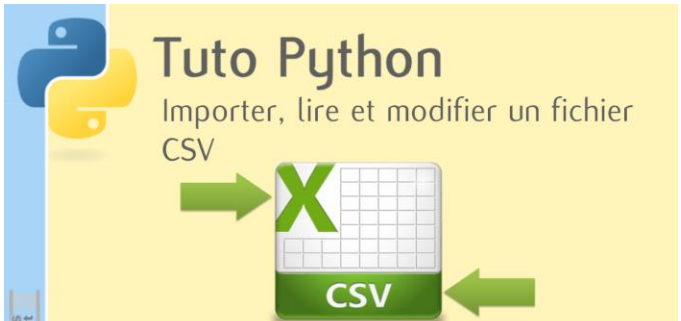
- Les fichiers -



Les fichiers

1. Introduction

Jusqu'à présent, les programmes que nous avons réalisés ne **traitaient qu'un très petit nombre de données**. Nous pouvions donc à chaque fois inclure ces données dans le corps du programme lui-même (par ex dans une liste). Cette façon de procéder devient cependant tout à fait inadéquate lorsque l'on souhaite traiter une quantité d'informations plus importantes.

2. Utilité des fichiers

Imaginons par exemple que nous voulions écrire un petit programme exerciseur qui fasse apparaître à l'écran des questions à choix multiple, avec traitement automatique des réponses de l'utilisateur. Comment mémoriser le texte des questions elles mêmes ?

Faisons appel à une liste:

```
liste = ["Qui a vaincu Napoléon à Waterloo ?",
         "Comment traduit-on 'informatique' en anglais ?",
         "Quelle est la formule chimique du méthane ?", ... etc ...]
```

On peut extraire n'importe quel élément de cette liste par son indice:

```
print(liste[2])        ===>  "Quelle est la formule chimique du méthane ?"
```

Mais plusieurs problèmes gênants se présentent:

* *la **lisibilité mauvaise** du programme si la liste est énorme,*
* *l'ajout de nouvelles questions, ou modifications imposeront de **rouvrir le code source**,*
* ***l'échange de données** avec d'autres programmes **impossible***

} séparer les données et les programmes en créant des **FICHIERS**

3. Les 2 familles de fichiers

Les fichiers sont tous écrits en **binaire**. Il est néanmoins possible de les séparer en 2 familles :
- *Les **fichiers binaires** qui nécessitent de connaître le format binaire d'écriture pour être lus,*
- *Les **fichiers texte** qui contiennent des caractères uniquement, et qui peuvent s'ouvrir sur un éditeur quelconque.*

Exemple

Les fichiers binaires :
— Images et documents (bmp, png, jpg, pdf, doc, etc)
— Son et vidéo (wav, mp3, mp4, etc)
— Exécutables (.exe)
— Archives compressées (zip, 7z, gz)
Les fichiers texte :
— Pages Web (html, css, etc)
— Fichier journal (log), Script shell (bat)
— Images vectorielles (svg)
— Programmes Python ou Scilab (py, sce)
— Les fichiers de données texte (txt, data, etc)
— Les fichiers texte formatés (xml)
Les fichiers texte compressés :
— Les fichiers bureautique (odt, ods, docx, xlsx)

	Les fichiers binaires	Les fichiers texte
Avantages	Moins volumineux Indépendants des standarts d'encodage des caractères dans les OS	Interprétables par l'homme Permettent des d'échanges plus simples entre logiciels Ne nécessitent généralement pas de bibliothèques
Inconvénients	Moins faciles à lire Nécessitent des bibliothèques pour les ouvrir	Plus volumineux Dépendants du format d'encodage des caractères

Les fichiers

4. Ouverture des fichiers

Si on souhaite ouvrir un fichier, on utilise la fonction « **open()** ». Elle prend en paramètre :

- *Chemin (absolu ou relatif) menant au fichier à ouvrir,*
- *Mode d'ouverture = chaine caractères*

Les principaux modes :

- « **r** » : ouverture en lecture (**Read**),
- « **w** » : ouverture en écriture (**Write**), *on écrase l'ancien fichier*
- « **a** » : ouverture en écriture en mode ajout (**Append**), *on écrase pas ancien fichier*

Ouverture d'un fichier : création d'un objet `file`

- Ouverture du fichier avec la fonction `open(nom, mode)`
  - `nom` : chaîne de caractères, nom du fichier
  - `mode` : chaîne de caractères, accès au fichier (`'r'` : *read*, `'w'` : *write*, `'a'` : *append*)

Les principales méthodes de la classe file sont les suivantes :

.read()	retourne tout le fichier comme un **str**
.readline()	retourne une ligne
.readlines()	retourne la liste de toutes les lignes du fichier
.close()	ferme le fichier

5. Les fichiers texte saisis dans Python

Un fichier texte est un fichier qui contient des **caractères « imprimables »** et des **espaces** organisés en lignes successives, ces lignes étant séparées les unes des autres par un caractère spécial non imprimable appelé ***marqueur de fin de ligne (/n).***

Lors des opérations de lecture, les lignes d'un fichier texte peuvent être extraites séparément les unes des autres. La méthode **readline(),** ne **lit qu'une seule ligne à la fois**. *(on considère que l'on dispose du fichier nommé fichiertexte dans le même répertoire que le script)*

```
f=open("fichiertexte.txt","r")
print(f.readline())
print(f.readline())
f.close()
```

```
>>>
Ceci est la ligne 1

Voici la ligne 2

>>>
```

La méthode **readlines()** transfère **toutes les lignes restantes** dans **une liste de chaînes** :

```
f=open("fichiertexte.txt","r")
print(f.readlines())
f.close()

print()
t=open("valeurs.txt","r")
print(t.readlines())
```

```
>>>
['Ceci est la ligne 1\n', 'Voici la ligne 2\n', 'Ceci est la ligne 3\n', 'Voici
la ligne 4\n']

['18 22 121 45 4 69\n', '78 7 11 12 33 \n', '12 12\n']
>>>
```

Les fichiers

La méthode **ligne.split()** retourne dans une **liste les mots** de la ligne (découpage en utilisant l'espace). Voici un exemple avec le fichier data1 suivant :

data1 - Bloc-notes

Fichier Edition Format Affichage Aide

```
# Temps[s]      Force [N]       Deplacement[m]
0.000000e+00    3.321566e+00    4.340203e-05
1.700000e-01    6.013489e+01    3.299359e-04
3.400000e-01    1.154852e+02    9.718301e-04
5.100000e-01    1.710066e+02    1.328353e-03
6.800000e-01    2.265946e+02    1.854428e-03
8.500000e-01    2.875566e+02    2.224948e-03
1.020000e+00    3.154837e+02    2.622302e-03
1.190000e+00    3.262366e+02    3.047014e-03
1.360000e+00    3.175871e+02    3.466145e-03
1.530000e+00    3.336673e+02    4.022022e-03
1.700000e+00    3.271517e+02    4.273263e-03
1.870000e+00    3.275063e+02    4.913872e-03
2.040000e+00    3.448111e+02    5.250066e-03
2.210000e+00    3.460045e+02    5.666204e-03
```

Voici le code pour découper la ligne 1 du fichier :

```
myfile=open("data1.txt","r")
data=myfile.readline()
print(data.split())
myfile.close()
```

```
>>>
['i»¿#', 'Temps[s]', 'Force', '[N]', 'Deplacement[m]']
>>>
```

6. Exercice synthèse

On vous donne un fichier data2.txt situé dans le répertoire courant, contenant un relevé de température en fonction du temps.

1°) Ouvrir ce fichier directement dans Python et regarder la structure de ce fichier.

data2.txt - C:\NICOLAS\CPGE\PTSI\INFORMATIQUE\INFO

File Edit Format Run Options Windows Help

```
# Time [s]      température [Deg. C]
0.000000e+00     3.321566e+01
1.700000e-01     6.013489e+01
3.400000e-01     1.154852e+02
5.100000e-01     1.710066e+02
6.800000e-01     2.265946e+02
# Création : Samedi 18 janvier 2014, 23:09:24
```

Comme l'interpréteur est un éditeur de texte, il ouvre data2.txt comme via blocnote et on **ne peut rien faire du fichier**…

2°) Ouvrir ce fichier en lecture via open() afin de l'afficher dans le shell. Vous créerez un script « lecture1temp » que vous enregistrerez dans le même répertoire que « data2.txt ».

```
# lecture d'un fichier texte et affichage

myFile = open("data2.txt", 'r')
t=myFile.read()
print(t)
```

```
>>>
# Time [s]      tempÃ©rature [Deg. C]
0.000000e+00     3.321566e+01
1.700000e-01     6.013489e+01
3.400000e-01     1.154852e+02
5.100000e-01     1.710066e+02
6.800000e-01     2.265946e+02
# CrÃ©ation : Samedi 18 janvier 2014, 23:09:24
```

3°) Ouvrir ce fichier en lecture via open() afin de l'afficher dans le shell dans une liste appelée « data ». Vous créerez un script « lecture2temp » que vous enregistrerez dans le même répertoire que « data2.txt ».

```
myFile = open("data2.txt", 'r')
data=myFile.readlines()
print(data)
```

```
>>>
['# Time [s]      tempÃ©rature [Deg. C]\n', '0.000000e+00\t3.321566e+01\n',
0000e-01\t6.013489e+01\n', '3.400000e-01\t1.154852e+02\n', '5.100000e-01\t1
66e+02\n', '6.800000e-01\t2.265946e+02\n', '# CrÃ©ation : Samedi 18 janvie
, 23:09:24\n']
```

Les fichiers

4°) Ouvrir ce fichier en lecture via open() afin de l'afficher dans le shell dans une liste appelée « data » composée de n liste issues des n ligne du fichier. Vous créerez un script « lecture3temp » que vous enregistrerez dans le même répertoire que « data2.txt ».

```
myFile=open("data2.txt","r")
data=[]
for lu in myFile:
    data.append(lu)
    print(lu.split())
myFile.close()
```

```
>>>
['#', 'Time', '[s]', 'tempÃ©rature', '[Deg.', 'C]']
['0.000000e+00', '3.321566e+01']
['1.700000e-01', '6.013489e+01']
['3.400000e-01', '1.154852e+02']
['5.100000e-01', '1.710066e+02']
['6.800000e-01', '2.265946e+02']
['#', 'CrÃ©ation', ':', 'Samedi', '18', 'janvier', '2014,', '23:09:24']
```

5°) Modifier votre code en n'affichant pas les lignes avec commentaires (#) puis enregistrer.

```
myFile = open("data2.txt", 'r')
data = []
for lu in myFile:
   if lu[0] != '#':
      data.append(lu)
      print(lu.split())
myFile.close()
```

```
>>>
['0.000000e+00', '3.321566e+01']
['1.700000e-01', '6.013489e+01']
['3.400000e-01', '1.154852e+02']
['5.100000e-01', '1.710066e+02']
['6.800000e-01', '2.265946e+02']
```

6°) Rajouter des instructions permettant de créer 2 listes : *t, temperature* dans lesquelles seront stockées les valeurs numériques du temps et de la température du fichier «data2.txt ». Puis, les afficher. Enregistrer votre code final.

```
myFile = open("data2.txt", 'r')
data = []
for lu in myFile:
   if lu[0] != '#':
      data.append(lu)
      print(lu.split())
myFile.close()
print()


t, temperature = [], []
for lu in data:
    (x1,x2)=lu.split()    #séparation par défaut les \t \n et espace...
    t.append(float(x1))
    temperature.append(float(x2))

print("t: ",t)
print("temp: ",temperature)

print(t[2])
print(t[2]+5)
```

```
>>>
['0.000000e+00', '3.321566e+01']
['1.700000e-01', '6.013489e+01']
['3.400000e-01', '1.154852e+02']
['5.100000e-01', '1.710066e+02']
['6.800000e-01', '2.265946e+02']

t : [0.0, 0.17, 0.34, 0.51, 0.68]
temp. : [33.21566, 60.13489, 115.4852, 171.0066, 226.5946]
```

Les fichiers

Ca y est, on extrait des valeurs d'un fichier et on pourra les traiter…

Exemple de récupération de données :

```
myFile = open("data2.txt", 'r')
data = []
for lu in myFile:
   if lu[0] != '#':
      data.append(lu)
myFile.close()
print()


t, temperature = [], []
for lu in data:
    (x1,x2)=lu.split() #séparation par défaut les \t \n et espace...
    t.append(float(x1))
    temperature.append(float(x2))

print("t: ",t)
print("temp: ",temperature)

print()

print("Au temps t=",t[2],"secondes, la temperature est de",temperature[2],"degres celcius")
```

```
t : [0.0, 0.17, 0.34, 0.51, 0.68]
temp. : [33.21566, 60.13489, 115.4852, 171.0066, 226.5946]

Au temps t= 0.34 secondes, la temperature est de 115.4852 degres celcius
```

7. L'instruction *.format()*

L'instruction **.format()** précédée d'accolades **{}** ou **\t** ou **\n** permet d'afficher le contenu de listes en colonnes et en clair.

```
liste_temps=["temps: ",0,1,2,3,4,5]
liste_vitesse=["vitesse: ",2.1,2.7,2.8,3.1,3.8,4.1]
print(liste_temps)
print(liste_vitesse)
print()

fo=open("resultats.txt","w")
n=len(liste_temps)
for i in range(n):
    ligne="{}\t{}\n".format(liste_temps[i],liste_vitesse[i])
    fo.write(ligne)
fo.close()
myFile= open("resultats.txt", 'r')
t2=myFile.read()
print(t2)
```

```
>>>
temps   vitesse
0       2.1
1       2.7
2       2.8
3       3.1
4       3.8
5       4.1
```

8. L'instruction *map(function, sequence)*

L'instruction ***map(function, sequence)*** permet d'appliquer la fonction (ex : float) à la sequence (ex : val)
Voici un fichier texte contenant n valeurs :

```
data3
Fichier  Edition  Format  Affichage  ?
0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114
0.0550 0.22 0.022 0.0444 0.07 0.1 056 0.89 0.058
0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114
0.0550 0.22 0.022 0.0444 0.07 0.1 056 0.89 0.058
0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114
0.0550 0.22 0.022 0.0444 0.07 0.1 056 0.89 0.058
0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114
0.0550 0.22 0.022 0.0444 0.07 0.1 056 0.89 0.058
0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114
0.0550 0.22 0.022 0.0444 0.07 0.1 056 0.89 0.058
0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114
0.0550 0.22 0.022 0.0444 0.07 0.1 056 0.89 0.058
0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114
0.0550 0.22 0.022 0.0444 0.07 0.1 056 0.89 0.058
0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114
0.0550 0.22 0.022 0.0444 0.07 0.1 056 0.89 0.058
0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114
0.0550 0.22 0.022 0.0444 0.07 0.1 056 0.89 0.058
```

On souhaite calculer la moyenne de ces valeurs (qui sont des string dans ce fichier texte !!). Pour cela on va ouvrir le fichier en créant une liste (readlines) puis créer une autre liste (data) contenant les valeurs en flottant des chiffres du fichier texte :

```python
myFile = open("data3.txt", 'r')
fichier=myFile.readlines()
print(fichier)
myFile.close()

print()

data = []
for val in fichier:
    data=data+list(map(float,val.split()))

print("les valeurs sont: ",data)

print()

som=0
moy=0
for x in data:
   som=som+x

moy=som/len(data)

print("la moyenne est:" ,moy)
```

```
>>>
['0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114\n', '0.0550 0.22 0.022 0.0
444 0.07 0.1 0.56 0.89 0.058\n', '0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0
.0114\n', '0.0550 0.22 0.022 0.0444 0.07 0.1 0.56 0.89 0.058\n', '0.01 0.02 0.06
 0.01 0 0.001 0.01 0.015 0.002 0.0114\n', '0.0550 0.22 0.022 0.0444 0.07 0.1 0.5
6 0.89 0.058\n', '0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114\n', '0.055
0 0.22 0.022 0.0444 0.07 0.1 0.56 0.89 0.058\n', '0.01 0.02 0.06 0.01 0 0.001 0.
01 0.015 0.002 0.0114\n', '0.0550 0.22 0.022 0.0444 0.07 0.1 0.56 0.89 0.058\n',
 '0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114\n', '0.0550 0.22 0.022 0.0
444 0.07 0.1 0.56 0.89 0.058\n', '0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0
.0114\n', '0.0550 0.22 0.022 0.0444 0.07 0.1 0.56 0.89 0.058\n', '0.01 0.02 0.06
 0.01 0 0.001 0.01 0.015 0.002 0.0114\n', '0.0550 0.22 0.022 0.0444 0.07 0.1 0.5
6 0.89 0.058\n', '0.01 0.02 0.06 0.01 0 0.001 0.01 0.015 0.002 0.0114\n', '0.055
0 0.22 0.022 0.0444 0.07 0.1 0.56 0.89 0.058']

les valeurs sont: [0.01, 0.02, 0.06, 0.01, 0.0, 0.001, 0.01, 0.015, 0.002, 0.011
4, 0.055, 0.22, 0.022, 0.0444, 0.07, 0.1, 0.56, 0.89, 0.058, 0.01, 0.02, 0.06, 0
.01, 0.0, 0.001, 0.01, 0.015, 0.002, 0.0114, 0.055, 0.22, 0.022, 0.0444, 0.07, 0
.1, 0.56, 0.89, 0.058, 0.01, 0.02, 0.06, 0.01, 0.0, 0.001, 0.01, 0.015, 0.002, 0
.0114, 0.055, 0.22, 0.022, 0.0444, 0.07, 0.1, 0.56, 0.89, 0.058, 0.01, 0.02, 0.0
6, 0.01, 0.0, 0.001, 0.01, 0.015, 0.002, 0.0114, 0.055, 0.22, 0.022, 0.0444, 0.0
7, 0.1, 0.56, 0.89, 0.058, 0.01, 0.02, 0.06, 0.01, 0.0, 0.001, 0.01, 0.015, 0.00
2, 0.0114, 0.055, 0.22, 0.022, 0.0444, 0.07, 0.1, 0.56, 0.89, 0.058, 0.01, 0.02,
 0.06, 0.01, 0.0, 0.001, 0.01, 0.015, 0.002, 0.0114, 0.055, 0.22, 0.022, 0.0444,
 0.07, 0.1, 0.56, 0.89, 0.058, 0.01, 0.02, 0.06, 0.01, 0.0, 0.001, 0.01, 0.015,
0.002, 0.0114, 0.055, 0.22, 0.022, 0.0444, 0.07, 0.1, 0.56, 0.89, 0.058, 0.01, 0
.02, 0.06, 0.01, 0.0, 0.001, 0.01, 0.015, 0.002, 0.0114, 0.055, 0.22, 0.022, 0.0
444, 0.07, 0.1, 0.56, 0.89, 0.058, 0.01, 0.02, 0.06, 0.01, 0.0, 0.001, 0.01, 0.0
15, 0.002, 0.0114, 0.055, 0.22, 0.022, 0.0444, 0.07, 0.1, 0.56, 0.89, 0.058]

la moyenne est: 0.11362105263157894
>>>
```