

Algorithmique - programmation :

Les fichiers

Année Scolaire : 2020-2021

Introduction	2
Ouverture-Fermeture	3
La lecture du fichier	4
L'écriture dans un fichier	5
ATTENTION	6

Introduction

Nos programmes gardent en mémoire vive toutes les informations dont ils ont besoin, lorsqu'un programme se termine toutes les informations disparaissent et si l'on relance ce programme on repart de rien. Comment conserver des informations qui peuvent nous servir pour une prochaine fois, les nom, date et score des joueurs, la liste des réponses données par l'utilisateur sur un questionnaire, un texte ?

Toutes ces informations peuvent être conservées dans une structure que l'on appelle le fichier. En fait il existe 2 types de fichiers :

- Les fichiers binaires (image, son, traitement de textes, tableurs...) qui posent le problème du décodage des blocs de données qui le composent.
- Les fichiers texte : beaucoup plus lisibles, on l'ouvre avec un éditeur de textes et l'on trouve du texte et des retours à la ligne.

Nous allons nous intéresser ici uniquement à ces fichiers texte, qui vont nous permettre de sauvegarder les informations de nos programmes

Dans ce cours nous traiterons un exemple avec le fichier suivant que vous pouvez réaliser directement dans le bloc-notes de windows



```
1 Dominik
2 09/01/2021
3 2
```

Vous le sauvegarderez sous le nom de scores.txt

Ligne 1 : nom du joueur
Ligne 2 : date du score
Ligne 3 : nombre d'essais

L'utilisation d'un fichier se fait en 3 étapes :

- Ouverture du fichier
- Traitement (Lecture, Ecriture, Ajout, Suppression)
- Fermeture du fichier

Ouverture-Fermeture

L'**ouverture** se fait de la manière suivante :

```
fichier = open(nom, mode)
```

On crée une variable pour laquelle nous allons indiquer le nom du fichier que l'on veut lire (ici le fichier "scores.txt", et dans quel but nous avons ouvert ce fichier (lecture, écriture...)

```
fich = open("scores.txt", "r")
```

Attention : le nom du fichier doit contenir son nom avec l'extension, mais aussi ce que l'on appelle le chemin pour le trouver s'il ne se trouve pas dans le même répertoire que le programme

Les modes possibles :

	Effet
"r"	Ouvre le fichier en mode lecture (read)
"w"	Ouvre le fichier en mode écriture (write). Si le fichier n'existe pas il est créé, sinon il est écrasé et toutes les données qu'il contient sont supprimées
"a"	Ouvre le fichier en mode ajout (append). Si le fichier n'existe pas il est créé, sinon les ajouts se font à la fin du fichier existant

Fermeture : à la fin de l'utilisation du fichier pensez à le fermer avec l'instruction

```
fichier.close()
```

Dans notre cas :

```
fich.close()
```

Cela permet :

- De finaliser l'écriture dans le fichier (parfois python attend la fermeture pour écrire)
- De libérer le fichier pour les autres utilisateurs

Cette fermeture est obligatoire, si elle n'est pas faite on peut bloquer le fichier

La lecture du fichier

La lecture du fichier peut se faire de plusieurs façons :

- Lecture totale du fichier :
 - o Avec la méthode `fichier.read()` vous lisez tout le fichier d'un coup

```
fich=open("scores.txt","r")
texte=fich.read()
print(texte)
fich.close()
```

```
>>> %Run fichiers.py
Dominik
10/01/2021
2
```

- o Une deuxième méthode de lecture du fichier avec une boucle for :

```
fich=open("scores.txt","r")
i=1
for ligne in fich:
    print("ligne",i,ligne, end="")
    i+=1
fich.close()
```

```
>>> %Run fichiers.py
Ligne 1 : Dominik
Ligne 2 : 10/01/2021
Ligne 3 : 2
```

- Si l'on veut lire le fichier ligne par ligne, on utilise alors la méthode :

`fichier.readline()`

Qui permet de lire une ligne, si vous renouvelez la méthode dans le même programme, avant la fermeture du fichier, vous lirez la 2ème ligne, en effet le pointeur de lecture se positionne sur la ligne suivante

```
fich = open("scores.txt","r")
l = fich.readline()
print(l)
l = fich.readline()
print(l)
fich.close()
```

```
>>> %Run fichiers.py
Dominik
10/01/2021
```

Remarque : il existe entre chaque ligne du fichier une ligne supplémentaire. Ceci est normal la fonction `print()` va à la ligne et dans le fichier au bout de chaque ligne nous avons un caractère de passage à la ligne, qui est lu par python et interprété comme un saut de ligne, donc au total 2 passages à la ligne. Pour éviter cela vous pouvez retirer cet espace avec la fonction `rstrip()`

```
fich = open("scores.txt","r")
l = fich.readline()
l = l.rstrip()
print(l)
l = fich.readline()
print(l)
fich.close()
```

```
>>> %Run fichiers.py
Dominik
10/01/2021
```

- Une 3^{ème} méthode pour lire un fichier :

`fichier.readlines()`

Cette méthode enregistre le fichier dans sa totalité mais dans une liste, chaque ligne devient un élément de la liste, ce qui permet un traitement beaucoup plus simple de ce fichier.

```
fich=open("scores.txt","r")
l= fich.readlines()
print(l)
fich.close()
```

```
>>> %Run fichiers.py
[ 'Dominik\n', '10/01/2021\n', '2\n' ]
```

Remarque : chaque élément possède en dernière position le signe "\n" qui est le signe du retour à la ligne.

L'écriture dans un fichier

Que ce soit en mode "w" ou en mode "a" vous pouvez écrire dans un fichier avec la méthode :

`fichier.write()`

Attention, si vous voulez un passage à la ligne dans le fichier après chaque élément il vous faudra rajouter dans votre écriture le signe du passage à la ligne "\n".

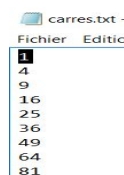
```
fich=open("scores.txt","a")
fich.write("Pierre\n")
fich.write("10/01/2021\n")
fich.write("4\n")
fich.close()
```



```
Dominik
09/01/2021
2
Pierre
10/01/2021
4
```

Voici un programme qui vous écrit les 10 premiers carrés et les enregistre dans le fichier "carres.txt"

```
fich=open("carres.txt","w")
for i in range (1,10):
    fich.write(str(i**2)+"\n")
fich.close()
```



```
carres.txt -
Fichier Editio
1
4
9
16
25
36
49
64
81
```

Remarque : la présence de la fonction `str()` avant le calcul du carré est obligatoire pour transformer le numérique en texte, nous écrivons dans un fichier texte et il est impossible d'y écrire des nombres.

ATTENTION

Cet écriture directe de lecture ou d'écriture dans les fichiers est à proscrire :

Lors du parcours du fichier, si une erreur se produit et que le fichier n'est pas fermé, il devient inutilisable dans la suite du programme. Il est donc préférable d'utiliser la fonction `open()` avec le mot clé `with` qui s'occupera de fermer le fichier en cas d'exception. Dans ce cas, il n'est plus nécessaire de fermer explicitement le fichier.

```
with open("scores.txt","r") as fich:
    for ligne in fich:
        ligne = ligne.rstrip()
        print(ligne)
```

```
with open("scores.txt","a") as fich:
    fich.write("Toto\n")
    fich.write("10/01/2021\n")
    fich.write("4\n")
```

A vous de jouer

Exercice n°1 : après avoir créé le fichier scores.txt (voir page 1), réaliser un programme qui lit ce fichier et affiche "Bonjour Dominik, le 09/01/2021, vous avez trouvé le bon nombre en 2 essais" Affichage sur une seule ligne bien sûr.

Exercice n°2 : réaliser un programme qui donne le nombre de mots dans le fichier "mots.txt"

Exercice n°3 : réaliser un programme qui lit tous les mots de 7 lettres du fichier "mots.txt" et les enregistre dans un fichier "mots7.txt"

Exercice n°4 : réalisez un programme affichant un mot pris au hasard dans le fichier "mots.txt"