Correction Jgb53d - 2016-2017

Denis Puthier

2/3/2017

Commandes Unix

Pas de gros soucis ici. A priori on ne se déplace jamais de répertoire.

Réponses

```
1- cd
# ou cd ~
# ou cd /home/ritchie
# ou cd $HOME

2- ls

3- wget  http ://ritchie-user.fr/transcript.fasta.gz

4- gunzip transcript.fasta.gz

5- wc -l transcript.fasta

6- head -n 20 transcript.fasta
# head -n20 transcript.fasta
# head -20 transcript.fasta

7- tail -n 20 transcript.fasta
# tail -n20 transcript.fasta
# tail -20 transcript.fasta
8- grep ">" transcript.fasta
# ou grep "^>" transcript.fasta
# ou grep ">.*" transcript.fasta
# ou grep "^>.*" transcript.fasta

9- mkdir ~/result
# ou mkdir /home/ritchie/result
# ou mkdir $HOME/result
# ou mkdir result # il se trouve dans son home
# ou mkdir ./result # il se trouve dans son home

10- mv  ~/result  ~/results
# ou  mv  result  results # il se trouve dans son home
# ou  mv  ./result  ./results # il se trouve dans son home
# ou  mv  result  ./results # il se trouve dans son home
# ou  mv  ./result  results # il se trouve dans son home
# ou  mv  $HOME/result  $HOME/results # il se trouve dans son home
# ... et toutes les autres combinaisons

11- mv transcript.fasta results
```

```
# ou mv transcript.fasta results/
# ou mv ./transcript.fasta ./results
# ou mv ~/transcript.fasta ~/results
# ou mv ~/transcript.fasta ./results
# ou mv $HOME/transcript.fasta ./results
# ... et toutes les autres combinaisons.
# On peut éventuellement envisager des déplacements (cd ...).
```

Expressions régulières

NB: les expressions régulières ne comportant pas d'espaces, elles pourront ou pas, être entourées de simples ou doubles guillement. Les trois instructions suivantes seront considérés comme équivalentes:

- grep A file
- grep "A" file
- grep 'A' file

Réponses proposées

```
1- grep ">" transcript.fasta
# ou grep ">.*" transcript.fasta
# ou grep ".*>.*" transcript.fasta
# A noter que 'grep >  transcript.fasta' (i.e sans guillemet autour du chevron) ne fonctionne pas (bash

2- grep ATGCC transcript.fasta
# ou grep .*ATGCC.* transcript.fasta
# ou grep ATGCC.* transcript.fasta
# ou grep .*ATGCC transcript.fasta

3- grep atgcc transcript.fasta
# ou grep .*atgcc.* transcript.fasta
# ou grep atgcc.* transcript.fasta
# ou grep .*atgcc transcript.fasta

4- grep ATC.{2,3}ATG transcript.fasta
# ou grep .*ATC.{2,3}ATG.*  transcript.fasta
# ou grep ATC.{2,3}ATG.*  transcript.fasta
# ou grep .*ATC.{2,3}ATG  transcript.fasta
# ou grep ATC..{1,2}ATG  transcript.fasta
# ou grep .*ATC..{1,2}ATG.*  transcript.fasta
# ou grep ATC..{1,2}ATG.*  transcript.fasta
# ou grep .*ATC..{1,2}ATG  transcript.fasta


5- grep ^AGG transcript.fasta
# ou grep ^AGG.* transcript.fasta

6- grep AAA$ transcript.fasta
# ou grep .*AAA$ transcript.fasta

7- grep ^AGG.*AAA$ transcript.fasta
```

```
8- grep ^AGG.*GGG.*AAA$ transcript.fasta

9- grep ^[ATGC]+$ transcript.fasta

10- grep ^$ transcript.fasta

11- grep ^[^ATGC]+$ transcript.fasta
```

Mince alors !

```
file_handler = open("transcript.fasta")
n=0
for line in file_handler:
    if n == 0:
        print("transcript\texon_number\tstrand\tsequence")
    line = line.rstrip("\n")
    if line.startswith(">"):
        line = line.lstrip(">")
        fields = line.split("|")
        transcript = fields[0]
        strand = fields[2]
        exon_number = fields[1]
        header =  transcript + "\t" +  exon_number + "\t" + strand
    else:
        print(header + "\t" + line)
    n += 1
```

Où sont les commentaires ?

Pas de grosse difficulté ici. Notez la comprehension mais aussi la manière dont l'étudiant s'est approprié la terminologie ad hoc. Le programme permet de sélectionner les séquences fasta exoniques des transcripts comportant au moins 4 exons.

NB: noter la petite erreur du développeur qui a laissé trainer un print de débogage en ligne 15 (affichage du contenu du dictionnaire).

Sélection aléatoire

Pas si facile que ça en fait. Noter convenablement si les premiers éléments sont présents (lecture du fichier, création d'un tableau de transcrits, tirages aléatoires).

```
# -*- coding: utf-8 -*-

##############################################################
# Un programme permettant de lire le fichier "transcript.fasta"
# et de sélectionner au hasard 10 transcripts
# dont les séquences des exons correspondant seront
# imprimées au format FASTA.
# last modification: 03 Fev 2017
# @author: D. Puthier
```

```
############################################################

from random import randint

# Variable definition
NB_RANDOM = 2
transcript_list = set()
selected_transcripts = list()

# reading the file

file_handler = open("transcript.fasta", "r")

for line in file_handler:
    line = line.rstrip("\n")
    if line.startswith(">"):
        line = line.lstrip(">")
        fields = line.split("|")
        tx_name =fields[0]
        transcript_list.add(tx_name)


transcript_list = list(transcript_list)

for i in range(NB_RANDOM):
    pos = randint(0,len(transcript_list)-1)
    selected_transcripts += [transcript_list[pos]]
    transcript_list.remove(transcript_list[pos])

file_handler = open("transcript.fasta", "r")

for line in file_handler:

    line = line.rstrip("\n")
    if line.startswith(">"):
        line = line.lstrip(">")
        fields = line.split("|")
        tx_name =fields[0]
        to_print = False
        if tx_name in selected_transcripts:
            print(line)
            to_print = True

    else:
        if to_print:
            print(line)
            to_print = False
```