UNIVERSITÉ Larbi Ben Mhidi

Faculté des Sciences de la Terre, de la Géographie et de l'Aménagement du Territoire

Département Géographie et aménagement du territoire

Matière : Programmation

Niveau : 1ère année Licence Géographie et aménagement du territoire — Semestre 2

Crédits : 2 | **Coefficient :** 1

Enseignant : Bezzaz Soumia

Année universitaire : 2025–2026

Manipulation de Fichiers

Lecture, Écriture, Modification et Suppression

PARTIE I — INTRODUCTION AUX FICHIERS

1.1. Qu'est-ce qu'un fichier ?

Un **fichier** est une collection de données stockée sur un support de stockage (disque dur, clé USB, etc.) et identifiée par un nom. Les fichiers permettent de sauvegarder des informations de manière permanente, contrairement aux variables qui existent uniquement pendant l'exécution du programme.

Types de fichiers courants :

• **Fichiers texte** : .txt, .csv, .log
• **Fichiers de configuration** : .ini, .conf, .json
• **Fichiers de données** : .csv, .xlsx, .xml
• **Fichiers binaires** : .exe, .jpg, .pdf

Pourquoi manipuler des fichiers en programmation ?

• Sauvegarder des résultats de calculs ou d'analyses
• Lire des données d'observation (météo, démographie, etc.)
• Générer des rapports ou des logs d'exécution
• Échanger des données entre différents programmes

1.2. Chemins de fichiers (paths)

Le chemin d'un fichier indique son emplacement dans le système de fichiers.

Chemin absolu :

Chemin complet depuis la racine du système.

• Windows : C:\\Users\\Utilisateur\\Documents\\donnees.txt

• Linux/Mac : /home/utilisateur/documents/donnees.txt

Chemin relatif :

Chemin par rapport au répertoire actuel du programme.

• Fichier dans le même dossier : donnees.txt

• Fichier dans un sous-dossier : data/donnees.txt

• Fichier dans le dossier parent : ../donnees.txt

PARTIE II — LECTURE DE FICHIERS

2.1. Ouvrir et fermer un fichier

Pour manipuler un fichier, on utilise la fonction **open()** qui retourne un objet fichier.

Syntaxe :

```
fichier = open("nom_fichier", "mode")
```

Modes d'ouverture principaux :

Mode	Description
"r"	Lecture (read). Le fichier doit exister.
"w"	Écriture (write). Écrase le fichier s'il existe, sinon le crée.
"a"	Ajout (append). Ajoute à la fin du fichier, le crée s'il n'existe pas.
"r+"	Lecture et écriture. Le fichier doit exister.

Toujours fermer un fichier après utilisation :

```
fichier = open("donnees.txt", "r")
```

```
# Opérations sur le fichier
fichier.close()  # Important : libère les ressources
```

2.2. Utilisation de with (recommandé)

Le mot-clé **with** ferme automatiquement le fichier, même en cas d'erreur. C'est la méthode recommandée.

Syntaxe :

```
with open("fichier.txt", "r") as fichier:
    contenu = fichier.read()
    # Le fichier se ferme automatiquement à la fin du bloc
```

2.3. Méthodes de lecture

Python offre plusieurs méthodes pour lire un fichier :

1. **read()** : lit tout le contenu du fichier en une seule chaîne
2. **readline()** : lit une ligne à la fois
3. **readlines()** : lit toutes les lignes et retourne une liste

Exemple : Lire tout le fichier avec read()

```
with open("villes.txt", "r") as fichier:
    contenu = fichier.read()
    print(contenu)
```

Exemple : Lire ligne par ligne avec readlines()

```
with open("villes.txt", "r") as fichier:
    lignes = fichier.readlines()  # Retourne une liste
    for ligne in lignes:
        print(ligne.strip())  # strip() enlève les espaces et les retours à la ligne
```

Exemple : Boucle directe (plus efficace)

```
with open("villes.txt", "r") as fichier:
    for ligne in fichier:  # Parcourt directement le fichier
        print(ligne.strip())
```

PARTIE III — ÉCRITURE DANS UN FICHIER

3.1. Écrire du texte

Pour écrire dans un fichier, on utilise le mode **"w"** (écrase le fichier existant) ou **"a"** (ajoute à la fin).

Exemple: Écrire dans un nouveau fichier

```
with open("rapport.txt", "w") as fichier:
    fichier.write("Rapport d'étude géographique\n")
    fichier.write("============================\n")
    fichier.write("Zone : Algérie du Nord\n")
```

Remarque :

• write() n'ajoute pas automatiquement de retour à la ligne. Il faut ajouter \n manuellement.

• Le mode "w" écrase complètement le fichier existant.

Exemple: Écrire plusieurs lignes avec writelines()

```
villes = ["Alger\n", "Oran\n", "Constantine\n"]


with open("villes.txt", "w") as fichier:
    fichier.writelines(villes)
```

3.2. Ajouter du contenu

Le mode **"a"** permet d'ajouter du contenu à la fin d'un fichier sans écraser ce qui existe déjà.

Exemple:

```
with open("log.txt", "a") as fichier:
    fichier.write("Nouvelle entrée ajoutée\n")
```

PARTIE IV — MODIFICATION DE FICHIERS

Python ne permet pas de modifier directement un fichier en place. La stratégie générale est :

1. Lire le contenu du fichier
2. Modifier le contenu en mémoire
3. Réécrire tout le fichier avec les modifications

Exemple: Remplacer un mot dans un fichier

```
# Lecture
with open("document.txt", "r") as fichier:
    contenu = fichier.read()

# Modification
contenu_modifie = contenu.replace("ancien", "nouveau")

# Réécriture
with open("document.txt", "w") as fichier:
    fichier.write(contenu_modifie)
```

PARTIE V — SUPPRESSION DE FICHIERS

Pour supprimer des fichiers, on utilise le module **os** (Operating System).

5.1. Supprimer un fichier

```
import os

# Vérifier si le fichier existe avant de le supprimer
if os.path.exists("fichier_temporaire.txt"):
    os.remove("fichier_temporaire.txt")
    print("Fichier supprimé")
else:
    print("Le fichier n'existe pas")
```

5.2. Vérifier l'existence d'un fichier

```
import os

if os.path.exists("donnees.txt"):
    print("Le fichier existe")
```

```
else:
    print("Le fichier n'existe pas")
```

EXEMPLES APPLIQUÉS EN GÉOGRAPHIE

1. Enregistrer des relevés de terrain

```
# Enregistrer des coordonnées GPS
points = [
    (36.7538, 3.0588, "Alger"),
    (35.6976, -0.6337, "Oran"),
    (36.3650, 6.6147, "Constantine")
]


with open("coordonnees.txt", "w") as fichier:
    fichier.write("Latitude,Longitude,Ville\n")
    for lat, lon, ville in points:
        fichier.write(f"{lat},{lon},{ville}\n")
```

2. Lire et analyser des données météorologiques

```
# Calculer la température moyenne à partir d'un fichier
temperatures = [ ]


with open("temperatures.txt", "r") as fichier:
    for ligne in fichier:
        temp = float(ligne.strip())
        temperatures.append(temp)


moyenne = sum(temperatures) / len(temperatures)
print(f"Température moyenne : {moyenne:.2f}°C")
```