åŠŇ int æĺěèąĺçď žïijŇ ä¡ăéIJĂèęĄåĘşåőŽå■Ůçňęäÿšå Ĺřåžţæ Ÿŕç ŤĺäÿĂäÿłåŐ§ågŃå■ŮèŁĆå■Ůçňęäÿšè£ŸæŸŕäÿĂäÿłUnicodeå■Ůç å■ŮèŁĆåŕźèšąåŘŕäžěåČŘäÿŃéİćè£Źæăůä¡£çŤĺ Py_BuildValue() æĺěædĎåžžijžŽ

```
char *s;      /* Pointer to C string data */
int   len;    /* Length of data */

/* Make a bytes object */
PyObject *obj = Py_BuildValue("y#", s, len);
```

åęĆædIJä¡ăèęĄåĹŽåžžäÿĂäÿłUnicodeå■ŮçňęäÿšijŇåźűäÿŤä¡ăç§ěéĄŞ s æŇĞåŘŠäžĘUTF-8çijŰçăĄçŽĎæŢŕæ■őïijŇåŘŕäžěä¡£çŤĺäÿŃéİćçŽĎæŰźåijŔïijŽ

```
PyObject *obj = Py_BuildValue("s#", s, len);
```

åęĆædIJ s ä¡£çŤĺåĚűäžŰçijŰçăĄæŰźåijŔïijŇéĆčäźĹåŘŕäžěåČŘäÿŃéİćä¡£çŤĺ PyUnicode_Decode() æĺěædĎåžžäÿĂäÿłå■ŮçňęäÿšïijŽ

```
PyObject *obj = PyUnicode_Decode(s, len, "encoding", "errors");

/* Examples /*
obj = PyUnicode_Decode(s, len, "latin-1", "strict");
obj = PyUnicode_Decode(s, len, "ascii", "ignore");
```

åęĆædIJä¡ăæĄřåě¡æIJL'äÿĂäÿłçŤÍ `wchar_t *, len` åŕźèąÍçd'žçŽĎåő¡å■ŮçňęäÿšiijŇ
æIJL'åĞăçğ■éĂL'æŃl'æĂğãĂĆéęŰåĚĹä¡ăåŔřäžěä¡£çŤÍ `Py_BuildValue()` ïijŽ

```
wchar_t *w;     /* Wide character string */
int len;        /* Length */

PyObject *obj = Py_BuildValue("u#", w, len);
```

åŘęåd'ŰïijŇä¡ăè£ŸåŔřäžěä¡£çŤÍ `PyUnicode_FromWideChar()`:

```
PyObject *obj = PyUnicode_FromWideChar(w, len);
```

åŕźäžŐåő¡å■ŮçňęäÿšiijŇåźűæšąæIJL'åŕźå■Ůçňęæţřæ■őè£ŻèąŇèğčædŘâĂŤâĂŤåőČèćńåĄĞåőŽæŸŕåŐ§

èőÍèőž

åřĘCäÿ■çŽĎå■Ůçňęäÿšè¡ňæ■ćäÿžPythonå■ŮçňęäÿšéĄţå¿łåŠŇI/OåŘŇæăůçŽĎåŐ§åĹŹãĂĆ
äź§åřśæŸŕèŕť'ïijŇæÍèèĞłCäÿ■çŽĎæţřæ■őå£Ëéążæăźæ■őäÿĂäžŻèğčçăĄåŹÍèćńæŸ¿åijŔçŽĎèğčçăĄäÿžäÿĂä
éĂŽåÿÿçijŰçăĄæăijåijŔåŇĚæŃňASCIIãĂĄLatin-1åŠŇUTF-8.
åęĆædIJä¡ăåźűäÿ■çąőåőŽçijŰçăĄæŰźåijŔæĹŰèĂĚæţřæ■őæŸŕäžŇè£ŻåĹűçŽĎïijŇä¡ăæIJĂåě¡åřĘå■Ůçňęäÿ
å¡ŞædĎéĂăäÿĂäÿłåŕźèśąçŽĎæŮűåĂŹïijŇPythonéĂŽåÿÿäijŽåd'■åĹűä¡ăæŔŘä¿ŻçŽĎå■Ůçňęäÿšæţřæ■õãĂĆ
åęĆædIJæIJL'å£ĚèęĄçŽĎèŕÍïijŇä¡ăéIJĂèęĄåIJÍåŘŐéÍćåŐžéĞŁæŤ¿Cå■Ůçňęäÿšã Ć
åŘŇæŮűïijŇäÿžäžĘèől'çÍŃåžŔæŻt'åŁăåĄěåčőïijŇä¡ăåžŤèŕěåŘŇæŮűä¡£çŤÍäÿĂäÿłæŇĞéŠĹåŠŇäÿĂäÿłåd'ğ
èĂŇäÿ■æŸŕä¿ÌètŰNULLçżŞår¿æţřæ■őæÍěåĹŻåžžå■Ůçňęäÿšã Ć

15.16 äÿ■çąőåőŽçijŰçăĄæăijåijŔçŽĎCå■Ůçňęäÿš

éŮőécŸ

ä¡ăèęĄåIJÍCåŠŇPythonçŻt'æŐěæÍěåŻđè¡ňæ■ćå■ŮçňęäÿšiijŇä¡ĘæŸŕCäÿ■çŽĎçijŰçăĄæăijåijŔåźűäÿ■ç
ä¿ŃåęĆïijŇåŔŕèČ¡Cäÿ■çŽĎæţřæ■őæIJ§æIJŻæŸŕUTF-8ïijŇä¡ĘæŸŕåźűæšąæIJL'åijžåĹűåőČå£Ëéążæ
ä¡ăæČşçijŰåĘŹäžčçăĄæÍěäžěäÿĂçğ■äijŸéŻĚçŽĎæŰźåijŔåd'ĎçŘĘè£ŹäžŻäÿ■åŘĹæăijæţřæ■õïijŇè£Źæăůå

èğčåĘşæŰźæąĹ

äÿŃéÍćæŸŕäÿĂäžŻCçŽĎæţřæ■őåŠŇäÿĂäÿłåĞ¡æţřæÍěæijŤçd'žè£ŹäÿłéŮőécŸïijŽ

```
/* Some dubious string data (malformed UTF-8) */
const char *sdata = "Spicy Jalape\xc3\xb1o\xae";
int slen = 16;

/* Output character data */
void print_chars(char *s, int len) {
  int n = 0;
  while (n < len) {
    printf("%2x ", (unsigned char) s[n]);
    n++;
```

```
  }
  printf("\n");
}
```

åIJÍè£ŹäÿłäżčçăĄäÿ■ïijŇå■Ůçňęäÿš sdata åŇĚåŘńäžĘUTF-8åŠŇäÿ■åŘĹæăijæŢřæ■őãĂĆ äÿ■è£ĞïijŇåęĆæđIJçŤÍæĹůåIJÍCäÿ■èřČçŤÍ print_chars(sdata, slen) ïijŇåőČçijžèČ¡æ■čåÿÿåůëä¡IJãĂĆ çŐřåIJÍåĄĞèő¿ä¡ăæČşåřĘ sdata çŽĎåĘĚåőźè¡ňæ■ćäÿžäÿĂäÿłPythonå■Ůçňęäÿšã Ć è£ŻäÿĂæ■ěåĄĞèő¿ä¡ăåIJÍåŘŐéÍćè£ŸæČşéĂŽè£ĞäÿĂäÿłæĽľ'åśŢåřĘéĆčäÿłå■Ůçňęäÿšäijääÿł print_chars() åĞ¡æŢřãĂĆ äÿŃéÍćæŸřäÿĂçğ■çŤÍæĺëä£ÍæŁd'åŐ§ågŃæŢřæ■őçŽĎæŰźæşŢïijŇåřśçőŮåć

```
/* Return the C string back to Python */
static PyObject *py_retstr(PyObject *self, PyObject *args) {
  if (!PyArg_ParseTuple(args, "")) {
    return NULL;
  }
  return PyUnicode_Decode(sdata, slen, "utf-8", "surrogateescape");
}

/* Wrapper for the print_chars() function */
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
  PyObject *obj, *bytes;
  char *s = 0;
  Py_ssize_t   len;

  if (!PyArg_ParseTuple(args, "U", &obj)) {
    return NULL;
  }

  if ((bytes = PyUnicode_AsEncodedString(obj,"utf-8",
↪"surrogateescape"))
        == NULL) {
    return NULL;
  }
  PyBytes_AsStringAndSize(bytes, &s, &len);
  print_chars(s, len);
  Py_DECREF(bytes);
  Py_RETURN_NONE;
}
```

åęĆæđIJä¡ăåIJÍPythonäÿ■ářÌèřŢè£ŹäžŽåĞ¡æŢřïijŇäÿŃéÍćæŸŕè£ŘèąŇæŢĹæđIJïijŽ

```
>>> s = retstr()
>>> s
'Spicy JalapeÃśo\udcae'
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f ae
>>>
```

äžŤçżĘèğĆåŕ§çżŞæđIJä¡ääijŽåŔŠçŐřïijŇäÿ■åŘĹæăijå■ŮçňęäÿšèćńçijŰçĂąĹřäÿĂäÿłPythonå■Ůçňęäÿ

åźűäÿŤå¡ŞåőČèćńåŻđäijäçżŹCçŽDæŮűåĂŹïijŇèćńè¡ňæ■ćäÿžåŠŇäźŇåĽ■åŐ§ågŃCå■Ůçňęäÿšäÿ Ăæăůç ŽDå

èőÍèőž

æIJňèŁĆåśŢçd'žäžĘåIJÍæĽl'åśŢæÍąåİŮäÿ■åd'DçŘĘå■Ůçňęäÿšæ Ůűäij ŽéĚ■åĹřçŽDäÿĂäÿłæčŸæĽŇåŔÍ
äź§åřśæŸřèřt'ïijŇåIJÍæĽl'åśŢäÿ■çŽDCå■ŮçňęäÿšåŕřèČ¡äÿ■äijŽäÿěæăijéĄţå¿łPythonæĽĂæIJ§æIJŻçŽDUni
åŻăæ■d'ïijŇå¿ĹåŔřèČ¡äÿĂäžŻäÿ■åŘĹæăijCæŢřæ■őäijăéĂŠåĹřPythonäÿ■åŐżãĂĆ
äÿĂäÿłå¿Ĺåě¡çŽDä¿Ňå■Řåřśæ ŸřæűĽåŔŁåĹřåžŢåśĆçşżçż§èřČçŤĺæŕŤåęĆæŰĞäżűåŘ■è£Źæăůç ŽDå■Ůçňęä
ä¿ŇåęĆïijŇåęĆædIJäÿĂäÿłçşżçż§èřČçŤĺè£ŤåŻđçżŹèğčéĞŁåŹĺäÿĂäÿłæ■§åİŔçŽDå■ŮçňęäÿšïijŇäÿ■èČ¡èćńa

äÿĂèĹňæİěèőšïijŇåŔřäžěéĂŽè£ĞåĹűåőŽäÿĂäžŻéŤŹèřřç■ŰçŢěærŤåęĆäÿěæăijãĂĄå£¡çŢěãĂĄæŻ£äżč
äÿ■è£ĞïijŇè£ŹäžŻç■ŰçŢěçŽDäÿĂäÿłçijžçĆźæŸřåőČäżňærÿäźĚæĂğçăť'åİŔäžĘåŐ§ågŃå■Ůçňęäÿšç ŽDåĘĚ
ä¿ŇåęĆïijŇåęĆædIJä¿Ňå■Řäÿ■çŽDäÿ■åŘĹæăijæŢřæ■őä¡£çŤĺè£ŹäžŻç■ŰçŢěäźŇäÿĂèğčçăĄïijŇä¡äăijŽå¿Ů

```
>>> raw = b'Spicy Jalape\xc3\xb1o\xae'
>>> raw.decode('utf-8','ignore')
'Spicy JalapeÃńo'
>>> raw.decode('utf-8','replace')
'Spicy JalapeÃńo?'
>>>
```

surrogateescape éŤŹèřřåď'DçŘĘç■ŰçŢěäijŽåřĘæĽĂæIJL'äÿ■åŔřèğčçăĄå■ŮèŁĆè¡ňåŇŰäÿžäÿĂä
ä¿ŇåęĆïijŽ

```
>>> raw.decode('utf-8','surrogateescape')
'Spicy JalapeÃńo\udcae'
>>>
```

å■ŢçŇňçŽDä¡Őä¡■äžčçŘĘå■Ůçňęæŕ ŤåęĆ \udcae åIJÍUni-
codeäÿ■æŸřéİđæşŢçŽDãĂĆ åŽăæ■d'ïijŇè£Źäÿłå■ŮçňęäÿšåřśæŸřäÿĂäÿłéİđæşŢèąĺçď'žãĂĆ
åőđéŹĚäÿŁïijŇåęĆædIJä¡ăåřĘåőČäijăäÿłäÿĂäÿłæĽğèąŇè¿ŞåĞžçŽDåĞ¡æŢřïijŇä¡äăijŽå¿ŮåĹřäÿĂäÿłéŤŹèřı

```
>>> s = raw.decode('utf-8', 'surrogateescape')
>>> print(s)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
UnicodeEncodeError: 'utf-8' codec can't encode character '\udcae'
in position 14: surrogates not allowed
>>>
```

çDűèĂŇïijŇåĚąèőÿäžčçŘĘè¡ňæ■ćçŽDåĚşéŤőçĆźåIJĺäžŐäżŐCäijăçżŹPythonåŔĹåŻđäijăçżŹCçŽDäÿ■
å¡Şè£Źäÿłå■Ůçňęäÿšåę■æěňąä¡£çŤĺ surrogateescape çijŰçăĄæŮűïijŇäžčçŘĘå■ŮçňęäijŽè¡ňæ■ćåŻđåŐ

```
>>> s
'Spicy JalapeÃńo\udcae'
>>> s.encode('utf-8','surrogateescape')
b'Spicy Jalape\xc3\xb1o\xae'
>>>
```

ä¡IJäÿžäÿĂèĹňåĞĘåĹŹïijŇæIJĂåě¡éĄ£åĚ■äžčçŘĘçijŰçăĄâĂŤâĂŤåęĆæđIJä¡ăæ■čçąőçŽĎä¡£çŤĺäžĘçij
äÿ■è£ĞïijŇæIJL'æŮűåĂŹçąőåőđäijŽåĞžçŐřä¡ăåźűäÿ■èČ¡æŐğåĹűæŢřæ■őçijŰçăĄåźűäÿŤä¡ăåŔĹäÿ■èČ¡å£¡ç
éĆčäźĹåřśåŔŕäžěä¡£çŤĺæIJňèŁĆçŽĎæŁĂæIJŕäžĘãĂĆ

æIJĂåŘŐäÿĂçĆźèęĄæşĺæĎŔçŽĎæŸŕïijŇPythonäÿ■èőÿåď'ŽéĺćåŘŚçşżçż§çŽĎåĞ¡æŢřïijŇçĽźåĹńæŸŕå
éČ¡äijŽä¡£çŤĺäžčçŘĘçijŰçăĄãĂĆä¿ŃåęĆïijŇåęĆæđIJä¡ăä¡£çŤĺåČŔ `os.listdir()`
è£ŹæăůçŽĎåĞ¡æŢřïijŇ äijååĚěäÿĂäÿłåŇĚåŘńäžĘäÿ■åŔŕèğčçăĄæŰĞäžűåŘ■çŽĎçŻőå¡ŢçŽĎèŕİïijŇåőČäijŽ
åŔĆèĂČ5.15çŽĎçŻÿåĚşçńäèŁĆãĂĆ

PEP 383 äÿ■æIJL'æŻť'åď'ŽåĚşäžŐæIJňæIJžæŔŘåĹřçŽĎäžěåŔŁåŠŇsurrogateescapeéŤŹèŕŕåď'ĎçŘĘçŻÿåĚşçŽĎä£ąæĄŕãĂĆ

15.17 äijåéĂŠæŰĞäžűåŘ■çżŹCæĽľåśŢ

éŮőécŸ

ä¡ăéIJĂèęĄåŘŚCåžŞåĞ¡æŢřäijåéĂŠæŰĞäžűåŘ■ïijŇä¡ĘæŸŕéIJĂèęĄçąőä£ÍæŰĞäžűåŘ■æăźæ■őçşżçż§a

èğčåĘşæŰźæąĹ

åĘŹäÿĂäÿłæŐěåŔŮäÿĂäÿłæŰĞäžűåŘ■äÿžåŔĆæŢřçŽĎæĽľåśŢåĞ¡æŢřïijŇåęĆäÿŃè£ŹæăůïijŽ

```
static PyObject *py_get_filename(PyObject *self, PyObject *args) {
  PyObject *bytes;
  char *filename;
  Py_ssize_t len;
  if (!PyArg_ParseTuple(args,"O&", PyUnicode_FSConverter, &bytes)) {
    return NULL;
  }
  PyBytes_AsStringAndSize(bytes, &filename, &len);
  /* Use filename */
  ...

  /* Cleanup and return */
  Py_DECREF(bytes)
  Py_RETURN_NONE;
}
```

åęĆæđIJä¡ăåůšçżŔæIJL'äžĘäÿĂäÿł `PyObject *` ïijŇåÿŇæIJŻårĘåĚűè¡ñæ■ćæĹŘäÿĂäÿłæŰĞäžűåŘ■ï

```
PyObject *obj;    /* Object with the filename */
PyObject *bytes;
char *filename;
Py_ssize_t len;

bytes = PyUnicode_EncodeFSDefault(obj);
PyBytes_AsStringAndSize(bytes, &filename, &len);
/* Use filename */
...
```

```
/* Cleanup */
Py_DECREF(bytes);

If you need to return a filename back to Python, use the following␣
↪code:

/* Turn a filename into a Python object */

char *filename;       /* Already set */
int   filename_len;   /* Already set */

PyObject *obj = PyUnicode_DecodeFSDefaultAndSize(filename, filename_
↪len);
```

èőÍèőž

äžěåŘŕçğžæď■æŰźåijŔæÍěåď ĎçŘĘæŰĞäžűåŘ■æŸŕäÿĂäÿłå¿ĹæčŸæĽ'ŃçŽĎéŮőécŸïijŇæIJĂåŘŐäžď åęĆæđIJä¡ăåIJÍæĽ'ľåśŢäžčçăĄäÿ■ä¡£çŤÍæIJňèŁĆçŽĎæŁĂæIJŕïijŇæŰĞäžűåŘ■çŽĎåď ĎçŘĘæŰźåijŔåŠŇåŠ åŇĚæŃňçijŰçăĄ/çŢŇéÍćå■ŮèŁĆïijŇåď ĎçŘĘåÍŔå■Ůçňęïij Ňäžčç ŘĘè¡ňæ■ćåŠŇåĚűäžŰåď ■æÍĆæČĚåĘţãĂ

15.18 äijăéĂŠåůšæĽŞåijĂçŽĎæŰĞäžűçżŹCæĽ'ľåśŢ

éŮőécŸ

ä¡ăåIJÍPythonäÿ■æIJL'äÿĂäÿłæĽŞåijĂçŽĎæŰĞäžűåŕźèśąïijŇä¡ĘæŸŕéIJĂèęĄåřĘåőČäijăçżŹèęĄä¡£çŤÍ

èğčåĘşæŰźæąĹ

èęĄåřĘäÿĂäÿłæŰĞäžűè¡ňæ■ćäÿžäÿĂäÿłæŢť'åđŃçŽĎæŰĞäžűæŔŔè£řçňęïijŇä¡£çŤÍ `PyFile_FromFd()` ïijŇåęĆäÿŃïijŽ

```
PyObject *fobj;     /* File object (already obtained somehow) */
int fd = PyObject_AsFileDescriptor(fobj);
if (fd < 0) {
   return NULL;
}
```

çżŞæđIJæŰĞäžűæŔŔè£řçňęæŸŕéĂŽè£ĞèřČçŤÍ `fobj` äÿ■çŽĎ `fileno()` æŰźæşŢèŐůå¿ŮçŽĎãĂĆ åŽäæ■ď'ïijŇäżżä¡Ţäžčè£Źçğ■æŰźåijŔæŽť'éIJšçżŹäÿĂäÿłæŔŔè£řåŹĺçŽĎåŕźèśąéČ äÿĂæŮęä¡ăæIJL'äžĘè£ŹäÿłæŔŔè£řåŹĺïijŇåőČåřśèČ¡èćńäijăéĂŠçżŹåď'Žäÿłä¡ÕçžğçŽĎåŔŕåď'ĎçŘĘæŰĞäžű

åęĆæđIJä¡ăéIJĂèęĄè¡ňæ■ćäÿĂäÿłæŢť'åđŃæŰĞäžűæŔŔè£řçňęäÿžäÿĂäÿłPythonåŕźèśąïijŇéĂĆçŤÍäÿŃé `PyFile_FromFd()`:

```
int fd;      /* Existing file descriptor (already open) */
PyObject *fobj = PyFile_FromFd(fd, "filename","r",-1,NULL,NULL,NULL,
↪1);
```

PyFile_FromFd() çŽĎåŔĆæŢřåŕźåžŤåĘĚç¡őçŽĎ open() åĞ¡æŢřãĂĆ NULL-Lèąĺçď žçijŰçăĄãĂĄéŤŹèŕŕåŠÑæ■ćèąŇåŔĆæŢřä¡£çŤĺéžŸèőď åĂijãĂĆ

èőÍèőž

åęĆæđIJåřĘPythonäÿ■çŽĎæŰĞäžűåŕźèšąäijąçžŹĆïijŇæIJL'äÿĂäžŽæşÍæĎŔäžŃéąźãĂĆ éęŰåĚĹïijŇPythonéĂŽè£Ğ io æĺąåİŮæL'ğèąŇèĞłåůśçŽĎI/OçijŞåĘšãĂĆ åIJÍäijãéĂŠäžżä¡Ţçśžåđ ŃçŽĎæŰĞäžűæŔŔè£řçňęçżŹĆäźŃåL'■ïijŇä¡ăéČ¡èęĄéęŰåĚĹåIJÍçŽÿåžŤæŰĞäžűåŕźäÿ■çĎűçŽĎèŕÍïijŇä¡ăäijŽæL'Şäźśæ ŰĞäžűçşżçż§äÿŁéÍćçŽĎæŢřæ■őãĂĆ

åĚűæňąïijŇä¡ăéIJĂèęĄçL'źåĹńæşÍæĎŔæŰĞäžűçŽĎå¡Šåśđè ĂĚäžěåŔŁåĚşéŮ■æŰĞäžűçŽĎèĄŇèt'čãĂĆ åęĆæđIJäÿĂäÿłæŰĞäžűæŔŔè£řçňęèćńäijąçžŹĆïijŇä¡ĘæŸŕåIJÍPythonäÿ■è£ŸåIJÍèćńä¡£çŤÍçİĂïijŇä¡ăéIJĂèęçśżäijijçŽĎïijŇåęĆæđIJäÿĂäÿłæŰĞäžűæŔŔè£řçňęèćńè¡ňæ■ćäÿžäÿĂäÿłPythonæŰĞäžűåŕźèšąïijŇä¡ăéIJĂèęPyFile_FromFd() çŽĎæIJĂåŘŐäÿĂäÿłåŔĆæŢřèćńèő¿ç¡őæĹŘ1ïijŇçŤÍæİěæŇĞåĞžPythonåžŤèŕěåĚşéŰ

åęĆæđIJä¡ăéIJĂèęĄäžŐCæăĞåĞĘI/OåžŞäÿ■ä¡£çŤÍåęĆãĂĂfdopen() åĞ¡æŢřæİěåĹŻåžžäÿ■åŘŇçśžåđŃçŽĎæŰĞäžűåŕźèšąæŕŤåęĆ FILE * åŕźèšąïijŇä¡ăéIJĂèęĄçL'źåĹńåřŔå£ĆäžĘãĂĆè£ŹæăůåĄŽäijŽåIJÍI/OåăĘæăĹäÿ■äžğçŤ§äÿd'äÿłåőŇåĚÍäÿ■åŘŇçŽĎI/Oçïijĺäÿ■äÿłæŸŕæİěèĞłPythonçŽĎ io æĺąåİŮïijŇåŔęäÿĂäÿłæİěèĞłCçŽĎ stdio ïijL'ãĂĆ åČŔCäÿ■çŽĎ fclose() äijŽåĚşéŮ■PythonèęĄä¡£çŤÍçŽĎæŰĞäžűãĂĆ åęĆæđIJèől'ä¡ăéĂL'çŽĎèŕÍïijŇä¡ăåžŤèŕěäijŽéĂL'æŇl'åŐžæđĎåžžäÿĂäÿłæL'l'åśŢäžčçăĄæİěåd'ĎçŘĘåžŢåśĆèĂŇäÿ■æŸŕä¡£çŤÍæİěèĞł<stdio.h>çŽĎénŸåśĆæŁ¡èśąåŁ§èČ¡ãĂĆ

15.19 äžŐCèŕ■èÍĂäÿ■èŕžåŔŰçśżæŰĞäžűåŕźèšą

éŮőécŸ

ä¡ăèęĄåĘŹCæL'l'åśŢæİěèŕžåŔŰæİěèĞłäžżä¡ŢPythonçśżæŰĞäžűåŕźèšąäÿ■çŽĎæŢřæ■őïijĹæŕŤåęĆæŹőé

èğčåĘşæŰźæąĹ

èęĄèŕžåŔŰäÿĂäÿłçśżæŰĞäžűåŕźèšąçŽĎæŢřæ■őïijŇä¡ăéIJĂèęĄéĞ■åd'■èřČçŤÍ read() æŰźæşŢïijŇçĎűåŘŐæ■čçąőçŽĎèğčçăĄèŐůå¿ŮçŽĎæŢřæ■őãĂĆ

äÿŃéÍćæŸŕäÿĂäÿłCæL'l'åśŢåĞ¡æŢřä¿Ńå■ŘïijŇäžĚäžĚåŔłæŸŕèŕžåŔŰäÿĂäÿłçśżæŰĞäžűåŕźèšąäÿ■çŽĎ

```
#define CHUNK_SIZE 8192

/* Consume a "file-like" object and write bytes to stdout */
static PyObject *py_consume_file(PyObject *self, PyObject *args) {
  PyObject *obj;
  PyObject *read_meth;
  PyObject *result = NULL;
```

```
  PyObject *read_args;

  if (!PyArg_ParseTuple(args,"O", &obj)) {
    return NULL;
  }

  /* Get the read method of the passed object */
  if ((read_meth = PyObject_GetAttrString(obj, "read")) == NULL) {
    return NULL;
  }

  /* Build the argument list to read() */
  read_args = Py_BuildValue("(i)", CHUNK_SIZE);
  while (1) {
    PyObject *data;
    PyObject *enc_data;
    char *buf;
    Py_ssize_t len;

    /* Call read() */
    if ((data = PyObject_Call(read_meth, read_args, NULL)) == NULL)
↪{
      goto final;
    }

    /* Check for EOF */
    if (PySequence_Length(data) == 0) {
      Py_DECREF(data);
      break;
    }

    /* Encode Unicode as Bytes for C */
    if ((enc_data=PyUnicode_AsEncodedString(data,"utf-8","strict
↪"))==NULL) {
      Py_DECREF(data);
      goto final;
    }

    /* Extract underlying buffer data */
    PyBytes_AsStringAndSize(enc_data, &buf, &len);

    /* Write to stdout (replace with something more useful) */
    write(1, buf, len);

    /* Cleanup */
    Py_DECREF(enc_data);
    Py_DECREF(data);
  }
  result = Py_BuildValue("");
```

```
 final:
  /* Cleanup */
  Py_DECREF(read_meth);
  Py_DECREF(read_args);
  return result;
}
```

èęĄæţŃèŕŢè£Źäÿłäžčçă Ąïij ŇåĚĹæđĎéĂääÿĂäÿłçśżæŰĞäžűåŕźèśąæŕŤåęĆäÿĂäÿłStringIOåőđä¿ŃïijŇç

```
>>> import io
>>> f = io.StringIO('Hello\nWorld\n')
>>> import sample
>>> sample.consume_file(f)
Hello
World
>>>
```

èőĺèőž

åŠŇæŹőéĂŽçşżçż§æŰĞäžűäÿ■åŘŇçŽĎæŸŕïijŇäÿĂäÿłçśżæŰĞäžűåŕźèśąåźűäÿ■éIJĂèęĄä¡£çŤĺä¡ŐçžğĂ
åŻăæ■d’ïijŇä¡ääÿ■èČ¡ä¡£çŤĺæŹőéĂŽçŽĎCåžŞåĞ¡æŢřæİèèő£éŮőåőČãĂĆ
ä¡ăéIJĂèęĄä¡£çŤĺPythonçŽĎC APIæİèåČŔæŹőéĂŽæŰĞäžűçśżäijijçŽĎéĆčæăůæŞ■ä¡IJçśżæŰĞäžűåŕźèśąãĂ

åIJĺæĹŚäžňçŽĎèğčåĘşæŰźæąĹäÿ■ïijŇ`read()` æŰźæşŢäžŐèćńäijăéĂŠçŽĎåŕźèśąäÿ■æŔŘåŔŰåĞžæİě
äÿĂäÿłåŔĆæŢřåĹŮèąĺèćńæđĎåżžçĎűåŘŐäÿ■æŰ■çŽĎèćńäijăçżŹ `PyObject_Call()`
æİèèřČçŤĺè£ŹäÿłæŰźæşŢãĂĆ èęĄæčĂæ§ěæŰĞäžűæIJńář¿ïijĹEOFïijL’ïijŇä¡£çŤĺäžĘ
`PySequence_Length()` æİěæ§ěçIJŃæŸŕåŘęè£ŤåŻđåŕźèśąéŢ£åžęäÿž0.

åŕźäžŐæL’ĂæIJL’çŽĎI/OæŞ■ä¡IJïijŇä¡ăéIJĂèęĄåĚşæşĺåžŢåśĆçŽĎçijŰçăĄæăijåijŔïijŇè£ŸæIJL’å■ŮèŁ
æIJňèŁĆæijŤçd’žäžĘåęĆä¡Ţäžěæ ŰĞæIJňæĺąåijŔèŕžåŔŰäÿĂäÿłæŰĞäžűåźűårĘçżŞæđIJæŰĞæIJňèğčçăĄäÿž
åęĆæđIJä¡ăæČşäžěäžŇè£ŻåĹűæĺąåijŔèŕžåŔŰæŰĞäžűïijŇåŔłéIJĂèęĄä£őæŤźäÿĂçĆźçĆźå■şåŔïijŇä¿ŃåęĆ

```
...
/* Call read() */
if ((data = PyObject_Call(read_meth, read_args, NULL)) == NULL) {
  goto final;
}

/* Check for EOF */
if (PySequence_Length(data) == 0) {
  Py_DECREF(data);
  break;
}
if (!PyBytes_Check(data)) {
  Py_DECREF(data);
  PyErr_SetString(PyExc_IOError, "File must be in binary mode");
  goto final;
}
```

```
/* Extract underlying buffer data */
PyBytes_AsStringAndSize(data, &buf, &len);
...
```

æIJňèŁĆæIJĂéŽ¿çŽĎåIJřæŰźåIJÍäžŐåęĆä¡Ţè£ŻèąÑæ■čçąőçŽĎåĘĚå■Ÿçőąç ŘĘãĂĆ å¡Şåď ĎçŘĘ `PyObject * `` åŔŸéĞŔçŽĎæŮűåĂŹïijÑéIJĂèęĄæşÍæĎŔçőąçŘĘåijŢçŤÍèőąæŢi åŕź ``Py_DECREF()` çŽĎèřČçŤÍåřśæŸŕæİěåĄŽè£ŹäýłçŽĎãĂĆ

æIJňèŁĆäžčçăĄäžëäÿĂçğ■éĂŽçŤÍæŰźåijŔçijŰåĘŹïijÑåŻäæ■ď'äžŰäź§èČ¡éĂĆçŤÍäžŐåĚűäżŰçŽĎæŰČ ä¿ÑåęĆïijÑèęĄåĘŹæŢřæ■őïijÑåŔłéIJĂèęĄèŐůåŔŰçśżæŰĞäžűåŕźèśąçŽĎ `write()` æŰźæşŢïijÑářĘæŢřæ■őè¡ňæ■ćäÿžåŘĹéĂĆçŽĎPythonåŕźèśą ïijĹå■ŮèŁĆæĹŰUnicodeïijL'ïijÑçĎűåŘŐèřČçŤÍèŕěæŰźæşŢářĘè¿ŞåĚěåĘŹåĚěåĽřæŰĞäžűãĂĆ

æIJĂåŘŐïijÑář¡çőąçśżæŰĞäžűåŕźèśąéĂŽåÿÿè£ŸæŔŘä¿ŻåĚűäżŰæŰźæşŢïijĹæŕŤåęĆreadline(), read_info()ïijL'ïijÑ æĹŚäžňæIJĂåě¡åŔłä¡£çŤÍå§žæIJńçŽĎ `read()` åŠŇ `write()` æŰźæşŢãĂĆ åIJÍåĘŹCæL'l'åśŢçŽĎæŮűåĂŹïijÑèČ¡çőĂå■Ţåřśář¡éĞŔçőĂå■ŢãĂĆ

15.20 åď ĎçŘĘCèŕ■èÍĂäÿ■çŽĎåŔŕè£■äžčåŕźèśą

éŮőécŸ

ä¡ăæČşåĘŹCæL'l'åśŢäžčçăĄåď ĎçŘĘæİěèĞłäżżä¡ŢåŔŕè£■äžčåŕźèśąåęĆåĹŮèąÍãĂĄåĚČçżĎãĂĄæŰĞäž

èğčåĘşæŰźæąĹ

äÿŃéİćæŸŕäÿĂäÿłCæL'l'åśŢåĞ¡æŢřä¿Ńå■ŘïijÑæijŤçď'žäžĘæĂŐæăůåď ĎçŘĘåŔŕè£■äžčåŕźèśąäÿ■çŽĎ

```
static PyObject *py_consume_iterable(PyObject *self, PyObject␣
↪*args) {
  PyObject *obj;
  PyObject *iter;
  PyObject *item;

  if (!PyArg_ParseTuple(args, "O", &obj)) {
    return NULL;
  }
  if ((iter = PyObject_GetIter(obj)) == NULL) {
    return NULL;
  }
  while ((item = PyIter_Next(iter)) != NULL) {
    /* Use item */
    ...
    Py_DECREF(item);
  }

  Py_DECREF(iter);
  return Py_BuildValue("");
}
```

èőÍèőž

æIJňèŁĆäÿ■çŽĎäžčçăĄåŠŇPythonäÿ■åŕźåžŤäžčçăĄçśżäijijãĂĆ `PyObject_GetIter()` çŽĎèřČçŤĺåŠŇèřČçŤÍ `iter()` äÿĂæăůåŘŕèŐůå¿ŮäÿĂäÿłè£■äžčåŹÍãĂĆ `PyIter_Next()` åĞ¡æŢřèřČçŤÍ `next` æŰźæşŢè£ŤåŻđäÿŃäÿĂäÿłåĚČçť'ăæĹŰNULL(åęĆæđIJæšąæIJL'åĚČçť'ăäžĘ)ãĂĆ èęĄæşÍæĎŔæ■čçąőçŽĎåĘĚå■Ÿçőąç ŘĘâĂŤâĂŤ `Py_DECREF()` éIJĂèęĄåŘŇæŮűůåIJÍäžğçŤ§çŽĎåĚČçť'ăåŠŇè£■äžčåŹĺåŕźèśąæIJňèžńäÿŁåŘŇæŮűèćńèřČçŤÍïijŇ äžěéĄ£åĚ■åĞžçŐřåĘĚå■ŸæşĎéIJšãĂĆ

15.21 èŕŁæŰ■åĹĘæőţéŤŹèŕŕ

éŮőécŸ

èğčéĞŁåŹÍåŻääÿžæ§Řäÿłå ĹĘæőţéŤŹèŕŕãĂĄæĂżçž£éŤŹèŕŕãĂĄèő£éŮőèűŁçŢŇæĹŰůĚűäžŰèĞť'åŚ¡éŤ ä¡ăæČşèŐůå¿ŮPythonåăĘæăĹä£ąæĄŕïijŇäžŐèĂŇæL'¿åĞžåIJÍåŘŚçŤ§éŤŹèŕŕçŽĎæŮűůåĂŹä¡ăçŽĎçÍŃåžŘè£

èğčåĘşæŰźæąĹ

`faulthandler` æÍąåİŮèČ¡èćńçŤÍæİěåÿőä¡äèğčåĘşè£ŹäÿłéŮőécŸãĂĆ åIJÍä¡ăçŽĎçÍŃåžŘäÿ■åijŢåĚëäÿŃåĹŮäžčçăĄïijŽ

```
import faulthandler
faulthandler.enable()
```

åŘęåď'Ûè£ŸåŘŕäžěåČŘäÿŃéİćè£ŹæăůäĄ£çŤÍ `-Xfaulthandler` æİěè£ŘèąŇPythonïijŽ

```
bash % python3 -Xfaulthandler program.py
```

æIJĂåŘŐïijŇä¡ăåŘŕäžěèő¿ç¡ő `PYTHONFAULTHANDLER` çŐŕåćČåŔŸéĞŘãĂĆ åijĂåŘŕ-faulthandleråŘŐïijŇåIJÍCæL'ľåśŢäÿ■çŽĎèĞť'åŚ¡éŤŹèŕŕäijŽåŕijèĞť'äÿĂäÿłPythonéŤŹèŕŕåăĘæăĹèćńæL'Şå■ř

```
Fatal Python error: Segmentation fault

Current thread 0x00007fff71106cc0:
  File "example.py", line 6 in foo
  File "example.py", line 10 in bar
  File "example.py", line 14 in spam
  File "example.py", line 19 in <module>
Segmentation fault
```

åř¡çőąè£Źäÿłåźűäÿ■èČ¡åŚŁèŕL'ä¡ăCäžčçăĄäÿ■åŞłéĞŇåĞžéŤŹäžĘïijŇä¡ĘæŸŕèĞşåŕŚèČ¡åŚŁèŕL'ä¡ăPyth

èőÍèőž

faulthandleräijŽåIJÍPythonäžčçăĄæL’ğèąŇåĞžéŤŹçŽĎæŮűåĂŹåŘŚä¡ăåśŢçď’žèů§èÿłä£ąæĄŕãĂĆ
èĞşåřŚïijŇåőČäijŽåŚŁèŕL’ä¡ăåĞžéŤŹæŮűèćńèřČçŤÍçŽĎæIJĂéąűçžğæL’ľ’åśŢåĞ¡æŢřæŸŕåŞłäÿłãĂĆ
åIJÍpdbåŠŇåĚűäžŰPythonèřČèŕŢåŹÍçŽĎåÿőåŁľ’äÿŇïijŇä¡ăåřśèČ¡è£¡æäźæžŕæžŘæL’¿åĹřéŤŹèŕŕæL’ĂåIJÍçŽĎ

faulthandleräÿ■äijŽåŚŁèŕL’ä¡ăäżżä¡ŢTCèŕ■èÍĂäÿ■çŽĎéŤŹèŕŕä£ąæĄŕãĂĆ
åŽăæ■d’ïijŇä¡ăéIJĂèęĄä¡£çŤÍäijăçż§çŽĎCèřČèŕŢåŹÍïijŇæŕŤåęĆgdbãĂĆ
äÿ■è£ĞïijŇåIJÍfaulthandlerè£¡èÿłä£ąæĄŕåŘŕäžěèőľ’ä¡ăåŐżåĹd’æŰ■äżŐåŞłéĞŇçİĂæL’ŃãĂĆ
è£ŸèęĄæşÍæĎŘçŽĎæŸŕåIJÍCäÿ■æ§ŘäžŽçśżåđŇçŽĎéŤŹèŕŕåŘŕèČ¡äÿ■åd’łăőźæŸŞæĄćåd’■ãĂĆ
ä¿ŃåęĆïijŇåęĆæđIJäÿĂäÿłCæL’ľåśŢäÿćâijČäžEçÍŃåžŘåäĘæĂĹä£ąæĄŕïijŇåőČäijŽèőľ’faulthandleräÿ■åŘŕçŤ
éĆčäźĹä¡ääź§å¿Ůäÿ■åĹřäżżä¡Ţè¿ŞåĞžïijĹéŹd’ážEçÍŃåžŘåěŤæžČåd’ŰïijL’ãĂĆ

éŹĎå¡ŢA

åIJÍçž£èţĎæžŘ

http://docs.python.org

åęĆæđIJä¡ăéIJĂèęĄæůśåĚěäžĘèğčæŐćçľ’űěŕ■èÍĂåŠŇæÍąåİŮçŽĎçżĘèŁĆïijŇéĆčäźĹäÿ■å£Ěèŕť’ïijŇPyth
3 çŽĎæŰĞæąčèĂŇäÿ■æŸŕäžěåL’■çŽĎèĂĄçL’ĹæIJň

http://www.python.org/dev/peps

åęĆæđIJä¡ääŘŞçŘĘèğčäÿžpythonèŕ■èÍĂæůżåŁăæŰřçL’źæĂğçŽĎåŁÍæIJžäžěåŘŁåőđçŐřçŽĎçżĘèŁĆïijŇ
Enhancement Proposals—-PythonåijĂåŘŞçijŰçăĄèğĎèŇČïijL’çżİåŕźæŸŕéİđåÿÿåőÌèť’ţçŽĎèţĎæžŘãĂĆåřď’å

http://pyvideo.org

è£ŹéĞŇæIJL’æİèèĞłæIJĂè£ŞçŽĎPyConåď’ğäijŽãĂĄçŤÍæĹůçżĎèğĄéİćäijŽç■L’çŽĎåď’ğéĞŔèğĘécŚæi
3äÿ■æůżåŁăçŽĎçŽĎæŰřçL’źæĂğãĂĆ

http://code.activestate.com/recipes/langs/python

éŢ£æIJ§äžěæİěïijŇActiveStateçŽĎPythonçL’ĹåİŮåůšçżŔæĹŘäÿžäÿĂäÿłæL’¿åĹřæŢřäžěå■Čèőąç ŽĎéŠĹ

http://stackoverflow.com/questions/tagged/python

Stack Overflow çŻőåL’■æIJL’èűĚè£Ğ175,000äÿłéŮőécŸècńæăĞèőřäÿžPythonçŻÿåĚşïijÍèĂŇåĚűäÿ■åd’
3çŽĎïijL’ãĂĆår¡çőąéŮőécŸåŠŇåŻđç■ŤçŽĎèť’íéĞŔäÿ■åŘŇïijŇä¡ĘæŸŕäž■çĎűèČ¡åŘŚçŐřå¿Ĺåď’Žåě¡äijŸçğ

Pythonå■ęäźääźęçś■

äÿŃéİćè£ŹäžŻäźęçś■æŔŘä¿ŻäžĘåŕźPythonçijŰçÍŃçŽĎåĚěéŮÍäżŃçż■ïijŇäÿŤéĞ■çĆźæŤ¿åIJÍäžĘPytho
3äÿŁãĂĆ

Beginning Python: From Novice to Professional, 2nd Edition, by Magnus Lie HetâĂŘ land, Apress (2008). Programming in Python 3, 2nd Edition, by Mark Summerfield, Addison-Wesley (2010).

- *Learning Python*ïijÑçňňåŻŻçL'Ĺ ïijÑä¡IJèĂĚ Mark Lutzïij Ñ OâĂŹReilly & Associates åĞžçL'Ĺ (2009)ãĂĆ
- *The Quick Python Book*ïijÑä¡IJèĂĚ Vernon Cederïij Ñ Manning åĞžçL'Ĺ(2010)ãĂĆ
- *Python Programming for the Absolute Beginner*ïijÑçňňäÿL'çL'ĹïijÑä¡IJèĂĚ Michael Dawsonïij ÑCourse Technology PTR åĞžçL'Ĺ(2010).
- *Beginning Python: From Novice to Professional*ïijÑçňňäžÑçL'ĹïijÑ ä¡IJèĂĚ Magnus Lie HetâĂŘ landïij Ñ Apress åĞžçL'Ĺ(2008).
- *Programming in Python 3*ïijÑçňňäžÑçL'ĹïijÑä¡IJèĂĚ Mark Summerfieldïij ÑAddison-Wesley åĞžçL'Ĺ (2010).

énŸçžğäźęçś■

äÿŃéİćçŽĎè£ŹäžŻäźęçś■æŔŘä¿ŻäžĘæŻťåď'ŽénŸçžğçŽĎèŇČåŻťïijÑäź§åŇĚåŘńPython 3æŰźéİćçŽĎåĘĚåőźãĂĆ

- *Programming Python*ïijÑçňňåŻŻçL'Ĺ, by Mark Lutz, OâĂŹReilly & Associates åĞžçL'Ĺ(2010).
- *Python Essential Reference*ïijÑçňňåŻŻçL'ĹïijÑä¡IJèĂĚ David Beazley, Addison-Wesley åĞžçL'Ĺ(2009).
- *Core Python Applications Programming*ïijÑçňňäÿL'çL'ĹïijÑä¡IJèĂĚ Wesley Chun, Prentice Hall åĞžçL'Ĺ(2012).
- *The Python Standard Library by Example* ïijÑ ä¡IJèĂĚ Doug Hellmannïij ÑAddison-Wesley åĞžçL'Ĺ(2011).
- *Python 3 Object Oriented Programming*ïijÑä¡IJèĂĚ Dusty Phillips, Packt Publishing åĞžçL'Ĺ(2010).
- *Porting to Python 3*ïijÑ ä¡IJèĂĚ Lennart Regebroïij ÑCreateSpace åĞžçL'Ĺ(2011), http://python3porting.com.

åĚşäžŐèŕŚèĂĚ

åĚşäžŐèŕŚèĂĚ

- åğŞåŘ■ïijŽ çĘŁèČ¡
- å¿őä£ąïijŽ yidao620
- Emailïij Ž yidao620@gmail.com
- å■Žåőćïij Ž http://yidao620c.github.io/
- GitHubïij Ž https://github.com/yidao620c

Roadmap

2014/08/10 - 2014/08/31:

```
|    githubéąźçŻőæŘ■åżžïijŇreadthedocsæŰĞæąčçŤ§æĹŘãĂĆ
|    æŢt’äÿłéąźçŻőçŽĎæąĘæđűåőŇæĹŘ
```

2014/09/01 - 2014/10/31:

```
|    åL’■4çńăç£żèŕŚåőŇæĹŘ
```

2014/11/01 - 2015/01/31:

```
|    åL’■8çńăç£żèŕŚåőŇæĹŘ
```

2015/02/01 - 2015/03/31:

```
|    åL’■9çńăç£żèŕŚåőŇæĹŘ
```

2015/04/01 - 2015/05/31:

```
|    10çńăç£żèŕŚåőŇæĹŘ
```

2015/06/01 - 2015/06/30:

```
|    11çńăç£żèŕŚåőŇæĹŘ
```

2015/07/01 - 2015/07/31:

```
|    12çńăç£żèŕŚåőŇæĹŘ
```

2015/08/01 - 2015/08/31:

```
|    13çńăç£żèŕŚåőŇæĹŘ
```

2015/09/01 - 2015/11/30:

```
|    14çńăç£żèŕŚåőŇæĹŘ
```

2015/12/01 - 2015/12/20:

```
|    15çńăç£żèŕŚåőŇæĹŘ
```

2015/12/21 - 2015/12/31:

```
|    åŕźåĚĺéČÍç£żèŕŚè£ŻèąŇæăąåŕźäÿĂæňą
```

2016/01/01 - 2016/01/10:

```
|    åŕźåd’ŰåĚňåijĂåŔŚåÿČåőŇæŢt’çL’Ĺ1.
↪0ïijŇåŇĚæŃňè¡ňæ■ćåŘŐçŽĎPDFæŰĞäżű
```