Langage de Programmation Evolué – BI

Mini-projet : **Mots anglais dans le protéome humain**

Délai de remise du projet : **la semaine du 08 décembre 2025**
Dates des mini-soutenances : **la semaine du 08 décembre 2025**
Mode de travail : **en binôme**

Enoncé du projet

L'objectif de ce premier projet est de découvrir si des mots anglais peuvent se retrouver dans les séquences du protéome humain, c'est-à-dire dans les séquences de l'ensemble des protéines humaines. Vous aurez à votre disposition :

- Le fichier « **english-common-words.txt** », qui contient les 3000 mots anglais les plus fréquents, à raison d'1 mot par ligne.

- Le fichier « **human-proteome.fasta** » qui contient le protéome humain sous la forme de séquences au format FASTA. Attention, ce fichier est assez gros. Ce fichier provient de la banque de données UniProt à partir de cette page : https://www.uniprot.org/proteomes/UP000005640.

L'objectif de ce premier projet est de découvrir si des mots anglais peuvent se retrouver dans les séquences du protéome humain, c'est-à-dire dans les séquences de l'ensemble des protéines humaines.

1. Des mots

On considère le fichier « **english-common-words.txt »**. Ce fichier contient les 3000 mots anglais les plus fréquents, à raison d'1 mot par ligne.

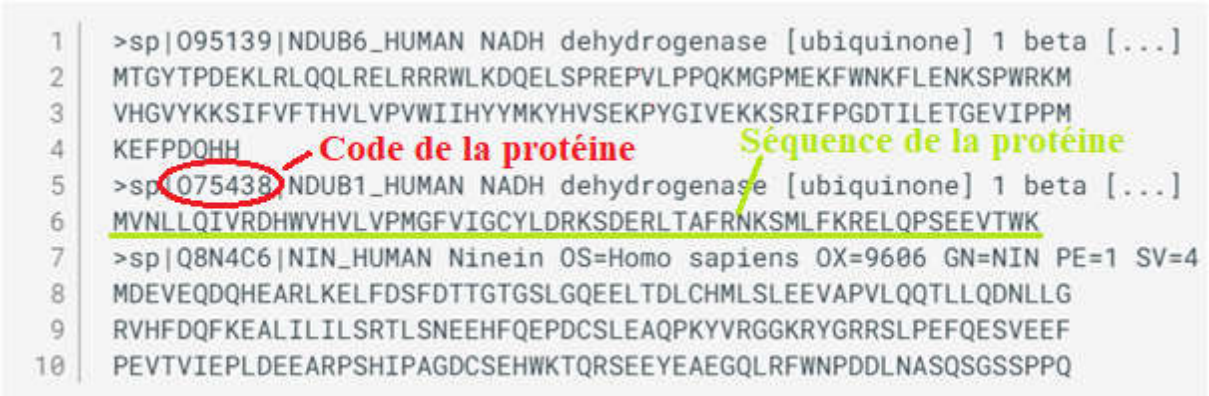
Créez un script **words_in_proteome.py** et écrivez la fonction **read_words()** qui va lire les mots contenus dans le fichier dont le nom est fourni en argument et renvoyer une liste contenant les mots convertis en majuscule et composés de 3 caractères ou plus

Dans le programme principal, affichez le nombre de mots sélectionnés.

2. Des protéines

On considère maintenant le fichier « **human-proteome.fasta** ».

Voici les premières lignes de ce fichier ([...] indique une coupure que nous avons faite) :



Toujours dans le script **words_in_proteome.py**, écrivez la fonction **read_sequences()** qui va lire le protéome dans le fichier dont le nom est fourni en argument. Cette fonction va renvoyer un dictionnaire dont les clefs sont les identifiants des protéines (par exemple, O95139, O75438, Q8N4C6) et dont les valeurs associées sont les séquences.

Dans le programme principal, affichez le nombre de séquences lues. À des fins de test, affichez également la séquence associée à la protéine O95139.

3. À la pêche aux mots

Écrivez maintenant la fonction **search_words_in_proteome()** qui prend en argument la liste de mots et le dictionnaire contenant les séquences des protéines et qui va compter le nombre de séquences dans lesquelles un mot est présent. Cette fonction renverra un dictionnaire dont les clefs sont les mots et les valeurs le nombre de séquences qui contiennent ces mots. La fonction affichera également le message suivant pour les mots trouvés dans le protéome :

```
ACCESS found in 1 sequences
ACID found in 38 sequences
ACT found in 805 sequences
[...]
```

Cette étape prend quelques minutes. Soyez patient.

4. Et le mot le plus fréquent est...

Pour terminer, écrivez maintenant la fonction **find_most_frequent_word()** qui prend en argument le dictionnaire renvoyé par la précédente fonction **search_words_in_proteome()** et qui affiche le mot trouvé dans le plus de protéines, ainsi que le nombre de séquences dans lesquelles il a été trouvé, sous la forme :

=> xxx found in yyy sequences

Quel est ce mot ?

Quel est le pourcentage des séquences du protéome qui contiennent ce mot ?

5. Pour être plus complet

Jusqu'à présent, nous avions déterminé, pour chaque mot, le nombre de séquences dans lesquelles il apparaissait. Nous pourrions aller plus loin et calculer aussi le nombre de fois que chaque mot apparaît dans les séquences.

Pour cela modifiez la fonction **search_words_in_proteome()** de façon à compter le nombre d'occurrences d'un mot dans les séquences.

Déterminez alors quel mot est le plus fréquent dans le protéome humain.

6. Interface graphique

Créez une interface graphique pour votre projet en utilisant le module « tkinter ».

Dans ce qui suit, vous trouverez une annexe pour apprendre à créer des GUI simples avec tkinter. Sinon, de nombreux guides en ligne sont disponibles. Exemple :

https://python.doctor/page-tkinter-interface-graphique-python-tutoriel

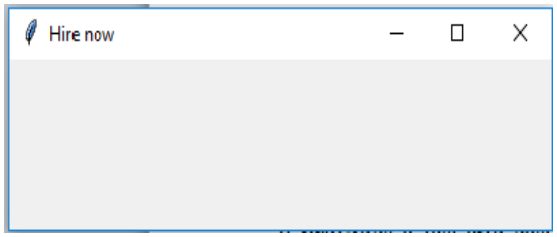
Annexe : création des interfaces graphiques en Python avec le module tkinter :

Python fournit diverses options pour développer des interfaces utilisateur graphiques (GUI) comme : Tkinter, wxPython, PyQt, JPython ...

Tkinter est le module GUI standard pour Python. Il fournit un moyen rapide et facile de créer des applications GUI. Tkinter fournit une puissante interface orientée objet au toolkit Tk GUI.

1. Créer une interface :

```
from tkinter import*
#constractor to instantiate and create the interface
main = Tk()
#add a title to the interface
main.title("Hire now")
# Code to add widgets will go here...
main.mainloop()
```

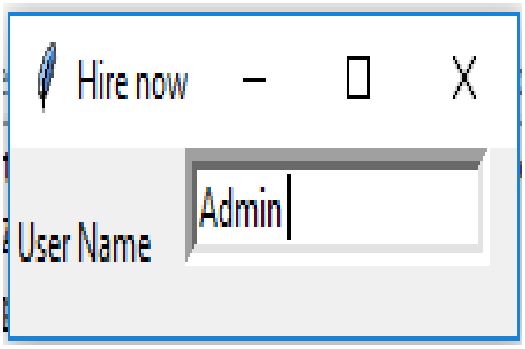


Le widget de saisie permet d'accepter des chaînes de texte saisies par l'utilisateur.

- Pour afficher plusieurs lignes de texte qui peuvent être modifiées, il faut utiliser le widget « Text ».
- Pour afficher une ou plusieurs lignes de texte qui ne peuvent pas être modifiées par l'utilisateur, il faut utiliser le widget « Label ».

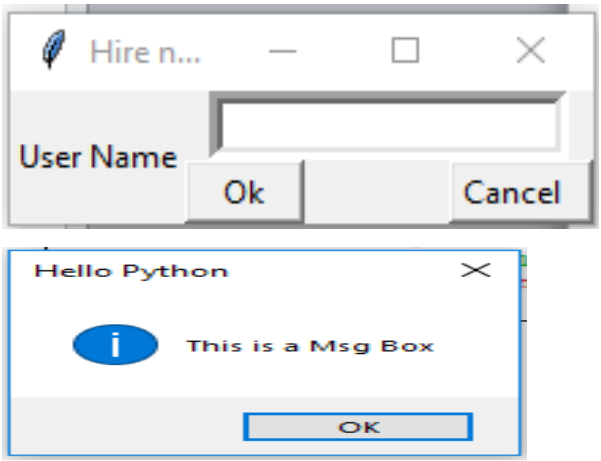
Syntaxe pour créer ce widget: w = Entry(master: name of parent window, option, ...)

```
from tkinter import*
#constractor to instantiate and create the interface
main = Tk()
#add a title to the interface
main.title("Hire now")
#add label on top
L1 = Label(main, text = "User Name")
#label will be showed at the left
L1.pack( side = LEFT)
#The size of the border around the indicator Default is 2 pixels
E1 = Entry(main, bd = 5)
#label will be showed at the left
E1.pack(side = RIGHT)
# Code to add widgets will go here...
main.mainloop()
```



2. Ajouter un bouton et un message box :

```
from tkinter import*
from tkinter import messagebox
#add Button
B = Button(main, text = "   Ok   ")
#add Button to the window
B.pack(side= LEFT)
Bl = Button(main, text = "Cancel  ")
#add Button to the window
Bl.pack(side= RIGHT)
#add and show the message box
msg = messagebox.showinfo( "Hello Python", "This is a Msg Box")
# Code to add widgets will go here..."""
main.mainloop()
```



Le tableau suivant décrit quelques options :

Option	**Description**
show	Affiche les caractères tapés par l'utilisateur lors d'une saisie. Pour la saisie d'un mot de passe, mettez show = "*".
selectforeground	Couleur de premier plan pour les éléments sélectionnés.
selectbackground	Couleur de fond pour les éléments sélectionnés.
justify	Contrôle la justification du texte : CENTER, LEFT, or RIGHT.
Bg	Couleur de fond du widget.
bd	Taille de la bordure. Par défaut, c'est 2 pixels.
command	Donner le nom d'une fonction à l'option *Command* du widget permettra l'appel automatique de la fonction lorsqu'un événement (click, appui sur une touche du clavier, etc.) se produit sur le widget.
Fg	Couleur de premier plan du widget.
Font	La police utilisée pour le texte.

Bon travail