Lecture / écriture de fichiers

Gestion de données

1 Fonctions et méthodes des chaînes de caractères

1.1 Les fondamentaux

Une chaîne de caractères est une séquence ordonnée et indexée de caractères, non modifiable. Le premier caractère a pour indice 0. Les chaînes de caractères sont définies entre apostrophes ou guillemets. Elles sont concaténables, multipliables par un entier, et on peut « extraire » la partie de notre choix grâce au slicing.

Si les chaînes de caractères s'y prêtent, on peut également les convertir en entier : int("666"), en réel : float("3.14") ou float("1e-4"), voire même en complexe : complex("1+4j").

La fonction len() qui appliquée à la chaîne de caractères « ch="babaLLe" » renvoie le nombre de caractères de ch (donc ici « len(ch) » est l'entier 7).

```
>>> ch="babaLLe"
>>> len(ch)
7
```

1.2 Méthodes utiles

Voici quelques méthodes utiles pour la gestion des chaînes de caractères.

```
>>> ch.upper()
'BABALLE'
>>> ch.lower()
'baballe'
>>> ch.capitalize()
'Baballe'
>>> ch.count("a")
2
>>> ch.find("aL")
3
>>> ch.split("a")
['b', 'b', 'LLe']
```

La méthode « split » est particulièrement utile.

1.3 Le slicing

Activité 1 **Rappel sur le slicing**

Exécutez le code ci-dessous pour vous remémorer la technique de slicing sur les chaînes de caractères. Le code est déjà écrit dans le fichier nommé `Activité_1_2.py`.

```
test="abcdefghijAB"
a=test[:] ; print(a) ; print(len(a))
b=test[:len(test)] ; print(b) ; print(len(b))
c=test[:len(test)-1] ; print(c) ; print(len(c))
d=test[:-1] ; print(d) ; print(len(d))
e=test[1:] ; print(e) ; print(len(e))
f=test[1:3] ; print(f) ; print(len(f))
```

1.4 Le changement de ligne

Le symbole pour changer de ligne est « \n ». Lorsqu'il est inclus dans une chaîne de caractère, ce symbole ne prend qu'**un seul** caractère.

Activité 2 **Le changement de ligne**

Exécutez le code ci-dessous pour assimiler l'effet de « \n », et de constater qu'il ne compte que pour un caractère. Le code est déjà écrit dans le fichier nommé `Activité_1_2.py`.

```
test2="abcdefghijAB\nccc"; print(test2) ; print(len(test2))
g=test2[:-1] ; print(g) ; print(len(g))
```

2 Ecrire et lire des chaînes de caractères dans des fichiers

2.1 Où suis-je ? Où vais-je ?

Lorsqu'on veut lire un fichier, il est fondamental de savoir exactement à quel endroit il se trouve sur le disque. On doit donc savoir préalablement à quel endroit python pointe sur le disque. Il n'est pas certain que Python pointe dans le dossier où se trouve le fichier que l'on désire lire.

Lorsqu'on a fabriqué un fichier que l'on veut sauvegarder, il est extrêmement important de savoir à quel endroit sur le disque il va être inscrit. Il faut donc être capable de spécifier le point d'écriture pour ce fichier ou ce document.

Il existe une suite de commandes qui permet de savoir où Python pointe sur le disque, et qui permet également de changer ce point de lecture ou d'écriture.

```
>>> import os

>>> os.getcwd() # on lit où Python pointe
'/Users/ericnoizet/Documents/Tests_Python'

>>> os.chdir('/Users/ericnoizet/Documents/Calculs') # on change le point de lecture/criture

>>> os.getcwd()
'/Users/ericnoizet/Documents/Calculs'
```

On peut également écrire ces lignes de commande directement dans le script. Il est important de noter que ces lignes de commande changent le point de lecture/écriture **pour l'ensemble des lignes de commande qui suivront**.

Il y a également une façon de spécifier le point de lecture ou d'écriture directement dans des lignes de commande sans que le point de lecture/écriture soit changé pour tout le script. Ce changement est effectué **une seule fois** lors de l'appel de lecture ou d'écriture de fichier. La syntaxe est vue par la suite sur des exemples.

2.2 Lecture / écriture

L'interaction avec un fichier annexe, présent dans le répertoire de travail se fait à l'aide d'un "objet-fichier" (noté simplement « f » par la suite), auquel on applique des méthodes d'ouverture, d'écriture ou de lecture, et de fermeture.

1ère étape : Ouverture du fichier objet : f =open('Nom_du_fichier.txt', paramètre d'ouverture)

Ici, le chemin vers le fichier est inclus dans 'Nom_du_fichier.txt'. Contrairement à ce qui était fait avant, le pointage vers le dossier où se trouve le dossier n'est pas modifié pour tout le script.

Il y a 3 possibilités pour le paramètre d'ouverture :
— 'r' (pour read) si le but est de lire le contenu du fichier
— 'w' (pour write) si le but est d'écrire dans un nouveau fichier (ou d'écraser le contenu d'un ancien fichier !)
— 'a' (pour append) si le but est d'ajouter du contenu dans un fichier (à la suite de ce qu'il contient déjà).

2ère étape : lecture OU écriture (pas les deux en même temps !)

Ecriture :

f.write("Ce que j'écris doit nécessairement être une chaîne de caractères !")

f.write("Si je ne précise pas de revenir à la ligne, tout s'écrit à la suite...\n")

f.write("C'est mieux comme ça, non ? \n")

Lecture :

ch = f.read() permet de récupérer l'intégralité du contenu du fichier dans une seule chaîne de caractères, nommée ch.

ou bien :

ch1 = f.readline() ; ch2 = f.readline() permet de récupérer le contenu du fichier ligne par ligne (la première dans la chaîne de caractères ch1, la seconde dans ch2, etc.)

ou bien :

liste_ch = f.readlines() permet de récupérer l'intégralité du contenu du fichier dans une liste de chaînes de caractères, chaque élément de la liste correspondant à une ligne du fichier.

3ère étape : Fermeture du fichier objet :

f .close()

N'oubliez jamais de refermer le fichier objet en fin de travail, sinon vous aurez bien du mal à y accéder par la suite...

Activité 3 **Lecture basique d'un fichier texte**

Exécutez le fichier `Poesie_lecture.py` et analysez bien ce qui est obtenu dans le script.
Comptez le nombre de fois où l'on trouve le mot « maître ».

Activité 4 **Écriture basique d'un fichier texte**

Exécutez le fichier `Ecriture_fichier.py` et analysez bien le résultat.

3 Applications

La majorité des logiciels qui traitent des séquences de nucléotides peuvent comprendre un format appelé FASTA. Le format FASTA est un fichier texte dépourvu de chiffres ou toute autre annotation qui est précédée par une ligne descriptive de texte. Un exemple de fichier, contenant comme information une suite de nucléotides A C G T, vous est proposé : `query_fasta.txt`. Ouvrez ce fichier avec n'importe quel logiciel capable d'ouvrir un fichier .txt, et examinez sa structure.

Dans la suite de cette section, je fais référence au fichier `Query_Fasta_lecture_et_analyse_a_completer.py`.

3.1 Nettoyage

Activité 5 **S'approprier un code**

Que fait le code ci-dessous ? (Adaptez le chemin de lecture de fichier)

```
fichier=open("/Users/ericnoizet/Documents/2ème année/Réforme 2014 2ème année/Python -
    Exemples/BCPST1_2021_2022/03 - Lecture_Ecriture/query_fasta.txt","r")

ch=fichier.readlines()

final=""
for i in range (len(ch)):
    if ch[i][0] != '>':
        final+=ch[i][:len(ch[i])-1]

print (final)
print (len(final))

fichier.close()
```

3.2 Pistage

Soit une chaîne de caractères 'ch' que l'on veut étudier. On cherche à savoir si un mot, qui est aussi une chaîne de caractères, est présente dans 'ch', et à quelle(s) position(s).

Par exemple, si 'ch' = "J'adore écrire des fonctions en Python", alors la recherche du mot 'on' doit renvoyer 20, 25 et 35.

Pour ce faire, on doit construire deux fonctions utiles.

Activité 6 **Ecriture d'une fonction**

nom : `mot_position_i`

arguments : texte, mot, i (str, str, int)

effet : Vérifie si mot a une occurrence dans texte en position i du texte. Doit renvoyer True si c'est le cas, False sinon.

Testez ensuite votre fonction sur cet exemple :

```
test="abcdabcd";mo="ab"
print (mot_position_i(test,mo,0));print(mot_position_i(test,mo,2));print(mot_position_i(test,mo,4))
```

Activité 7 **Ecriture d'une fonction**

nom : `cherche_occurence`

arguments : texte, mot (str, str)

effet : Donne la liste de toutes les occurrences de mot dans texte. L'occurence est un entier correspondant à la position de mot dans texte.

Testez ensuite votre fonction sur cet exemple :

```
chaine = "Ceci est un essai de phrase"
es=cherche_occurrences(chaine,"es")
print (es)
```

Chercher ensuite les occurrences pour 'ACG' dans la liste de nucléotides du fichier FASTA proposé. Combien y a-t-il d'occurrences au total ?