

## Représentation des données : les tableaux

### 1 Définition et exemples

**Définition 1.** Un **tableau** (le mot clé correspondant en Python est `list`) est un objet informatique (une **structure de données**) qui permet de stocker plusieurs valeurs.

**Exemple 1.** Au premier trimestre, Sarah a obtenu les notes suivantes : 10, 15, 11, 18, 5, et 20. Si on souhaite stocker ces notes dans des variables, on doit donc écrire :

Code Source

```
1 sarah_note1 = 10
2 sarah_note2 = 15
3 sarah_note3 = 11
4 sarah_note4 = 12
5 sarah_note5 = 18
6 sarah_note6 = 5
7 sarah_note7 = 20
```

En prenant exemple sur les tableurs comme LibreOffice Calc, on pourrait plutôt décider de stocker ces variables dans un tableau :

Notes de Sarah	10	15	11	12	18	5	20
----------------	----	----	----	----	----	---	----

En Python nous allons déclarer une variable `notes_sarah` de type tableau qui stocke les notes de Sarah. On utilise pour cela les symboles **crochet** `'[` et `']`, et on sépare les éléments du tableau par des **virgules**.

Console Python

```
1 >>> notes_sarah = [10, 15, 11, 12, 18, 5, 20]
2 >>> notes_sarah[0]
3 10
4 >>> notes_sarah[3]
5 12
```

C'est beaucoup plus court! Allons voir maintenant comment créer et manipuler les tableaux en Python.

## 2 Création et manipulation de tableaux

### 2.1 Création d'un tableau

**Définition 2.** Pour créer un tableau en Python, trois méthodes sont possibles :

- par **énumération** : on définit le tableau par la liste de ses valeurs, comme dans l'exemple 1 :

```
_____ Console Python _____
1  >>> notes_sarah = [10, 15, 11, 12, 18, 5, 20]
2  >>> print(notes_sarah)
3  [10, 15, 11, 12, 18, 5, 20]
```

- par **initialisation** : on définit un tableau de n cases, dont tous les contenus sont nuls :

```
_____ Console Python _____
1  >>> n = 10
2  >>> tableau = [0]*n
3  >>> print(tableau)
4  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

- par **ajouts successifs** : on rajoute les éléments au tableau les uns après les autres (à la main ou avec une boucle) :

```
_____ Console Python _____
1  >>> tableau = []
2  >>> tableau.append(5)
3  >>> tableau.append(120)
4  >>> print(tableau)
5  [5, 120]
```

**Remarques.** Il existe un objet particulier, le **tableau vide** : c'est un tableau qui ne contient aucune valeur.

"append" signifie "ajouter à la fin". Que se serait-il passé si on avait inversé les lignes 2 et 3 du code correspondant aux ajouts successifs ?

**Définition 3.** La **taille** d'un tableau est le nombre d'éléments qui le composent. En Python, len(tableau) renvoie la taille de tableau.

**Exemple 2.** Quelle est la taille des tableaux suivants ?

- notes\_sarah = [10,15,11,12,18,5,20] a pour taille 7.
- [4,5,2,3,8,5,65,8] a pour taille : .....
- [0]\*123 a pour taille : .....
- [] a pour taille : .....

### 2.2 Manipulation d'un tableau

**Définition 4.** Soit t un tableau. Les éléments du tableau sont ordonnés et chaque valeur du tableau est repérée par un unique **indice**. On accède à chaque élément du tableau par son indice.

**ATTENTION.** En Python, les indices commencent à 0.

**Exemple 3.** Reprenons l'exemple des notes de Sarah.

notes_sarah =	[10,	15,	11,	12,	18,	5,	20]
Indice	0	1	2	3	4	5	6

Pour accéder à la cinquième note (18/20) de Sarah :

```
Console Python
1 >>> notes_sarah = [10, 15, 11, 12, 18, 5, 20]
2 >>> notes_sarah[4]
3 18
```

**Exercice 1.** On considère le tableau suivant :

```
test = [555, 951, -159, 258, 852, -456, 654, 123, 987]
```

1. Quelle est la taille du tableau? .....
2. Quel est l'élément d'indice 2? .....
3. Quel est l'indice de 852? .....
4. Quel est l'indice du dernier élément? .....

**Erreur d'indice.** Lorsque l'on demande un indice trop grand pour la taille du tableau, Python renvoie une erreur appelée `IndexError`.

```
Console Python
1 >>> notes_sarah = [10, 15, 11, 12, 18, 5, 20]
2 >>> notes_sarah[10]
3 Traceback (most recent call last):
4   File "<stdin>", line 1, in <module>
5   IndexError: list index out of range
```

**Remarque.** L'indice du dernier élément du tableau est la taille du tableau **moins 1**.

**Modification des éléments d'un tableau.** Si `t` est une variable de type tableau, on peut modifier les valeurs de ses éléments à l'aide de l'indice correspondant :

```
Console Python
1 >>> notes_sarah = [10, 15, 11, 12, 18, 5, 20]
2 >>> notes_sarah[5]
3 5
4 >>> notes_sarah[5] = 15
5 >>> print(notes_sarah)
6 [10, 15, 11, 12, 18, 15, 20]
```

**Exercice 2.** Le professeur a été trop sévère avec les élèves dans sa notation. Il décide donc d'augmenter les notes des premier et troisième contrôle de 1 point.

Écrire une fonction `mise_a_jour` qui prend en entrée un tableau de notes `t` de taille supérieure ou égale à 3 et renvoie en sortie le tableau des notes mises à jour. Vous testerez votre fonction :

`mise_a_jour([5, 7, 6, 9, 10, 11, 10])` renvoie `[6, 7, 7, 9, 10, 11, 10]`.

## Correction.

Code Source

```
1 def mise_a_jour(t):
2     # On ne peut pas avoir 21/20.
3     if t[0] <= 19:
4         t[0] = t[0] + 1
5     if t[2] <= 19:
6         t[2] = t[2] + 1
7     return t
8
9 # Test
10 t = [5, 7, 6, 9, 10, 11, 10]
11 print('Après mise à jour des notes :', mise_a_jour(t))
12 # Output :
13 # Après mise à jour des notes : [6, 7, 7, 9, 10, 11, 10]
```