TP 10 - Écriture/lecture dans un fichier

➜ Lorsque l'on ouvre un fichier, on précise ce que l'on va faire (lire, écrire, créer, effacer, compléter) et on donne un nom au flot de données :

Ouvrir un fichier

```
f=open("fichier",mode='rt',encoding='utf8')
```

La variable f est le nom du flot de données. Le mode détermine les opérations qui seront effectuées ; par défaut, le mode est `'rt'`, *lecture de texte*.
Ici, l'encodage utilisé est le standard UTF8.

Mode	Description
`'r'`	Ouverture en lecture
`'t'`	Ouverture en mode texte
`'b'`	Ouverture en binaire (code des caractères)
`'w'`	Création/ouverture en écriture avec effacement du fichier existant
`'x'`	Création en écriture si le fichier n'existe pas
`'a'`	Création/ouverture en écriture positionnée après le contenu existant

➜ Les autres commandes sont :

Notation	Description
`f.write('texte')`	Ecrit une chaîne de caractères
`f.read()`	Renvoie le contenu du fichier sous la forme d'une chaîne de caractères
`f.readline()`	Renvoie la ligne en cours (terminée par \n)
`f.readlines()`	Renvoie la liste des lignes
`f.close()`	Ferme le flot

➜ Exemple d'écriture :

```
f=open('test.txt',mode='wt',encoding='utf8')
f.write("Ceci␣est␣un␣")
f.write("test␣!␣\n␣")
f.write("Bon␣courage␣pour␣la␣suite.")
f.close()
```

Cela donne un fichier *test.txt* contenant les deux lignes :

```
Ceci est un test !
Bon courage pour la suite.
```

Remarque : On rappelle que le caractère de retour à la ligne est \n.
Remarque : Il est possible de commander une boucle for sur les lignes d'un fichier.
L'exemple suivant renvoie le nombre de caractères de chaque ligne d'un fichier :

```
f=open('test.txt','rt',encoding='utf8')
for i in f:
    print(len(i))
f.close()
# reponse
19
26
```

Attention ! Veillez à ce que tous vos fichiers dans dans le même répertoire et que l'adresse de shell soit réinitialisée (Ctrl+Maj+E)

⇛ La méthode .strip() (ou .rstrip() ou .lstrip() permet de retirer les espaces et \n qui commencent ou terminent une chaîne de caractère. En particulier, le résultat des méthode méthodes readline() ou readlines() sont des chaine de caractères terminant par \n.

méthode .strip

```
-->c='Avec un retour a ligne !\n'
-->C=c.strip()
-->c
'Avec un retour a ligne !\n'
-->C
'Avec un retour a ligne !'
```

⇛ La méthode .split('c') permet d'éclater une chaine de caractères en fonction du caractère spécifié c. Elle retourne la liste des morceaux. Souvent il s'agit d'éclater en fonction d'une virgule ou d'un espace :

méthode .split('c')

```
--> t='Il fait beau!'
-->T=t.split(' ')
-->t
'Il fait beau!'
-->T
['Il', 'fait', 'beau!]
--> t='21,0.5,1'
-->t.split(',')
['21', '0.5', '1']
```

Rappel sur les listes définies par compréhension : obtention de la liste des longueurs des mots d'une liste

```
--> L=['aze','rt','yuio']
--> [len(e) for e in L]
[3,2,4]
```


Lycée Saint-Exupery – maffre.saintexupery@gmail.com – MPSI

Rappel sur la conversion de type :

```
--> a=12;
--> str(a) # en chaine de caracteres
'12'
--> float('2.1') # en reel
2.1
--> int('17') # en entier
17
--> eval('2.1'),eval('17') # type induit
(2.1, 17)
```

Lecture et sauvegarde des fichiers

Il est important d'avoir conscience qu'il y a un **répertoire de travail** (celui du shell) qui peut différer de celui du fichier contenant le script. Il convient de donner à Pyzo la position des fichier manipuler (en écriture ou en lecture).

⇛ Méthode 1 : Sous Pyzo, on peut redémarrer le Shells dans le répertoire d'un script (fichier .py enregistré) :
- Menu Exécuter/Démarrer le script : Ctrl+Mal+E

⇛ Méthode 2 : Utiliser les fonctions du module os

Notation	Explication
`import os`	Importe le module os
`os.getcwd()`	Donne le répertoire courant
`os.chdir('I:\Python')`	Change le répertoire courant
`os.mkdir(TP')`	Créer un répertoire TP dans le répertoire courant

⇛ Méthode 3 : Décrire l'adresse exacte du fichier à chaque appel (lecture ou écriture)

```
f=open('F:\\TP\\test.txt','wt',encoding='utf8')
```

Exercices

Exercice 1 Écrire un script qui créer un fichier de nom essai.txt qui contient le texte suivant :

```
1 : Mathematiques
2 : Physique
3 : Sciences de l'ingenieur
```

Attention ! Veiller à placer le fichier dans un répertoire TP préalablement défini.

Exercice 2 *Un poème sur π*
Dans le fichier poeme.txt (à récupérer dans le répertoire commun de la classe) il y a un poème dont la longueur de chaque mot donne une décimale de π. Notons qu'un mot de longueur 10 correspond à un 0.
Écrire une fonction pi(fichier) qui va ouvrir le document fichier et parcourir le texte afin de lister la longueur de chaque mot et d'en déduire l'écriture décimale de π.

Exercice 3 *Taille d'un fichier* Écrire une fonction format(f) qui considère un fichier texte, f, et retourne les trois nombres suivant :
- le nombre de caractères
- le nombre de lignes
- la longueur de la plus longue ligne

Exercice 4 *Tableau → fichier* Considérant un tableau de nombres de dimension 2 (de type array), donner une procédure sauvegarde_M(M,fichier) qui un fichier texte, f, contenant la matrice ligne par ligne, les nombres étant séparés par une virgule.

Matrice aléatoire

```
import numpy as np
import numpy.random as rd
l,c=5,4
M=rd.randint(-5,6,size=(l,c))
```

A compléter : exercice inc.py

```
def sauvegarde_matrice(M,fichier):
    ff=open(fichier,mode='wt',encoding='utf8')
    l,c=...   # taille de la matrice
    for i in ...   # parcours des lignes
        for j ... # parcours des colonnes
            ff.write(...)
        # Cas du dernier coeff de la ligne
        ff.write(...)
    ff.close()
```

Exercice 5 *Fichier → tableau*
Considérons un fichier texte contenant un matrice, écrire la fonction lire_matrice(f) qui retourne la matrice contenue dans le fichier texte f.

Pour terminer la séance

Exercice 6 *Retourner Titi* Écrire divers fonctions permettant de retourner, renverser, ... le Titi ou le noeud (que vous trouverez dans le répertoire transmis).