Chapitre 4: Modules/bibliothèques et lecture/écriture de fichiers

I- Bibliothèque, module, fonction W8XQMP

Une bibliothèque contient des modules contenants eux-mêmes plusieurs fonctions :

`import bibliothèque`	Permet l'importation de toutes les fonctions de tous les modules de `bibliothèque`. Pour appeler une fonction, il suffit d'écrire : `bibliothèque.nom_fonction`
`import bibliothèque as surnom`	Permet l'importation de toutes les fonctions de `bibliothèque`. Pour appeler une fonction, il suffit d'écrire : `surnom.nom_fonction`
`from bibliothèque import mon_module`	Permet l'importation de toutes les fonctions du module mon_module appartenant à `bibliothèque`. Pour appeler une fonction, il suffit d'écrire : `mon_module.nom_fonction`
`from bibliothèque import nom_fonction`	Permet l'importation de la fonction `nom_fonction` appartenant à `bibliothèque` Pour appeler une fonction, il suffit d'écrire : `nom_fonction`
`from bibliothèque.mon_module import nom_fonction`	Permet l'importation de la fonction `nom_fonction` appartenant au module `mon_module` de `bibliothèque` Pour appeler une fonction, il suffit d'écrire : `nom_fonction`

Il est possible de créer un module en créant un fichier .py contenant plusieurs fonctions. Si l'on souhaite utiliser la fonction *test* du module *mon_module.py* alors on écrit : `from mon_module import test`

II- Exemples de bibliothèques MO27AZ

a) La bibliothèque numpy

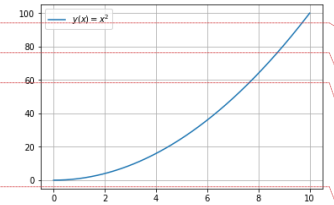
`import numpy as np`	`y=np.linspace(0,10,11)#11 pts entre 0 et 10 y=[ 0. 1. 2. 3. 4. 5. 6. 7. 8. 9. 10.]`

`from bibliothèque import mon_module`	`from numpy import random`
	`y=[random.randint(0,10) for i in range(10)]`
	`y=[8, 2, 6, 2, 7, 0, 5, 0, 6, 4]`

`from bibliothèque import nom_fonction`	`from numpy import linspace` `y=linspace(0,10,11)`

b) Le module pyplot de la bibliothèque matplotlib 8OD5QB

```
import matplotlib.pyplot as plt
import numpy as np
x=np.linspace(0,10,1000)
y=x**2
plt.plot(x,y,label="$y(x)=x^2$")
plt.legend()#affichage du label
plt.grid()#quadrillage
plt.show()#affichage de la courbe
```



III- Lecture et écriture de fichiers textes ZXLQ8M

a) Ouverture et fermeture d'un fichier

`fic=open("fichier_test.txt","w")`	Ouvre le fichier en mode écriture
`fic=open("fichier_test.txt","r")`	Ouvre le fichier en mode lecture
`fic=open("fichier_test.txt","a")`	Ouvre le fichier en mode ajout à la fin du fichier
`fic.close()`	Permet de fermer le fichier

b) Exemples

`fic=open("fichier_test.txt","w")`
`fic.write("j'écris dans le fichier \net je le ferme")`
`fic.close()`
	Le caractère \n force un retour à la ligne. Si le fichier est fermé puis à nouveau ouvert en écriture alors ce qui était écrit avant est perdu

`fic=open("fichier_test.txt","a")`
`fic.write("\nj'écris encore dans le fichier \net je le ferme")`
`fic.close()`
	Permet de rajouter sans perte

`fic2=open("mesures.csv","w")`
`a,b,c=1,2,3`
`fic2.write(str(a)+";"+str(b)+";"+str(c)+"\n")`
`fic2.write(str(a**2)+";"+str(b**2)+";"+str(c**2))`
`fic2.close()`
`fic2=open("mesures.csv","r")`
`tab=fic2.readlines()#lecture de toutes ls lignes`
`x,y,z=[],[],[]`
`for i in tab:`
`  ligne = i.split(";")`
`  x.append(float(ligne[0]))`
`  y.append(float(ligne[1]))`
`  z.append(float(ligne[2]))`
`fic2.close()`
	« ; »permet de séparé les données
	La méthode split coupe la ligne suivant le délimiteur « ; » et renvoie une liste.
	Ici : ["1","2","3"]
	["1","4","9"]

Commenté [AM6]: A noter l'obtention d'histogramme avec :
$plt.hist(x, y)$

Commenté [AM1]: En général, dans l'écriture d'un programme, il n'est pas question de réécrire ce qui existe déjà. Ce serait beaucoup trop long et improductif. Un programme en Python commence souvent par des lignes permettant d'importer des modules et des bibliothèques.

Commenté [AM2]: On parle aussi de library (traduction anglaise)

Commenté [AM3]: Lors de l'installation d'une nouvelle bibliothèque, son chemin d'accès est connu de manière automatique.

Commenté [AM4]: `from bibliothèque import *` permet d'importer toutes les fonctions sans aucun préfixe => dangereux car une même fonction peut appartenir à plusieurs modules avec des spécifications différentes

Commenté [AM7]: Dans le format CSV, les données sont séparées par un point-virgule :

```
fic2=open("mesures.csv","w")
a,b,c=1,2,3
fic2.write(str(a)+";"+str(b)+";"+str(c))
fic2.close()
```

Commenté [AM8]: Le fichier est créé si inexistant

Commenté [AM5]: Attention, cette importation ne fonctionne que si la module est enregistré dans le même dossier que le fichier.py de travail.
Si le module est enregistré dans un fichier quelconque alors il faut préciser le chemin d'accès :

```
from os import chdir
chdir("C:/Users/User/Documents/travail/TSI2_2021-2022/informatique/livre")#avec inversion des \ en /
import mon_module2
```

Commenté [AM9]: C pour comman (virgule)
D'autres séparateur de champ sont possible ; ou l'espace (tubalator)

Commenté [AM10]: Cette fonction renvoie une liste : chaque élément est une ligne.
La méthode read renvoie toutes les lignes sans séparation