Les listes en Python

Les **listes**(ou **list**) en **python**sont une variable dans laquelle on peut mettre plusieurs variables qu’on appellera des items   
On l’a déjà dit mais je le redis : l’avantage des listes par rapport aux tableaux est qu’on peut en modifier la taille.

### Création de la liste :

Créer une liste vide : **MaListe = []** Créer une liste contenant des valeurs : **MaListe = [1,2,3,6,9]**

### Afficher une valeur de la liste Les éléments composant la liste sont « repérés » par l’index du tableau. Ainsi :

**Print(MaListe[0])** affiche 1  
**Print(MaListe[1])** affiche 2  
**Print(MaListe[4])** affiche 9

### Modification de la liste : Pour ajouter une valeur en queue de liste : append

**MaListe.append(8)** transformera notre liste en **[1,2,3,6,9,8]** (on a ajouté la valeur 8 )

Pour modifier une valeur dans la liste :  
**MaListe[0]=15** transformera notre liste en **[15,2,3,6,9,8]**

## Compter le nombre d'items d'une liste Print (len(MaListe)) renvoi 6

**Quelques tips pratiques** :

>>> liste[-1] *# Cherche la dernière occurence*

500

>>> liste[-4:] *# Affiche les 4 dernières occurrences*

[500, 250, 100, 10]

>>> liste[:] *# Affiche toutes les occurences*

[1, 10, 100, 250, 500]

>>> liste[2:4] = [69, 70]

[1, 10, 69, 70, 500]

>>> liste[:] = [] *# vide la liste*

[]

**Copier une liste**. ! Attention, piège, ça ne fonctionne pas comme des variables de type « classiques » !

Beaucoup de débutants font l'erreur de copier une liste de cette manière

x = [1,2,3]

y = x

Or si vous changez une valeur de la liste **y**, la liste **x**sera elle aussi affectée par cette modification:

>>> x = [1,2,3]

>>> y = x

En fait cette syntaxe permet de travailler sur un même élément nommé différemment

Alors comment copier une liste qui sera indépendante?

>>> y[0] = 4

>>> x

[4, 2, 3]

Vous pouvez utiliser la fonction *deepcopy* du module *copy :*

**import copy**

**x = [[1,2,3,2,6]**

**y = deepcopy(x)**

**ou**

**x = [[1,2,3,2,6]**

**y = x[ :]**

On peut faire beaucoup d’autres manipulations avec les listes (les trier, les assembler, chercher des éléments dedans…)

Listes à deux dimensions

Des tables à deux dimensions sont appelées matrices ou tableaux bidimensionnels. En Python, n'importe quelle table peut être représentée comme une liste de listes (une liste, où chaque élément est à son tour une liste). Par exemple, voici le programme qui crée un tableau numérique avec deux lignes et trois colonnes, puis fait quelques manipulations avec celui-ci:

*Sorties :*

[1, 2, 3]

[4, 5, 6]

[1, 2, 3]

3

[[1, 7, 3], [4, 5, 6]]

[1, 7, 3]

[1, 7, 9]

[1, 7, 9]

Le premier élément d' a ici - a[0] - est une liste de nombres [1, 2, 3] . Le premier élément de cette nouvelle liste est a[0][0] == 1 ; de plus :

, a[0][1] == 2 , a[0][2] == 3 , a[1][0] == 4 ,

a[1][1] == 5 , a[1][2] == 6

a = [[1, 2, 3], [4, 5, 6]]

print(a[0])

print(a[1])

b = a[0]

print(b)

print(a[0][2])

a[0][1] = 7

print(a)

print(b)

b[2] = 9

print(a[0])

print(b)

Pour traiter un tableau à deux dimensions, on utilise généralement des boucles imbriquées. La première boucle parcourt le numéro de ligne, la seconde boucle parcourt les éléments à l'intérieur d'une rangée. Par exemple :

**a = [[1, 2, 3, 4], [5, 6], [7, 8, 9]]**

*Sortie :*

1 2 3 4 5 6 7 8 9

*Alternative :*

a = [[1, 2, 3, 4], [5, 6], [7, 8, 9]]

for row in a:

for elem in row:

print(elem)

**for i in range(len(a)):**

**for j in range(len(a[i])):**

**print(a[i][j])**