□ 1. Les Tableaux (numpy array)

Un tableau contient des valeurs du **même type** avec une taille fixe. On utilise **numpy**. Les indices commencent à **0**.

□ Importer et déclarer

python

```python
from numpy import array
T = array([0] * N)              # N entiers
T = array([float()] * N)        # N réels
T = array(['']  * N, dtype='U20') # N chaînes
```

Opération	Code Python
Accéder à l'élément i	`T[i]`
Modifier l'élément i	`T[i] = valeur`
Nombre d'éléments	`len(T)`
Parcourir tous les éléments	`for i in range(n):`

□□ Saisie et affichage

python

```python
n = int(input('Taille : '))
T = array([0] * n)
for i in range(n) :
    T[i] = int(input('Valeur ' + str(i) + ' : '))
for i in range(n) : print(T[i])
```

□□ Ne jamais écrire **print(T)** — afficher chaque élément un par un avec une boucle for.

□□ Exercice 1 — Somme et Moyenne

□ **Objectif :** Saisir n entiers dans un tableau, afficher leur somme et leur moyenne.

terminal

```
Taille : › 4
Valeur 0 : › 8
```

```
Valeur 1 : › 12
Valeur 2 : › 6
Valeur 3 : › 10
Somme = 36
Moyenne = 9.0
```

□□ Exercice 2 — Maximum et son Indice

□ **Objectif :** Saisir un tableau de n entiers. Trouver la valeur maximum et son indice.

terminal

```
Taille : › 5
Valeur 0 : › 3
Valeur 1 : › 7
Valeur 2 : › 2
Valeur 3 : › 9
Valeur 4 : › 4
Max = 9 Indice = 3
```

□□ Exercice 3 — Inverser un Tableau

□ **Objectif :** Saisir un tableau de n entiers, le recopier dans l'ordre inversé dans un 2ème tableau.

terminal

```
Taille : › 5
Valeur 0 : › 1
Valeur 1 : › 2
Valeur 2 : › 3
Valeur 3 : › 4
Valeur 4 : › 5
5 4 3 2 1
```

□ 2. Les Dictionnaires (dict)

Un dictionnaire sert à créer des **enregistrements** — comme une fiche élève avec nom, âge et note.

□ Créer un dictionnaire / enregistrement

python

```python
notes = {'Ali': 12, 'Sara': 15, 'Omar': 8}
eleve = dict(nom=str(), note=float())
```

```
eleve['nom']  = 'Ali'
eleve['note'] = 14.5
D = {}   # Dictionnaire vide
```

Opération	Code Python
Lire une valeur	`D['cle']`
Ajouter / modifier	`D['cle'] = valeur`
Tester si clé existe	`cle in D`
Nombre de paires	`len(D)`
Parcourir les clés	`for cle in D:`

python

```
nom = input('Chercher : ')
if nom in notes :
    print(notes[nom])
else :
    print('Introuvable')
```

□□ Toujours vérifier **cle in D** avant d'accéder à **D[cle]** — sinon le programme plante.

□□ Exercice 4 — Fiche Élève

□ **Objectif :** Créer un enregistrement élève (nom, prénom, note), saisir les valeurs, afficher la fiche.

terminal

```
Nom : › Ali
Prénom : › Ben
Note : › 13.5
Ali Ben — Note : 13.5
```

□□ Exercice 5 — Annuaire Téléphonique

□ **Objectif :** Saisir n contacts (nom + numéro) dans un dictionnaire, puis rechercher un nom.

terminal

```
Nombre de contacts : › 2
Nom : › Ali
Numéro : › 2222
```

```
Nom : › Sara
Numéro : › 5555
Chercher : › Sara
Sara → 5555
```

□□ Exercice 6 — Fréquence des Lettres

□ **Objectif :** Saisir un mot et compter combien de fois chaque lettre apparaît.

terminal

```
Mot : › banana
b : 1
a : 3
n : 2
```

□□ Exercice 7 — Modifier un enregistrement

□ **Objectif :** Créer un enregistrement produit (nom + prix). Afficher le prix initial, appliquer une réduction de 20%, afficher le nouveau prix.

terminal

```
Nom du produit : › Clavier
Prix initial : › 50.0
Prix après réduction : 40.0
```

□□ Exercice 8 — Notes de classe

□ **Objectif :** Saisir n élèves (nom + note). Afficher les admis (note ≥ 10) puis le nom de l'élève ayant la meilleure note.

terminal

```
Nombre d'élèves : › 3
Nom : › Ali
Note : › 12
Nom : › Sara
Note : › 8
Nom : › Omar
Note : › 15
Admis : Ali Omar
Meilleur : Omar (15)
```

□□ Exercice 9 — Enregistrement avec tableau

□ **Objectif :** Créer un enregistrement étudiant avec nom (str) et un tableau de 3 notes (numpy). Calculer et afficher la moyenne.

□ Un enregistrement peut contenir un tableau comme champ : **e['notes'] = array([0.0]*3)**

terminal

```
Nom : › Youssef
Note 0 : › 14
Note 1 : › 16
Note 2 : › 13
Youssef — Moyenne : 14.33
```

□□ 3. Les Fichiers (texte & binaire)

Un fichier permet de sauvegarder des données même après la fin du programme. **Texte** : lignes lisibles. **Binaire** : objets Python avec pickle.

□ 3.1 Fichiers Texte

Opération	Code Python
Ouvrir en lecture	`f = open('nom.txt', 'r')`
Ouvrir en écriture	`f = open('nom.txt', 'w')`
Ouvrir en ajout	`f = open('nom.txt', 'a')`
Lire une ligne	`ch = f.readline()`
Écrire + saut de ligne	`f.write(ch + '\n')`
Fermer	`f.close()`

□□ Écriture

python

```python
f = open('notes.txt', 'w')
n = int(input('Nombre : '))
for i in range(n) :
    note = input('Note ' + str(i+1) + ' : ')
    f.write(note + '\n')
f.close()
```

□ Lecture ligne par ligne

```python
f = open('notes.txt', 'r')
ch = f.readline()
while ch != '' :
    print(ch.strip())
    ch = f.readline()
f.close()
```

□□ Exercice 10 — Écrire des Notes

□ **Objectif :** Saisir n notes et les écrire dans 'notes.txt', une note par ligne.

```terminal
Nombre : › 3
Note 1 : › 12
Note 2 : › 15
Note 3 : › 9
→ notes.txt créé avec 3 lignes
```

□□ Exercice 11 — Afficher les Admis

□ **Objectif :** Lire 'notes.txt' et afficher uniquement les notes ≥ 10.

Contenu de notes.txt : **12 / 7 / 15 / 9 / 10**

```terminal
12
15
10
```

□ 3.2 Fichiers Binaires (pickle)

Opération	Code Python
Importer	`from pickle import load, dump`
Ouvrir en écriture bin.	`f = open('nom.dat', 'wb')`
Ouvrir en lecture bin.	`f = open('nom.dat', 'rb')`
Ouvrir en ajout bin.	`f = open('nom.dat', 'ab')`
Écrire un objet	`dump(objet, f)`
Lire un objet	`objet = load(f)`

Fermer	`f.close()`

□□ Écriture d'enregistrements

```python
from pickle import load, dump
f = open('eleves.dat', 'wb')
n = int(input('Nombre : '))
for i in range(n) :
    e = dict(nom=str(), note=float())
    e['nom']  = input('Nom : ')
    e['note'] = float(input('Note : '))
    dump(e, f)
f.close()
```

□ Lecture jusqu'à la fin

```python
from pickle import load, dump
f = open('eleves.dat', 'rb')
fin = False
while not fin :
    try :
        e = load(f)
        print(e['nom'], e['note'])
    except :
        fin = True
f.close()
```

□ Fichier texte : fin quand **readline()** retourne **''**. Fichier binaire : fin détectée par **try / except**. Mode **'ab'** pour ajouter sans écraser. Toujours fermer avec **f.close()**.

□□ Exercice 12 — Écrire des enregistrements

□ **Objectif :** Saisir n enregistrements (nom + note) et les sauvegarder dans 'eleves.dat'.

```terminal
Nombre : › 2
Nom : › Ali
```

```
Note : › 14.0
Nom : › Sara
Note : › 12.5
→ eleves.dat créé avec 2 enregistrements
```

□□ Exercice 13 — Lire et afficher tous les enregistrements

□ **Objectif :** Lire 'eleves.dat' et afficher le nom et la note de chaque élève.

terminal

```
Ali 14.0
Sara 12.5
```

□□ Exercice 14 — Moyenne depuis fichier binaire

□ **Objectif :** Lire 'eleves.dat' et calculer la moyenne des notes. Contenu : Ali:12 Sara:15 Omar:9

terminal

```
Moyenne = 12.0
```

□□ Exercice 15 — Afficher les admis depuis fichier binaire

□ **Objectif :** Lire 'eleves.dat' et afficher uniquement les noms des élèves ayant une note ≥ 10.

terminal

```
Ali — 14.0 ■
Sara — 12.5 ■
```

□□ Exercice 16 — Ajouter un enregistrement

□ **Objectif :** Ajouter un nouvel élève dans 'eleves.dat' sans effacer les données existantes. Utiliser le mode **'ab'**.

□ Mode **'wb'** écrase le fichier. Mode **'ab'** ajoute à la fin — utiliser pour ne pas perdre les données.

terminal

```
Nom : › Omar
Note : › 11.0
→ Omar ajouté dans eleves.dat
```

□□ Exercice 17 — Rechercher un élève dans le fichier

□ **Objectif :** Lire 'eleves.dat', demander un nom, afficher sa note ou 'Introuvable' si absent.

terminal

```
Chercher : › Sara
Sara — Note : 12.5
```

Autre exemple :

terminal

```
Chercher : › Khalil
Introuvable
```

□ Récapitulatif — Les 3 structures en un coup d'œil

□ Tableaux numpy

Importer	`from numpy import array`
Déclarer entiers	`T = array([0] * N)`
Déclarer réels	`T = array([float()] * N)`
Accéder	`T[i]`
Modifier	`T[i] = valeur`
Taille	`len(T)`
Afficher	`for i in range(n): print(T[i])`

□ Dictionnaires

Créer vide	`D = {}`
Enregistrement	`e = dict(nom=str(), note=float())`
Lire	`D['cle']`
Ajouter/modifier	`D['cle'] = valeur`
Tester existence	`if cle in D:`
Parcourir	`for cle in D: print(D[cle])`

□□ Fichiers Texte

Ouvrir	`f = open('f.txt', 'r')`
Lire une ligne	`ch = f.readline()`
Fin du fichier	`while ch != '':`
Écrire	`f.write(ch + '\n')`
Fermer	`f.close()`

□ Fichiers Binaires (pickle)

Importer	`from pickle import load, dump`
Écrire (wb)	`f = open('f.dat', 'wb')`
Lire (rb)	`f = open('f.dat', 'rb')`
Ajouter (ab)	`f = open('f.dat', 'ab')`
Écrire objet	`dump(objet, f)`
Lire objet	`objet = load(f)`
Fin fichier	`try: load(f) except: fin = True`
Fermer	`f.close()`