1 Définition

Un **tableau indexé** est une **séquence d'éléments identifiés par leur indice** qui est leur position dans le tableau.

Rappel : en informatique, **on compte** les séquences **à partir de l'indice 0** (comme pour les tuples).

2 Tableau en Python

En Python, les tableaux indexés sont implémentés par des **listes**.
(*Remarque* : une liste au sens général de l'informatique est un concept un peu différent : cf cours Term)

Les listes sont définies **entre crochets**.

Création d'un tableau (*list* python) :

```
liste = ['Henri', 'Louis', 'ELISABETH']
```

Accès à un élément (on accède par indice comme pour les tuples) :

```
liste[0]  # renvoie 'Henri'
```

Accès à un élément (à partir de la fin) :
Les indices (ou index) d'un liste sont comptés de 0 à $(n-1)$ dans l'ordre de la liste (de « la gauche vers la droite »), mais aussi de (-1) à $(-n)$ dans l'ordre inverse (« la gauche vers la droite »).

valeurs	'Henri'	'Louis'	'ELISABETH'
indice (à l'endroit)	0	1	2
indice (à l'envers)	-3	-2	-1

```
liste[-1]  # vaut 'ELISABETH'
liste[-3]  # vaut 'Henri'
```

Les listes de Python sont des objets mutables, contrairement aux tuples :

Modification d'un élément :

```
liste[2]  = 'Elisabeth'
```

Ajout d'un élément à la fin de la liste : méthode **append()** :

```
liste.append('Cesar')
```

3 Création d'un tableau par compréhension

Une structure de création de tableau est assez courante : on désire ajouter une série d'éléments à un tableau sous certaines conditions.

Voir l'exemple suivant :
liste ⟵ *tableau_vide* ;
n ⟵ *borne* ;
pour *i* = *1* **à** *n* **faire**
  si *i est pair* **alors**
    ajouter *i* à *liste*
  fin si
fin pour

Cette structure de création de tableau peut être effectuée en une ligne dite « **en compréhension** ».

En Python, voici l'implémentation correspondante pour l'exemple précédent :

```
liste = [i for i in range(1, n+1) if i % 2 == 0]
```

La structure générale est donc :

```
liste = [expression for compteur in range(etendue) if conditions]
```

4 Représentation de matrices : tableau de tableaux

4.1 Principe

Une matrice est un objet mathématique qui peut être présenté comme un tableau de m lignes et n colonnes.
En programmation, une matrice s'implémente facilement comme un tableau de tableaux. Le tableau principal possède m éléments, qui sont chacun un tableau de n éléments.

$$\begin{pmatrix} 00 & 01 & \dots & 0n_{-1} \\ 10 & 11 & \dots & 1n_{-1} \\ \vdots & \vdots & \dots & \vdots \\ m_{-1}0 & m_{-1}1 & \dots & m_{-1}n_{-1} \end{pmatrix}$$

Exemple : création d'une matrice de nombres aléatoires entre 1 et 6 :

```
m, n = 3, 5 # 3 lignes de 5 colonnes
matrice = [ [randint(1, 6) for i in range(n)] for j in range(m)]
```

$$matrice \longmapsto \begin{pmatrix} 6 & \underline{3} & 1 & 1 & 4 \\ 4 & 5 & 2 & 2 & 4 \\ 6 & 1 & 1 & 2 & 5 \end{pmatrix}$$

Pour accéder à un élément de la matrice, en accédant d'abord à une ligne de la matrice (entre 0 et $m-1$) puis à un élément de la ligne (entre 0 et $n-1$).

Exemple :
Le chiffre 3 souligné de l'exemple précédent est accessible par : `matrice[0][1]`.

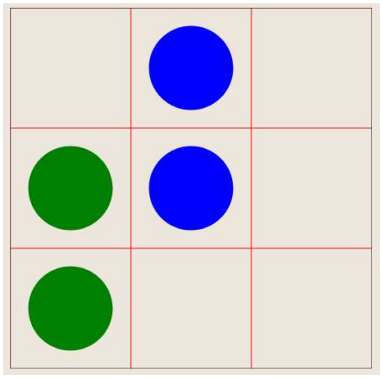
4.2 Application classique : grille de jeu

On désire représenter une grille du jeu Puissance 4.
Un choix de représentation assez classique peut être le suivant :

- chaque case est représentée par un nombre :
  Ex : case vide = 0, pion jaune = 1 , pion rouge = 2.
- chaque ligne est représentée par une liste de cases :
  Ex : [0, 0, 1, 1, 0, 2, 1, 0]
- la grille complète est une liste de lignes :
  Ex : [[0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0], ... , [0, 0, 1, 1, 0, 2, 1, 0], [0, 0, 1, 1, 0, 2, 1, 0], [0, 0, 1, 1, 0, 2, 1, 0]]

Exercice :

1. Écrire la liste correspondant à la grille suivante du jeu de Morpion :



2. Comment accéder à la case du coin inférieur gauche ?
3. Écrire un code Python qui affiche ligne après ligne l'état de la grille.
   Par exemple, pour la grille donnée précédemment, l'affichage doit être :

```
0 1 0
2 1 0
2 0 0
```