FunctionalX.py Documentation

Release 0.0.1

Yuhang(Steven) Wang

July 05, 2016

Contents

Contents:

CHAPTER 1

lists

1.1 cart2()

FunctionalX.src.lists._cart2.**cart2**(*list1: list*, *list2: list*) → list

Cartesian product of two lists.

Parameters

- **list1** (*list*) – input list 1
- **list2** (*list*) – input list 2

Returns a new list contains all Cartesian products of the two lists.

Return type list

```
>>> cart2(['a','b'], [1,2])
[['a',1],['a',2],['b',1], ['b',2]]
```

1.2 cartn()

FunctionalX.src.lists._cartn.**cartn**(**all_lists: list*) → list

Cartesian product of n lists.

Parameters **all_lists (vararg)** (*list*) – variable number of lists

Returns a new list

Return type list

```
>>> cartn(['a','b'], [1,2], ['A','B'])
[['a', 1, 'A'], ['a', 1, 'B'], ['a', 2, 'A'], ['a', 2, 'B'], ['b', 1, 'A'], ['b', 1, 'B'], ['b',
```

1.3 cart2_append()

FunctionalX.src.lists._cart2_append.**cart2_append**(*list1*, *list2*)

Append items from list2 to list1 for all possible pairs.

Append an element from the second list to the each element from the first list therefore increase the number of elements in each item in the first list.

Parameters

- **list1** (*list*) – input list 1
- **list2** (*list*) – input list 2

Returns a new list.

Return type list

```
>>> cart_append([[1,2]], ['a','b'])
[[1,2,'a'], [1,2,'b']]
```

1.4 cartn_append()

FunctionalX.src.lists._cartn_append.**cartn_append**(**list_of_lists: list*) → list

Cartesian product of n lists.

The difference from *cartn* is that the first element of the input *list_of_lists* is itself a list of lists. The elements from other lists are appended (inserted) to each sub-list of the first list.

Parameters **list_of_lists (vararg)** (*list*) – variable number of lists

Returns a new list

Return type list

```
>>> cartn_append([['a','b'],['c','d']], [1,2], ['A','B'])
[['a', 'b', 1, 'A'], ['a', 'b', 1, 'B'], ['a', 'b', 2, 'A'], ['a', 'b', 2, 'B'], ['c', 'd', 1, '
```

1.5 dict2list()

FunctionalX.src.lists._dict2list.**dict2list**(*d: dict*) → list

Return an array given a dictionary"

Parameters **d** (*dict*) – input dictionary object

Returns a list for possible combinations of keys/values

Return type list [['a', 1, 'A'], ['a', 2, 'B'], ['b', 3, 'C'], ['b', 4, 'D']]

```
>>> dict2list({
    "a": {
        1: "A",
        2: "B"
    },
    "b": {
        3: "C",
        4: "D"
    }
    })
[['a', 1, 'A'], ['a', 2, 'B'], ['b', 3, 'C'], ['b', 4, 'D']]
```

1.6 head()

FunctionalX.src.lists._head.**head**(*list1: list*) → object

Return the head of a list.

If the input list is empty, then return *None*.

Parameters **`list1`** (*list*) – input list

Returns the first item

Return type object

```
>>> head([])
None
```

```
>>> head([1,2,3])
1
```

1.7 tail()

`FunctionalX.src.lists._tail.`**`tail`**(*list1: list*) → list

Return the tail of a list

Return the rest of the list (beyond the first element). If the input list is empty or has only one element, then return *[]*.

Parameters **`list1`** (*list*) – input list

Returns a new list

Return type list

```
>>> tail([])
[]
```

```
>>> tail([1])
[]
```

```
>>> tail([1,2,3])
[2,3]
```

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Index