Polytech
Biotech III

Module : Bioinformatique et analyse de données.
Responsable : D. Puthier.
Date : Jeudi 28 Janvier.
Durée : 2 heures.
Calculatrices/portables : non autorisés.
Documents : non-autorisés.
Nombre de pages : 5

1. Vous répondrez aux questions qui suivent sur la feuille d'examen qui vous sera fournie.
2. Ecrivez lisiblement.
3. Respectez bien les consignes qui vous sont données dans chacune des questions.
4. N'oubliez pas de signer la feuille d'émargement après avoir rendu votre copie.

Le fichier *exons.txt*, dont les premières lignes sont affichées ci-dessous, sera utilisé dans certains des exercices proposés. Ce fichier contient des informations issues de la base de données *ensembl*. Il s'agit d'un fichier tabulé à six colonnes. **Chaque ligne contient les informations sur un exon** d'un transcrit humain.

- La colonne 1 contient le nom du chromosome portant le transcrit dont est issu l'exon.
- La colonne 2 contient la coordonnée de départ de l'exon.
- La colonne 3 contient la coordonnée de fin de l'exon.
- La colonne 4 contient l'identifiant du gène.
- La colonne 5 contient l'identifiant du transcrit associé à cet exon,
- La colonne 6 contient le brin (+ ou -).

```
chr1 11869 12227 ENSG00000223972 ENST00000456328 +
chr1 12613 12721 ENSG00000223972 ENST00000456328 +
chr1 12010 12057 ENSG00000223972 ENST00000450305 +
chr1 12179 12227 ENSG00000223972 ENST00000450305 +
chr1 12613 12697 ENSG00000223972 ENST00000450305 +
chr1 29534 29570 ENSG00000227232 ENST00000488147 -
chr1 17369 17436 ENSG00000278267 ENST00000619216 -
...
```

1 Premiers pas sous Unix (5 points)

Attention, vous devrez, toujours utiliser une instruction Unix pour répondre à la question posée.
Mr R. Stallman est un étudiant de la promo Polytech 2015 qui fait ses premiers pas dans l'environnement Unix/Linux et qui s'est vu confié une analyse bioinformatique. Le chemin de son répertoire d'utilisateur (*home*) est :

```
/etudiants/2016/stallman
```

R. Stallman s'est placé à la racine de l'arborescence, dans le répertoire **/tmp** (c'est donc le répertoire *courant*).

Proposez une instruction lui permettant :

1) De lister les fichiers et dossiers présents dans le répertoire courant.
2) De lister les fichiers et dossiers présents dans le répertoire courant en affichant les informations associées (propriétaire, groupe, droits,...).
3) De lister les fichiers et dossiers présents dans le répertoire courant y compris les fichiers cachés.
4) De lister sans se déplacer les fichiers et dossiers présents dans le répertoire "/".
5) De lister sans se déplacer les fichiers et dossiers présents dans le répertoire "/" (proposez une autre solution).
6) De lister, sans se déplacer, les fichiers présents dans son répertoire utilisateurs ("home").
7) De visualiser, avec la commande *less* le contenu du fichier "exons.txt" présent dans son répertoire utilisateur (*home*).
8) D'afficher les 10 premières lignes du fichier "exons.txt" présent dans son répertoire utilisateur.
9) D'afficher les 10 dernières lignes du fichier "exons.txt" présent dans son répertoire utilisateur.
10) De changer les droits du fichier "exons.txt" présent dans son répertoire utilisateur pour que les personnes de son groupe puissent lire et écrire dans ce fichier.
11) D'extraire la colonne 2 du fichier "exons.txt" présent dans son répertoire utilisateur.
12) De rechercher, dans ce même fichier, les lignes commençant par "ENST".
13) De rechercher, dans ce même fichier, les lignes commençant par "ENST" et finissant par un 2, suivi d'un chiffre quelconque et d'un 3.
14) De rechercher, dans ce même fichier, toutes les lignes contenant exactement trois caractères.

15) De rechercher les lignes vides.

16) De copier le fichier "exons.txt" depuis son répertoire utilisateur vers le répertoire courant.

17) De renommer le fichier "exons.txt" (répertoire courant) en "exons_1.txt".

2 Mince alors ! (6 points)

R. Stallman débute. Il a par erreur utilisé la commande gshuf qui a randomisé les lignes de son programme. Dommage... Ce programme assez rudimentaire permet de calculer la taille des cDNA et de n'imprimer que ceux dont la taille est supérieur à 200 nucléotides. Le code du programme est indiqué ci-dessous.

```
01     line = line.rstrip("\n")
02     if not tx in tx_len:
03     else:
04         print(tx + "\t" + str(tx_len[tx]))
05     cols = line.split("\t")
06     start = int(cols[1])
07     tx = cols[4]
08 for line in f:
09         tx_len[tx] = ex_size
10 for tx in tx_len:
11 f = open("exons.txt")
12     if tx_len[tx] > 200:
13     ex_size = end - start
14 tx_len = dict()
15         tx_len[tx] =  tx_len[tx] + ex_size
16     end = int(cols[2])
```

Question : réorganiser les lignes du programme pour que celui-ci puisse à nouveau faire ce pour quoi il a été écrit. Reportez le code réorganisé dans votre copie d'examen.

3 Où sont les commentaires ! (6 points)

Pour son premier stage, R. Stallman dois reprendre le code produit par un autre étudiants ayant quitté le laboratoire. Le fichier, dont le nom est peu explicite *script.py*, comporte quelque lignes de code non commentées. Cependant, on comprend vite que le script prend en entrée ce même fichier "exons.txt".

```
01 f = open("exons.txt")
02 tx_5p = dict()
03
04 for line in f:
05     line = line.rstrip("\n")
06     cols = line.split("\t")
07     tx = cols[4]
08     start = int(cols[1])
09     end = int(cols[2])
10     strand = cols[5]
11     if strand == "+":
12         if tx not in tx_5p:
13             tx_5p[tx] = start
14         else:
15             if start < tx_5p[tx]:
16                 tx_5p[tx] = start
17     else:
18         if tx not in tx_5p:
19             tx_5p[tx] = end
20         else:
21             if end > tx_5p[tx]:
22                 tx_5p[tx] = end
23 for tx in tx_5p:
24     print(tx + "\t" + str(tx_5p[tx]))
```

Questions 1 : Recopiez le code sur votre copie en le commentant abondamment (4 points).
Questions 2 : Qu'est censé faire ce programme ? Expliquez. (2 points)

4 Sélection aléatoire (3 points)

Ecrivez un programme permettant de lire le fichier "exons.txt" et d'imprimer pour chaque gène une ligne contenant : son identifiant et l'un de ses transcrits tiré au hasard.

1. Le pseudo-code permettant de décrire la stratégie (2 points)
2. Le code du programme écrit en Python ? (2 points)

Vous devez dans ce programme utiliser la fonction randint() du module random. Celle-ci permet de tirer, au hasard, un entier dans un interval [n, m] (exemple ci-dessous)

```
from random import randint
n=2
m=5
randint(n,m) # un entier au hasard parmi: 2,3,4,5
```