

Institut Préparatoire aux Études d'Ingénieurs de Tunis



Leila Ben Othman

Deuxième année de classes préparatoires

SM - SP

Année universitaire: 2018 - 2019

Les fichiers en PYTHON

Plan

- ➊ Introduction
- ➋ Opérations sur les fichiers
- ➌ Module os
- ➍ Module pickle

Motivation

- Les données manipulées jusque ici sont introduites par l'utilisateur au moment de l'exécution.
- Le résultat est perdu dès que le programme s'arrête.

⇒ Les fichiers sont un excellent moyen pour la sauvegarde des données.

Contexte

Programmes qui récupèrent les données à partir d'un fichier et qui sauvegardent le résultat dans un fichier.

Fichier en PYTHON

- PYTHON dispose d'un objet de type **file** qui va permettre de sauvegarder et de récupérer les données à partir d'un fichier.
- Les opérations sur un fichier:
 - ouverture/création
 - fermeture
 - lecture
 - écriture

Ouverture/Création (1)

Syntaxe

```
f=open(nom_fichier, mode)
```

- *nom_fichier*: une chaîne de caractères indiquant le nom du fichier à ouvrir (exemple: 'test.txt')
- *mode*: le mode d'ouverture du fichier est donné sous forme de chaîne de caractères. Voici les principaux modes d'ouverture:
 - 'r' : ouverture en lecture seule (read). Si le fichier n'existe pas, une erreur **IOError** est levée.
 - 'w' : ouverture en écriture (write). Si le fichier existe, alors il sera écrasé, sinon il sera créé.
 - 'a' : ouverture en écriture en mode ajout (append). Si le fichier existe, l'écriture sera faite à la fin du fichier. Sinon, le fichier est créé.
- On peut parler de mode 'rb', 'wb' et 'ab' pour ouvrir le fichier en mode binaire. Nous verrons un peu plus loin ce mode particulier avec le module **pickle**.

Ouverture/Création (2)

- Pour ouvrir en lecture, le fichier 'test.txt' du répertoire courant:

```
f=open('test.txt','r')
```

- Si le fichier existe dans un autre emplacement, le nom du fichier doit être précédé par son chemin d'emplacement.

Exemple: `f=open('/home/leila/Bureau/test.txt','r')`

NB: Le répertoire courant peut-être changé: voir Section module **os**.

- **Attention:** ne pas confondre le nom physique du fichier 'test.txt' et le nom logique *f* utilisé par la suite dans le programme. *f* **fait référence au fichier 'test.txt': les caractères/lignes sont lus séquentiellement, c'est-à-dire les uns après les autres, sans possibilité de retour en arrière ni de saut en avant.**

Fermeture

Syntaxe

```
f.close()
```

NB: N'oubliez pas de fermer un fichier après l'avoir lu!

Lecture

Pour lire le contenu d'un fichier, on utilise l'une des méthodes suivantes:

- **f.read()**: permet de lire tout le contenu du fichier. Cette méthode retourne une chaîne de caractères représentant tout le contenu du fichier.
- **f.read(n)**: permet de lire n caractères.
- **f.readlines()**: permet de lire tout le contenu du fichier. Cette méthode retourne une liste composée de chaînes de caractères. Chaque chaîne représente une ligne du fichier.
- **f.readline()**: permet de lire une ligne à la fois. Cette méthode retourne une chaîne de caractère représentant la ligne courante lue.

Écriture

Pour écrire dans un fichier, on utilise l'une des méthodes suivantes:

- **f.write('texte')**: permet d'écrire la chaîne de caractères ('texte') passée en paramètre.
- **f.writelines(L)**: permet d'écrire une liste L, où chaque élément de L est une chaîne de caractères.

NB: pour constituer des lignes dans le fichier, on doit ajouter le symbole `"\n"` (symbole retour chariot).

Le module os

Par défaut, l'interprète PYTHON effectue les opérations d'ouverture/lecture/écriture à partir du répertoire courant. Il se peut que le fichier avec lequel vous travaillez soit placé dans un autre répertoire. Pour cela, on peut changer de répertoire de travail $\Rightarrow$ utiliser le module **os**.

- L'import du module os

```
import os as os
```

- Pour connaître le répertoire courant

```
os.getcwd()  
'/home/leila'
```

- Pour changer de répertoire courant

```
os.chdir('/home/leila/Bureau/Cours_LBO_2eme')
```

Le module pickle

Le module pickle permet d'enregistrer des objets de n'importe quel type en mode binaire. L'avantage est que ces objets peuvent être récupérés facilement par la suite.

- Pour importer le module pickle

```
import pickle as p
```

- Pour enregistrer un objet (ici un dictionnaire)

```
f=open('filename', 'wb')  
obj = {1 : 'a', 2 : 'b'}  
p.dump(obj, f)
```

- Pour récupérer l'objet

```
f=open('filename', 'rb')  
obj = p.load(f)
```

Exemple

```
import pickle as p
# creation d'objets de differents types
etudiants = {1: 'Ali', 2: 'Khaoula', 3: 'Nesrine'}
L=['Bonjour', 123, 'abc']
T=(L, 'soleil')
a=12
b=12.5
# enregistrement des objets
f = open('mes_donnees.bin', 'wb')
p.dump(etudiants, f)
p.dump(L, f)
p.dump(T, f)
p.dump(a, f)
p.dump(b, f)
f.close()
```

Exemple

```
# recuperation des objets
f = open( 'mes_donnees.bin' , 'rb' )
o1=p.load(f)
o2=p.load(f)
o3=p.load(f)
o4=p.load(f)
o5=p.load(f)
print(o1)
print(o2)
print(o3)
print(o4)
print(o5)
f.close()
```

Exemple

On peut récupérer les objets autrement:

```
# recuperation des objets
f = open( 'mes_donnees.bin' , 'rb' )
while True:
    try :
        obj = p.load(f)
        print( obj , type( obj ))
    except EOFError:
        break
f.close()
```