Informatique PCSI
Prérequis TP 3 : fichiers

Lecture et écriture de fichiers textes

Ouverture et fermeture d'un fichier

La fonction `open` prend deux paramètres, le nom du fichier et le mode d'ouverture : 'w' pour le mode "écriture", 'r' pour le mode "lecture" et 'a' pour le mode "ajout", (write, read, append).

La syntaxe est :

- `fic = open('fichier', 'w')` pour écrire ;
- `fic = open('fichier', 'r')` pour lire ;
- `fic = open('fichier', 'a')` pour ajouter du texte à la fin du fichier.

Il est essentiel de fermer un fichier qui a été ouvert. L'instruction est : `fic.close()`.

Écriture dans un fichier

La méthode `write` prend en paramètre une chaîne de caractères (type str), par exemple :
`fic.write("J'écris dans un fichier")`.

Pour écrire sur plusieurs lignes, on utilise le caractère '\n' qui force un retour à la ligne. Par exemple :
`fic.write("J'écris dans un fichier\nA la fin, je le ferme.")`,
ou en plus lisible
`fic.write("J'écris dans un fichier" + "\n" + "A la fin, je le ferme.")`.

Si le fichier est fermé puis à nouveau ouvert en écriture, c'est un nouveau fichier qui est écrit. Ce qui était écrit auparavant est perdu. Pour écrire à nouveau dans un fichier déjà fermé, on l'ouvre en mode "ajout". Le texte est écrit à la suite du précédent.

```
fic = open('fichier.txt', 'w')
fic.write("J'écris dans un fichier\nA la fin, je le ferme.")
fic.close()

fic = open('fichier.txt', 'a')
fic.write("Je continue.")
fic.close()
```

L'extension "txt" peut être remplacée par "dat" ou "csv". Le fichier écrit peut être ouvert dans un éditeur de texte. Par défaut, il est créé dans le dossier où est enregistré le fichier Python.

Si les données à écrire sont de type numérique, il faut les convertir au préalable en type str. Dans le format CSV, les données sont séparées par un point-virgule (une virgule chez les anglophones).

```
a, b, c = 2, 5, 8
fic = open('fichier.csv', 'w')
fic.write(str(a) + ";" + str(b) + ";" + str(c) + "\n")
fic.write(str(2*a) + ";" + str(2*b) + ";" + str(2*c))
fic.close()
```

Lecture d'un fichier

L'ouverture d'un fichier en mode lecture s'effectue avec la méthode `read`.

Plusieurs instructions sont disponibles :
- `ch = fic.read(n)` lit n caractères,
- `ch = fic.read()` lit tout le fichier,
- `ch = fic.readline()` lit la ligne courante et passe à la suivante.

Dans les trois cas, la variable `ch` obtenue est une chaîne de caractères.

L'instruction `ch = fic.readlines()` lit toutes les lignes. Dans ce cas, la variable `ch` obtenue est une liste de chaînes de caractères. Chaque élément est une ligne du fichier.
L'instruction `ch = [x for x in fic]` produit le même résultat.

Deux méthodes sur les chaînes de caractères sont importantes dans le traitement des données lues. Si `ch` est une chaîne de caractères, alors :
`ch.rstrip()` supprime le caractère de fin de ligne (par exemple "\n");
`ch.split(sep)` coupe la chaîne `ch` suivant le délimiteur `sep`, et renvoie une liste de sous-chaînes de `ch`. Avec `ch = "un,deux,trois"`,`ch.split(',')` renvoie `['un', 'deux', 'trois']`.
Le séparateur est une chaîne de caractères. Par défaut, c'est l'espace.

Si les données à lire sont des nombres qui doivent être utilisés dans des calculs, il faut les convertir en type int ou en type float suivant les besoins.
Voici un exemple :

```
fic = open("fichier.dat", "w")
fic.write(str(5) + "\t" + str(8.3) + "\t" + str(1e-4) + "\n")
fic.write(str(8) + "\t" + str(32.7) + "\t" + str(1e2))
fic.close()

fic = open("fichier.dat", "r")
for ligne in fic:
    liste = ligne.rstrip().split("\t")
    a, b, c = [float(x) for x in liste]
    print(a, b, c)

fic.close()
```