Unix - Advanced II

(a bit of programming)

Libor Mořkovský, Václav Janoušek

https://ngs-course.readthedocs.io/en/praha-february-2019/

`awk`: Scripting in one line

- Simple programming language – very useful for reordering columns

```
echo "  3  5 7" | awk '{ print $3, $1, $2}'
```

`awk`: Scripting in one line

***Use Atom to write a longer code*

- Simple programming language

```
awk 'BEGIN{ ... }{ ... }END{ ... }' f1.txt > f2.txt
```

Do something before going through the data

Do something while going through the data => for each line/record separately

Do something at the very end

Exercise (FASTQ)

1. Extract IDs of a FASTQ file and count the number of reads

```
cd

< data/fastq/HRTMUOC01.RL12.00.fastq
awk '{ if( (NR + 3) % 4 == 0 ){ print $0 } }' |
wc -l
```

Exercise (FASTQ)

2. Make a file with read ID and read lengths in one line

```
cd

< data/fastq/HRTMUOC01.RL12.00.fastq
awk 'BEGIN{OFS="\t"}{
if(( NR + 3 ) % 4 == 0 ){
   id = $0
}else{
   if( (NR + 3) % 4 == 1 ){
      print id,length($0)
   }
}
}' | less
```

Exercise (FASTQ)

3. Get average read length

```
cd ~

< data/fastq/HRTMUOC01.RL12.00.fastq \
awk 'BEGIN{ OFS="\t"; l=0; n=0 }{
   if( ( NR + 3 ) % 4 == 1 ){
      l = l + length($0);
      n = n + 1;
   }
}END{
   print "Average read length:", l/n
}'
```

Functions in Shell

Functions enable to use a routine multiple times without necessity of writing the same code multiple times

```
greetings(){ echo hello, hello; }

# To call the function:
greetings
```

Passing Things to Functions from Outside

```
greetings(){ echo $1; }

# To call the function:
greetings hello
```

Functions: Exercise

Write a function which would remove white symbols at the beginning of line in the output of uniq -c

```
uniqt(){ uniq -c | sed -r 's/^ *([0-9]+) /\1\t/'; }
```

```
## Get number of variants per chromosome
< something.vcf grep -v '^#' | cut -f1 | uniqt
```

Shell scripts

The code is saved in executable file (.sh)

```
bash script_name.sh (arg1) (arg2) (argN)
```

Shell scripts

The code is saved in executable file (.sh)

```
nano script.sh
```

```
#!/bin/sh
echo $1
```

```
bash script.sh hello
```

Exercise (FASTQ)

Write a shell script filter_fastq.sh to filter out short sequences (set the minimum size allowed)

```
#!/bin/sh

FILE=$1
LENGTH=$2
OUT=$1-filtered

< $FILE awk -v l=$LENGTH '{
        if( (NR + 3) % 4 == 0 ){
          id=$0;
        }else if( (NR + 3) % 4 == 1 ){
          seq=$0;
        }else if( (NR + 3) % 4 == 2 ){
          q=$0;
        }else{
          if( length(seq) >= l ){
            print id"\n"seq"\n"q"\n+";
          }
        }
      }' > $OUT

echo File `basename $FILE` done
```

Run Program in Parallel

Use fully the functionality of high-throughput computing

```
parallel -j 5 'bash script.sh {} > {}.out' ::: {1..10}
```

Run filter_fastq.sh in Parallel

```
parallel -j 1 'bash filter_fastq.sh {} 80' ::: *.fastq
```

Coffeeee…