□ 1. Les Tableaux (numpy array)

Un tableau contient des valeurs du **même type** avec une taille fixe. On utilise la bibliothèque **numpy** pour le créer. Les indices commencent toujours à **0**.

□ Importer et déclarer

python

```python
from numpy import array
# Tableau de N entiers (initialisés à 0)
T = array([0] * N)
# Tableau de N réels (initialisés à 0.0)
T = array([float()] * N)
# Tableau de N chaînes de caractères vides
T = array([''] * N, dtype='U20')
```

□□ Opérations

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
from numpy import array
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.

□ Exercices — Tableaux 1D

□□ 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, puis afficher ce 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 Matrices (tableaux 2D)

Une matrice, c'est un **tableau dans un tableau** — comme une grille avec des lignes et des colonnes. Chaque case est repérée par **deux indices** : le numéro de ligne **i** et le numéro de colonne **j**.

	j=0	j=1	j=2	j=3
i=0	[0][0] 10	[0][1] 20	[0][2] 30	[0][3] 40
i=1	[1][0] 50	[1][1] 60	[1][2] 70	[1][3] 80
i=2	[2][0] 90	[2][1] 11	[2][2] 12	[2][3] 13

Matrice 3 lignes × 4 colonnes — chaque case repérée par M[i][j]

□ Déclarer une matrice

```python
from numpy import array
# Matrice de n lignes et m colonnes (entiers)
n = 3   # nombre de lignes
m = 4   # nombre de colonnes
M = array([[0] * m] * n)
# Accéder à la case ligne i, colonne j
M[i][j] = valeur
print(M[i][j])
```

□□ Saisie et affichage d'une matrice

```python
from numpy import array
n = int(input('Lignes : '))
m = int(input('Colonnes : '))
M = array([[0] * m] * n)
# Saisie
for i in range(n) :
    for j in range(m) :
        M[i][j] = int(input('M['+str(i)+']['+str(j)+'] : '))
# Affichage ligne par ligne
for i in range(n) :
    ligne = ''
    for j in range(m) :
        ligne = ligne + str(M[i][j]) + '\t'
    print(ligne)
```

□ Deux boucles imbriquées : la boucle **externe** parcourt les **lignes (i)**, la boucle **interne** parcourt les **colonnes (j)**.
Ne jamais écrire **print(M)** — afficher ligne par ligne avec concaténation.

Opération	Code Python
Déclarer n×m entiers	`M = array([[0] * m] * n)`

Accéder case (i,j)	`M[i][j]`
Modifier case (i,j)	`M[i][j] = valeur`
Parcourir toutes les cases	`for i in range(n): for j in range(m):`
Afficher une ligne	`ligne = '' for j: ligne += str(M[i][j])+' ' print(ligne)`

□ Exercices — Matrices

□□ Exercice 4 — Afficher une matrice

□ **Objectif :** Saisir une matrice de n lignes et m colonnes d'entiers, puis l'afficher ligne par ligne.

terminal

```
Lignes :
› 2
Colonnes :
› 3
M[0][0] :
› 1
M[0][1] :
› 2
M[0][2] :
› 3
M[1][0] :
› 4
M[1][1] :
› 5
M[1][2] :
› 6
1 2 3
4 5 6
```

□□ Exercice 5 — Somme de chaque ligne

□ **Objectif :** Saisir une matrice n×m. Afficher la somme des éléments de chaque ligne.

terminal

```
Lignes :
› 3
Colonnes :
› 3
... (saisie)
Somme ligne 0 : 6
Somme ligne 1 : 15
Somme ligne 2 : 24
```

□□ Exercice 6 — Diagonale principale

□ **Objectif :** Saisir une matrice carrée n×n. Afficher uniquement les éléments de la **diagonale principale** (cases où i == j).

□ La diagonale principale : cases (0,0), (1,1), (2,2)… — là où i == j.

terminal

```
Taille n :
› 3
... (saisie)
Diagonale : 1 5 9
```

□□ Exercice 7 — Matrice identité

□ **Objectif :** Créer et afficher une matrice identité de taille n×n (1 sur la diagonale, 0 partout ailleurs) sans saisie.

terminal

```
Taille n :
› 4
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
```

□ Récapitulatif

□ Tableaux 1D

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])`

□ Matrices 2D

Déclarer n×m	`M = array([[0] * m] * n)`
Accéder (i,j)	`M[i][j]`
Modifier (i,j)	`M[i][j] = valeur`
Parcourir	`for i in range(n): for j in range(m):`
Diagonale	`if i == j :`
Afficher ligne	`ligne = '' for j: ligne += str(M[i][j])+' ' print(ligne)`