



- ◇ Les fichiers permettent d'enregistrer des données sur disque dur. Les avantages :
 - On peut ainsi enregistrer un grand nombre de données et l'enregistrement est permanent (par rapport à des données stockées en mémoire) ;
 - Ceci permet aussi de séparer les données du code qui les manipule, ce qui augmente la lisibilité du code ;
 - Enfin, les fichiers peuvent permettre à différents programmes d'échanger des données.

On présente ici les commandes les plus simples sur un exemple.

△ Un fichier n'est rien d'autre qu'une suite de nombres enregistrés sur un support. Pour donner un sens à cette suite de nombres, il faut connaître le *format* du fichier. On s'intéresse ici à des *fichiers texte* : ils sont constitués de chaînes de caractères et peuvent être consultés et modifiés à l'aide d'un simple éditeur.

Exemple. On considère le fichier ci-contre qui représente des mesures effectuées au cours d'un TP dont le but est d'obtenir une valeur approchée d'une résistance avec la relation $U = RI$. C'est un fichier texte qui comporte $2n + 2$ lignes (où n est le nombre de mesures). Il y a deux lignes particulières qui commencent par # qui servent juste à séparer les deux types de données. On va tout d'abord construire le fichier en deux temps (tout d'abord les valeurs de U puis celles de I). Ensuite on relira le fichier et on calculera la moyenne de U/I .

```
# Tension (V)
2
4
7
# Intensite (A)
0.079
0.17
0.277
```

◇ Première étape : ouvrir le fichier en mode *écriture* (c'est le rôle du `w`). De cette manière, un nouveau fichier vide est créé. Ensuite on remplit le fichier avec les valeurs de U puis on le ferme. Le caractère "`\n`" représente le retour à la ligne, c'est lui qui permet d'avoir des lignes distinctes dans le fichier.

```
nom_fichier = 'resultats-tp.txt'
fichier = open(nom_fichier, 'w')
print(type(fichier))
```

```
fichier.write("# Tension (V)\n")
for U in [2,4,7]:
    fichier.write(str(U)+"\n")
fichier.close()
```

◇ Étape suivante : on ouvre à nouveau le fichier, en mode *ajout* (avec **a**) ce qui permet d'écrire dedans en ajoutant les données à la fin du fichier existant. On remplit avec les valeurs de l'intensité et on ferme le fichier.

```
fichier = open(nom_fichier, 'a')
fichier.write("# Intensite (A)\n")
for I in [0.079, 0.170, 0.277]:
    fichier.write(str(I)+"\n")
fichier.close()
```

◇ Maintenant on ouvre à nouveau le fichier, cette fois-ci en mode lecture (**r**). On lit chaque ligne du fichier ce qui nous donne une chaîne de caractères. On vérifie que la première chaîne obtenue commence bien par "# " puis on lit les lignes suivantes pour remplir une liste **U** avec les tensions. On fait ceci jusqu'à obtenir à nouveau une ligne qui commence par "# ".

```
fichier = open(nom_fichier, 'r')
s = fichier.readline()
if s[0:2]!="# ":
    raise "Fichier non conforme"
    # Ceci déclenche une erreur
U = []
s = fichier.readline()
while s[0:2]!="# ":
    U.append(float(s))
    s = fichier.readline()
```

◇ On peut maintenant remplir une liste avec les intensités jusqu'à obtenir une ligne vide qui signifie que le fichier est épuisé.

```
I = []
s = fichier.readline()
while s!="":
    I.append(float(s))
    s = fichier.readline()
fichier.close()
print("U (V) :", U, "I (A) :", I)
```

U (V) : [2.0, 4.0, 7.0] I (A) : [0.079, 0.17, 0.277]

◇ Pour finir, on calcule la moyenne de U/I :

```
m = 0
for i in range(len(U)):
    m = m+U[i]/I[i]
R = m/len(U)
print("Résistance", R)
```

Résistance 24.70554186121736

Remarque.

- Bien entendu, en général les fichiers utilisés sont beaucoup plus grands ;
- Ici le fichier considéré est simplement une liste de données. On peut avoir besoin de gérer à la fois des données et des relations entre ces données, on se tourne alors plutôt vers des systèmes de *bases de données*. □