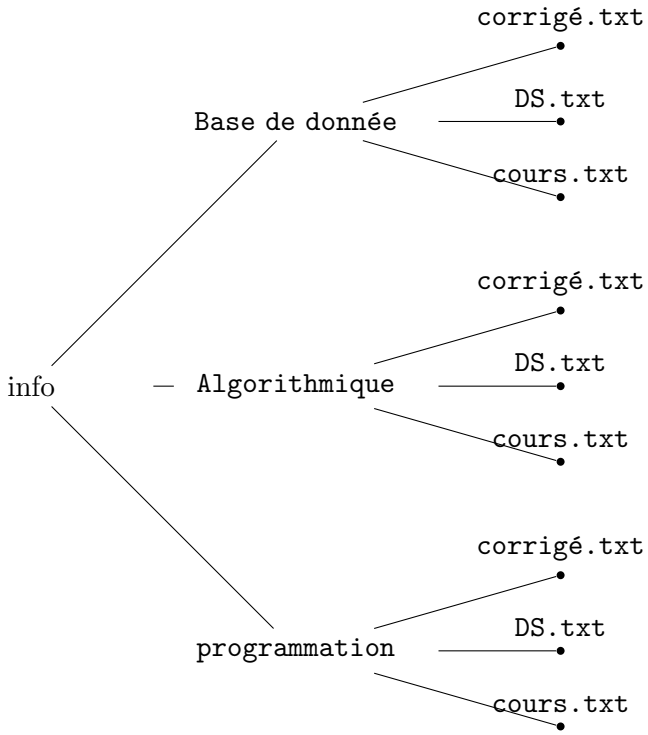
Écriture dans un fichier

Table des matières

Les fichiers permettent de sauvegarder des données de manière pérenne contrairement aux variables dont les valeurs disparaissent à la fin de l'exécution d'un programme. Ce chapitre montre comment ouvrir des fichiers en lecture ou en écriture.

I Répertoire

Les fichiers sont rangés dans l'ordinateur de manière arborescente. Par exemple sous Windows l'adresse (= la localisation) d'un fichier ressemble à :
`C:/users/moraux/travail/info/programmation/cours.txt`
Le fichier `cours.txt` se trouve dans le répertoire `programmation`, avec par exemple le fichier `corrigé.txt`. Le répertoire `programmation` est lui même dans le répertoire `info`, avec les répertoires `algorithmique` et `base de données`.



Lorsque vous travaillez en Python, vous êtes dans un répertoire.
Pour savoir où vous êtes en train de travailler, vous pouvez taper dans le shell la commande
`>>> cd`
Pour afficher les fichiers accessibles depuis ce répertoire, vous pouvez taper dans le shell la commande
`>>> ls`
Tous les fichiers affichés sont modifiables par Python. Dans ce répertoire de travail, vous pouvez aussi créer des fichiers.

Exercice 1

Taper dans le shell les commandes `>>> ls`

```
>>> open("test.txt","w")
>>> ls
```

Que se passe-t-il ? Ouvrir ce fichier à la main.

Remarque 2

On peut naviger dans les répertoires grâce aux commandes
`>>> cd ..` qui permet de revenir au répertoire précédent
`>>> cd nom_repertoire` qui permet d'accéder au sous-répertoire de nom `nom_repertoire`.
Par exemple, si vous êtes en train de travailler dans le répertoire `Base de donnée` et que vous voulez aller travailler dans le répertoire `programmation`, vous taperez sucessivement les commandes `cd ..` (à ce moment, vous êtes dans le répertoire `info`) puis `cd progrmmation`.

II Ouvrir et fermer un fichier

Pour ouvrir un fichier avec Python, on utilise la fonction `open`. Cette fonction prend comme premier argument le nom du fichier que l'on veut ouvrir sous la forme d'une chaine de caractère. Le deuxième argument est une chaine de caractère qui désigne le mode d'ouverture. On peut distinguer les modes suivants :

- "r" : lecture (≪ read ≫)
- "w" : écriture (≪ write ≫)
- "a" : écriture à partir de la fin du document (≪ append ≫)

Il ne faut pas oublier de fermer un fichier lorsqu'on a fini de l'utiliser. Cela est particulièrement important dans le cas où une autre application en aurait besoin. On utilise pour cela la méthode `close`.

Exemple 3

```
>>> f = open("essai.txt","w")
>>> type(f)
<class '_io.TextIOWrapper'>
>>> f.close()
```

Remarque 4

Le mode écriture "w" permet de créer un fichier s'il n'existe pas déjà, contrairement aux modes "r" et "a".

Exercice 5

Taper les commandes suivantes et expliquer le résultat :

```
>>> f = open("test1.txt","w")
>>> f = open("test1.txt","r")
>>> f.close()
>>> f = open("test2.txt","r")
```

III Lire et écrire dans un fichier

Pour écrire dans un fichier, on utilise la méthode `write` du fichier :

Exercice 6

Taper les commandes suivantes :

```
>>> f = open("essai.txt","w")
>>> f.write("Bonjour tout le monde !")
>>> f.write("Comment ça va ?\n")
>>> f.write('-"Ça va merci"\n ')
>>> f.close()
```

Aller maintenant ouvrir à la main le fichier essai.txt du répertoire de travail actuel.

Pour lire dans un fichier, on utilise les méthodes `read` et `readline` du fichier. La méthode `read` renvoie une chaine de caractère avec le contenu de tout le fichier. La méthode `readline` lit une ligne entière du fichier et place le curseur sur la ligne suivante.

Exercice 7

Taper les commandes suivantes et expliquer :

```
>>> f = open("essai.txt","r")
>>>  lect = f.read()
>>> lect
>>> g = f.readline()
>>> type(g)
>>> g
>>> f.readline()
>>> f.readline()
>>> f.close()
```

Pour lire toutes les lignes d'un fichier, on peut utiliser une boucle `for` puis lire les lignes une à une, ou utiliser la méthode `readlines` qui stocke toutes les lignes dans une liste.

Exercice 8

Taper les commandes suivantes et expliquer :

```
>>> f = open("es
>>>  listechar = f.readlines()
>>> listechar
>>> f.close()
>>> f = open("essai.txt","r")
>>> for laligne in f : print(laligne)
>>> f.close()
>>> f = open("essai.txt","r")
>>> for laligne in f : print(laligne)
>>>  listechar = f.readlines()
>>> listechar
>>> f.close()
```

IV Exercices

Exercice 9

1. Depuis mon site, enregistrer le fichier `poeme.txt` dans le répertoire de travail actuel.
2. Ouvrir, avec python, le fichier `poeme.txt` en mode lecture et l'affecter à la variable `poeme`.
3. Ajouter tout de suite l'instruction de fermeture du fichier, quelques lignes plus bas, pour ne pas l'oublier.
4. Entre l'instruction d'ouverture du fichier et celle de fermeture, créer une liste `Liste` contenant les lignes du fichier `poeme.txt`. En déduire le nombre de lignes du fichier `poeme.txt` (on affichera le résultat à l'aide d'un print).
5. Afficher (avec un print) les lignes 1 et 11 (uniquement).

Exercice 10

1. Créer, avec python, un fichier appelé `table_de_7.txt`. Dans ce fichier, écrire 30 lignes de texte de la forme : k fois 7 = k*7, où k varie entre 1 et 30 (exemple : 3 fois 7 = 21). Attention : la dernière ligne ne doit pas contenir de retour chariot `\n` (sinon, le fichier contiendra au total 31 lignes.) Ne pas oublier de fermer tout fichier ouvert.
2. Ouvrir (à la main, sans python) le fichier pour vérifier le contenu.
3. Ajouter la ligne suivante de la table de multiplication : 31×7, sur le même modèle que précédemment. Ne pas oublier pas d'aller à la ligne avant.
4. Ouvrir (directement, sans python) le fichier pour vérifier son contenu.
5. Recommencer ce travail, mais finalement écrire la table des 6 sous le même format, et dans le même fichier.
6. Ouvrir (directement, sans python) le fichier pour vérifier son contenu.

Exercice 11

Écrire une fonction `capslock` qui prend en argument une chaine de caractères contenant le nom d'un fichier de type texte se situant dans le répertoire de travail. Le programme imprime les lignes du texte en majuscules.

Exercice 12

Écrire une fonction `copier_coller` qui prend en argument deux chaines de caractères contenant le nom de deux fichiers de type texte, dont au moins un se situe dans le répertoire de travail. Le programme copie toutes les lignes du premier fichier dans le deuxième fichier. À la fin de l'exécution de la fonction, les deux fichiers sont identiques. Tester cette fonction avec le fichier `poeme.txt`