

Méthode *Listes en Python*

- ☞ Une séquence ordonnée d'éléments peut être stockée sous la forme d'une liste en Python.

Par exemple on peut stocker les quatre premiers termes de la suite (f_n) de Fibonacci dans une liste `fibonacci`. f_0 sera stocké dans `fibonacci[0]`, f_1 dans `fibonacci[1]` ...

 On écrit `fibonacci[0]` et non pas `fibonacci(0)`.

Les éléments de la liste sont séparés par une virgule et la liste est délimitée par des crochets.

```
fibonacci = [1, 1, 2, 3, 6]
```

- ☞  Les indices commencent à 0 dans une liste. Le premier élément de la liste est donc `fibonacci[0]`. La fonction `len` renvoie la longueur de la liste. Le dernier élément est à l'indice `len(fibonacci) - 1` et sa valeur est `fibonacci[len(fibonacci) - 1]`.

- ☞ Pour accéder à f_2 , on écrit `fibonacci[2]`.

On s'est trompé sur la valeur de f_4 qui est $f_2 + f_3 = 5$. Pour corriger, on écrit `fibonacci[4] = 5`.

On aurait pu aussi écrire :

```
fibonacci[4] = fibonacci[2] + fibonacci[3]
```

- ☞ Pour ajouter $f_5 = f_3 + f_4 = 3 + 5 = 8$ à la liste `fibonacci`, on écrit :

```
fibonacci.append(8)
```

La valeur de `fibonacci` est désormais `[1, 1, 2, 3, 5, 8]`.

- ☞ Considérons désormais la suite u définie par $u_0 = 2$ et $u_{n+1} = 0,8u_n - 1$. Pour créer la liste `listu` des six premiers termes de u (de u_0 à u_5), on peut écrire :

```
u = 2
listu = [u]
for k in range(5):
    u = 0.8 * u - 1
    listu.append(u)
```

- ☞ Considérons la suite v définie pour tout entier naturel n par $v_n = f_n + u_n$. On peut créer la liste `listv`, des six premiers termes de v à partir des listes `fibonacci` et `listu` :

```
listv = []
for k in range(6):
    listv.append(fibonacci[k] + listu[k])
```

- ☞ Enfin si on veut définir la liste des carrés des entiers entre 0 et 4, on peut le faire avec une liste en compréhension :

```
carre = [k ** 2 for k in range(5)]
```

La valeur de carre sera [0, 1, 4, 9, 16].

On aurait aussi pu créer `listv` par compréhension :

```
listv = [fibonacci[k] + listu[k] for k in range(6)]
```

Exercice 1 Bac Polynésie mars 2023

On considère la fonction `mystere` définie ci-dessous qui prend une liste `L` de nombres en paramètre. On rappelle que `len(L)` représente la longueur de la liste `L`.

```
def mystere(L):  
    S = 0  
    for i in range(len(L)):  
        S = S + L[i]  
    return S / len(L)
```

Que renvoie l'exécution de `mystere([1, 9, 9, 5, 0, 3, 6, 12, 0, 5])` ?

Exercice 2 Bac Asie mai 2022

On s'intéresse au développement d'une bactérie.

Dans cet exercice, on modélise son développement avec les hypothèses suivantes : cette bactérie a une probabilité 0,3 de mourir sans descendance et une probabilité 0,7 de se diviser en deux bactéries filles.

Dans le cadre de cette expérience, on admet que les lois de reproduction des bactéries sont les mêmes pour toutes les générations de bactéries qu'elles soient mère ou fille.

Pour tout entier naturel n , on appelle p_n la probabilité d'obtenir au plus n descendances pour une bactérie.

On admet que, d'après ce modèle, la suite (p_n) est définie de la façon suivante :

$p_0 = 0,3$ et, pour tout entier naturel n ,

$$p_{n+1} = 0,3 + 0,7p_n^2.$$

La fonction suivante, écrite en langage Python, a pour objectif de renvoyer les n premiers termes de la suite (p_n) .

```
1 def suite(n) :  
2     p= ...  
3     s=[p]  
4     for i in range (...) :  
5         p=...  
6         s.append(...)  
7     return s
```

Compléter les lignes 2, 4, 5 et 6 de cette fonction pour que `suite (n)` retourne, sous forme de liste, les n premiers termes de la suite.