Partie II : Algorithmique et Programmation

SOMMAIRE

Chapitre : Les fichiers

Les fichiers de données

I. Introduction

Un fichier est l'unité élémentaire de stockage des données dans les systèmes d'information,

Un fichier stocke des informations sur un support physique (disque dur, clé USB, CD, DVD, carte mémoire SD...).

Il existe deux types de fichiers :

- Les **fichiers textes** : l'information est stockée sous forme de caractères lisibles par un éditeur de texte (principalement des lettres et des chiffres).
- Les **fichiers binaires** : l'information est stockée en binaire (une suite d'octets dont la valeur est comprise entre 0x00 et 0xFF).

En Python, comme d'autres langages, tous les périphériques (clavier, écran, imprimante,...), peuvent être considérés comme des fichiers.

Les principales manipulations sur un fichier sont :

- ⌘ L'ouverture du fichier
- ⌘ La lecture ou l'écriture d'un élément dans un fichier
- ⌘ La fermeture du fichier.

II. Ouverture et fermeture d'un fichier

1. ***Ouverture d'un fichier***

Tout fichier doit être ouvert avant de pouvoir accéder à son contenu en lecture et écriture. L'ouverture d'un fichier est réalisée par la fonction **open** selon la syntaxe suivante :

f = open(chemin_fichier, mode_ouverture) ;

La fonction **open** a pour argument deux chaînes de caractères, le chemin du fichier et le mode d'ouverture. Elle donne comme résultat un objet de type **<class _io.TextIOWrapper>** qui est utilisé dans les écritures, les lectures ultérieures.

Exemple: **f = open("C:/TP/essai.txt", "w")**

L'instruction de l'exemple ouvre (ou crée s'il n'existe pas) un fichier texte nommé essai.txt dans le répertoire TP du disque C.

Les modes de base sont :

- **"r"** (read : lecture) : lecture seulement ; le fichier doit exister.
- **"w"** (write : écriture) : écriture seulement. Si le fichier n'existe pas, il est créé. S'il existe, son ancien contenu est perdu.
- **"a"** (append : ajout) : si le fichier existe déjà, il sera étendu. sinon, il sera créé.

2. ***Fermeture d'un fichier***

Avant d'étudier les méthodes permettant d'accéder aux données d'un fichier ouvert, considérons la fonction qui permet de terminer la manipulation d'un fichier ouvert. Cette fonction réalise la fermeture du fichier ouvert. Elle a la syntaxe suivante :

f.close( )

III. Lecture et Ecriture dans un fichier

1. ***Lecture dans un fichier***

Une fois le fichier ouvert, l'objet **f** permet plusieurs fonctions d'accès à un fichier :

- ***Lire l'intégralité du fichier :***

Pour ce faire, on utilise la méthode read de la variable représentant le ficher. Elle renvoie l'intégralité du fichier :

```
 f = open("C:/TP/fichier.txt", "r")
contenu = f.read()
print(contenu)
f.close()
```

La méthode read renvoie tout le contenu du fichier, que l'on capture dans une chaîne de caractères. Si le fichier contient des retours à la ligne vous auriez dans votre variable **contenu** les signes \n traduisant un saut de ligne.

Maintenant que vous avez une chaîne, vous pouvez naturellement tout faire : la convertir, tout entière ou en partie, si c'est nécessaire, split la chaîne pour parcourir chaque ligne et les traiter… bref, tout est possible.

- ***Lire l'intégralité du fichier divisé par ligne dans une liste :***

Pour ce faire, on utilise la méthode readlines de la variable représentant le ficher. Elle renvoie la liste des lignes du fichier :

```
 f = open("C:/TP/fichier.txt", "r")
liste = f.readlines()
print(liste)
f.close()
```

- ***Lire un nombre donné de caractères :***

On peut aussi lire un nombre donné de caractère de notre fichier à l'aide de la méthode read(n) qui lit et renvoie n caractères :

```
 f = open("C:/TP/fichier.txt", "r")
ch = f.read(5)
print(ch)
f.close()
```

Nb : Après chaque appel de la méthode read le curseur de lecture se déplace par le nombre de caractères lu.

2. ***Ecriture dans un fichier***

Les opérations d'écriture sont les symétriques de celles de lecture. On retrouve la même classification et à peu près la même syntaxe.

- *Écrire une chaîne*

Pour écrire dans un fichier, on utilise la méthode write en lui passant en paramètre la chaîne à écrire dans le fichier. Elle renvoie le nombre de caractères qui ont été écrits. On n'est naturellement pas obligé de récupérer cette valeur, sauf si on en a besoin.

```
f = open("C:/TP/fichier.txt", "w")
f.write("Premier test d'écriture dans un fichier via Python")
f.close()
```

- *Écrire une liste de chaînes*

On peut aussi utiliser la méthode writelines qui prend en paramètre une liste de chaîne de caractères et les écrire dans le fichier.

```
f = open("C:/TP/fichier.txt", "w")
L=[“Bonjour\n”, “Youssef”]
f.writelines(L)
f.close()
```