

Épisode II : Les listes

EXERCICE 1

Dans l'environnement Spyder, entrer l'instruction suivante et analyser ce qui est affiché :

```
for i in [2, 8, 5]:  
    print(i, i * i)
```

Même question pour :

```
u = [2, 8, 5]  
for i in u:  
    print(i, i * i)
```


La variable `i` est-elle définie après exécution de ces instructions ? Si oui, quelle est sa valeur ?

EXERCICE 2

Définir la liste $L = [17, 38, 10, 25, 72]$, puis effectuez les actions listées ci-dessous. Après chaque action, vérifiez que celle-ci s'est correctement effectuée.

1. Ajouter l'élément 12 en fin de la liste L .
2. Mettre dans une variable `val` la valeur de l'élément d'indice 4.
3. Mettre dans une variable `ind` l'indice de l'élément 25.
4. Mettre dans une variable `long` la longueur de la liste L .
5. Supprimer de la liste L l'élément 38.
6. Ajouter les éléments $[8, 14, 29]$ en fin de la liste L .
7. Supprimer de la liste L l'élément d'indice 2.
8. Remplacer dans la liste L l'élément d'indice 0 par l'élément 55.
9. Mettre dans une liste $L2$ la sous-liste de L allant du 2^{ème} au 5^{ème} élément.
10. Mettre dans une liste $L3$ la sous-liste de L allant du début de la liste au 3^{ème} élément.
11. Mettre dans une liste $L4$ la sous-liste de L allant du 3^{ème} élément à la fin de la liste.
12. Mettre dans une liste $L5$ la concaténation des liste $L2$, $L3$ et $L4$.
13. Recopier la liste L dans une liste LL , ajouter l'élément 99 en fin de la liste LL ; quelle est maintenant la valeur des listes L et LL ? Avez-vous réalisé une copie ou une identification ?
14. Définir en compréhension la liste $L6$, composée des cubes des entiers de 5 à 11.
15. Définir en compréhension la liste $L7$, composée des images des entiers de 1 à 8 par la fonction $x \mapsto 2x^2 - 3x + 5$.
16. Définir en compréhension la liste $L8$, composée des carrés des entiers pairs compris entre 9 et 21.
17. Définir en compréhension la liste $L9$, composée des multiples de 3 compris entre 111 et 130.

REMARQUE

 Définir une liste en compréhension est une particularité de Python.

EXERCICE 3

Dans l'environnement Spyder, tester chacun des groupes d'instructions suivantes et analyser les résultats comme précédemment :

```
for a in ["a", "b", "c"]:  
    for i in [1, 2, 3]:  
        print(a, i)  
  
u = [2, 6, 1, 10, 6]  
v = [2, 5, 6, 4]  
for x in u:  
    for y in v:  
        print(x, y, x + y, x == y)
```

EXERCICE 4

On considère la définition suivante :

```
def sommeListe(L):  
    s = 0  
    for i in L:  
        s = s + i  
    return s
```

Que retournent les appels `sommeListe([1, 3, 13])`, `sommeListe([1])` et `sommeListe([])` ? Pour vous aider, remplissez le tableau ci-dessous.

	étape 1	étape 2	étape 3	étape 4	étape 5	étape 6	étape 7	étape 8	étape 9
L	[1,3,13]								
i	-								
s	0								

L'instruction `return` termine l'exécution de la fonction qui la contient. Que calcule la fonction `sommeListe` si par malheur on indente trop la dernière ligne, comme ci-dessous ?

```
def sommeListe(L):  
    s = 0  
    for i in L:  
        s = s + i  
        return s # gros bug !
```

EXERCICE 5

1. En vous inspirant de la fonction `sommeListe` de l'exercice précédent, écrivez une fonction `sommeCarresListe(L)` calculant la somme des carrés des éléments de la liste `L`.
2. L'écart-type d'une liste de nombres `L` permet d'estimer dans quelle mesure les éléments de `L` s'éloignent de la moyenne des éléments de `L`. Par exemple l'écart-type de la liste `[8, 8, 8, 12, 12, 12]` est de 2 (puisque tous les éléments sont à distance 2 de la moyenne 10).

On peut le calculer en utilisant la somme des carrés des éléments de `L` et la somme des éléments de `L` (en notant `n` le nombre d'éléments de `L`) :

$$EcartType(L) = \sqrt{\frac{\sum_i L_i^2}{n} - \left(\frac{\sum_i L_i}{n}\right)^2}$$

Pour calculer une racine carrée, il faut ajouter `from math import *` au début du fichier pour avoir accès à la fonction `sqrt`. En utilisant en plus les fonctions `sommeListe` et `sommeCarresListe` écrites précédemment, écrivez une fonction `ecartTypeListe(L)` qui calcule l'écart-type de la liste `L`.



EXERCICE 6

Écrire les fonctions suivantes prenant en paramètre une liste de nombres `L`, et **testez** ces fonctions :

- une fonction `moyenneListe(L)` qui calcule et retourne la moyenne de ces nombres,
- une fonction `maximumListe(L)` qui calcule et retourne le maximum de ces nombres (supposés positifs).
- une fonction `nbPairsListe(L)` qui compte et retourne combien de ces nombres sont pairs.



EXERCICE 7

On considère les fonctions `existePairListe(L)`, qui renvoie `True` si au moins un des nombres de la liste est pair et `False` sinon, et `tousPairsListe(L)` qui renvoie `True` si tous les nombres de la liste sont pairs, et `False` sinon. Écrire ces deux fonctions en faisant en sorte qu'elles retournent leur résultat dès que celui-ci est déterminé. Testez ces fonctions.