

# La gestion des Fichiers en PYTHON

---

# Les fichiers de données

## ❖ *Notion de fichier*

Jusqu'à présent tous les programmes que nous avons écrits travaillaient sur des données qui étaient saisies à partir du **clavier** et **stockées en mémoire centrale (la RAM)**.

On peut cependant désirer **conserver des données** pour plus tard. Cette notion de conservation des données se concrétise sous la **forme de fichier**.

# Les fichiers de données

## ❖ *Définition d'un fichier:*

Un Fichier est un ensemble d'informations enregistrées sur un support physique (disquette, disque dur, etc.).

On distingue deux types de fichiers :

### ■ **Les fichiers textes :**

Les informations sont sous un format texte qui est lisible par n'importe quel éditeur de texte.

### ■ **Les fichiers binaires :**

Les informations ne sont lisibles que par le programme qui les a conçus.

---

# *Fichier texte*

---

# *Fichier texte*

## ❖ *Opérations sur les fichiers :*

Les opérations à appliquer sur un fichier :

- **Création**
- **Consultation(Lecture seulement)**
- **Modification(Lecture/Ecriture)**
- **Fermeture**

# Fichier texte

## ❖ *L'ouverture d'un fichier*

l'accès aux fichiers est assuré par l'intermédiaire d'un **objet-fichier** que l'on crée à l'aide de la fonction **open()**.

Cette fonction est définie ainsi par :

```
fichier1=open('nom_fichier', 'mode_ouverture')
```

```
#nom_fichier: représente le nom du fichier à ouvrir.
```

```
#mode_ouverture: représente le type d'ouverture(r,w,a)
```

# Fichier texte

## ❖ Mode d'ouverture d'un fichier :

Mode	Effet
'r'	Ouverture d'un fichier texte en <b>lecture seulement(read only)</b>
'w'	Ouverture d'un fichier texte en <b>écriture (write)</b> .si le fichier existe déjà, il sera écrasé. sinon sera crée
'a'	Ajouter des donnés (append), càd ouvrir <b>en écriture à la fin du fichier</b> . le fichier est crée s'il n'existe pas

# Fichier texte

## ❖ *La fermeture d'un fichier*

Lorsque les opérations sur un fichier sont terminées, il faut le fermer par un appel à la méthode **close()**.

```
#l'ouverture de fichier ayant comme objet fichier1
```

```
fichier1=open('nom_fichier1','w')
```

```
#la fermeture de fichier ayant comme objet fichier1
```

```
fichier1.close()
```



# Fichier texte

## ❖ *L'écriture dans un fichier*

Pour écrire dans un fichier nous utilisons la méthode **write()**.

**Exemple :**

```
#1'ouverture de fichier ayant comme objet fichier1
fichier1=open('nom_fichier1','w')
#Ajouter la chaine "langage Python" dans le fichier
    fichier1.write('Langage Python')
#1a fermeture de fichier ayant comme objet fichier1
fichier1.close()
```

# Fichier texte

## ❖ L'écriture dans un fichier :

### Remarque :

- L'argument de la méthode `write()` doit être une chaîne de caractères.
- Pour écrire des nombres avec la méthode `write()` n'oubliez donc pas de faire des transtypages si besoin avec la fonction `str(type)`. Exemple : `fichier.write(str(12.5))`

# Fichier texte

## ❖ *L'écriture dans un fichier*

La méthode `writelines(Liste)` permet d'écrire le contenu d'une liste dans un fichier.

### Exemple :

```
#1'ouverture de fichier ayant comme objet fichier1
fichier1=open('nom_fichier1','w')
#Ajouter la liste=['langage','Python'] dans le fichier
fichier1.writelines(['Langage','Python'])
#la fermeture de fichier ayant comme objet fichier1
fichier1.close()
```

# Fichier texte

## ❖ L'écriture à la fin d'un fichier

Pour écrire à la fin d'un fichier, on utilise la fonction `open ()` en mode ajout (append).

### Exemple :

```
#1'ouverture de fichier ayant comme objet fichier1
fichier1=open('nom_fichier1','a')
#Ajouter "Nouvelle ligne ajoutée avec append"
fichier1.write('nouvelle ligne ajoutée avec append')
#1a fermeture de fichier ayant comme objet fichier1
fichier1.close()
```

# *Fichier texte*

## ❖ *La lecture séquentielle dans un fichier :*

Il existe différentes façons de lire les données stockées dans un fichier:

- Lire l'intégralité d'un fichier
- Lire le fichier ligne par ligne
- Lire le fichier caractère par caractère

# Fichier texte

## ❖ *La lecture séquentielle dans un fichier :*

### ❖ *Lire l'intégralité d'un fichier :*

La méthode `read()` : Permet de lire l'intégralité du contenu de fichier et renvoyer une chaîne de caractères.

#### Exemple :

```
>>> f=open("texte1.txt", 'r')
>>> contenu=f.read()
>>> contenu
'Ceci est la ligne unCeci est la ligne deux'
>>> f.close()
```

```
>>> f.close()
```

# Fichier texte

## ❖ *La lecture séquentielle dans un fichier :*

### ❖ *Lire le fichier ligne par ligne :*

La méthode readline() : Permet de lire une ligne à fois et renvoyer une chaîne de caractères à partir de la position courante, si la fin est atteinte elle renvoie une chaîne vide.

# Fichier texte

❖ *La lecture séquentielle dans un fichier :*

❖ *Lire le fichier ligne par ligne :*

Exemple :

```
>>> f=open("textel.txt",'a')
>>> f.write("Ceci est la ligne un\n")
>>> f.write("Ceci est la ligne deux\n")
>>> f.write("Ceci est la ligne trois\n")
>>> f.close()
>>> f=open("textel.txt",'r')
>>> ligne1=contenu=f.readline()
>>> ligne1
'Ceci est la ligne un\n'
>>> ligne2=contenu=f.readline()
>>> ligne2
'Ceci est la ligne deux\n'
>>> ligne3=contenu=f.readline()
>>> ligne3
'Ceci est la ligne trois\n'
>>> f.close()
```



# Fichier texte

## ❖ *La lecture séquentielle dans un fichier :*

### ❖ *Lire le fichier ligne par ligne :*

La méthode `readlines()` : transfère toutes les lignes du fichier dans une liste de type chaîne de caractères.

### Exemple:

```
>>> f=open("texte1.txt",'r')
>>> contenu=f.readlines()
>>> contenu
['Ceci est la ligne un\n', 'Ceci est la ligne deux\n', 'Ceci est la ligne trois\n']
>>> f.close()
```

```
>>> f.close()
```

# Fichier texte

❖ *La lecture séquentielle dans un fichier :*

❖ *Lire le fichier caractère par caractère :*

La méthode read(n) : Permet de lire **n** caractères et retourne une chaîne vide "" si la fin du fichier est atteinte.

Exemple:

# Fichier texte

❖ *La lecture séquentielle dans un fichier :*

❖ *Lire le fichier caractère par caractère :*

Exemple:

```
>>> f=open("texte1.txt",'w')
>>> f.write("Ceci")
>>> f.close()
>>> f=open("texte1.txt",'r')
>>> caractere=f.read(1)
>>> while caractere!="":
>>>     print(caractere)
>>>     caractere=f.read(1)
C
e
c
i
>>> f.close()
```

# Fichier texte

## ❖ *Boucle de lecture séquentielle dans un fichier :*

Les objets-fichiers sont des **objets itérables**, c'est-à-dire qu'on peut en extraire les éléments un à un en utilisant la boucle **for** ou la boucle **while**.

### Exemple1:

```
>>> f=open("textel.txt",'r')
>>> for ligne in f :
        print(ligne)
```

```
Ceci est la ligne un
```

```
>>> f.close()
```

# Fichier texte

## ❖ Boucle de lecture séquentielle dans un fichier :

### Exemple 2 avec la méthode readline():

```
>>> f=open("texte1.txt",'r')
>>> contenu=f.readline()
>>> while contenu!="":
    print(contenu)
    contenu=f.readline()
```

Ceci est la ligne un

```
>>> f.close()
```

# Fichier texte

❖ *Boucle de lecture séquentielle dans un fichier :*

Exemple 3 avec la méthode `readlines()`:

```
>>> f=open("texte1.txt",'r')
>>> contenu=f.readlines()
>>> for ligne in contenu:
        print(ligne)
```

Ceci est la ligne un

```
>>> f.close()
```

# Exercice

## • Exercice 01:

1-Écrire un programme python qui permet de lire le nom, le prénom, la classe et la note de DS de cinq étudiants et les enregistrer dans un fichier texte nommé notes.txt dans le répertoire courant.

Nom	Prénom	Classe	DS
BAHIJ	CHAHER	T3	15.50
BALLOUM	EYA	M3	16.00
MOHAMED	HACHEM	T4	16.00
ATOUI	OUMAYMA	P3	14.50
AMAIDI	WAFI	M4	18.00

2-Écrire un programme python qui permet d'afficher le contenu de fichier notes.txt selon la forme ci-dessus.