

Chapitre 8 : Tableaux et fonctions

Syntaxe

Une fonction peut prendre un tableau en paramètre. Cela permet d'appliquer un même traitement à plusieurs tableaux.

Par exemple :

```
In [1]: def premier(tab):  
        print("premier élément : " + str(tab[0]))  
  
        t = [34, 22, 6, 70]  
        premier(t)  
  
premier élément : 34
```

Une fonction peut aussi renvoyer un tableau :

```
In [2]: def cree_tab():  
        t = [1,2,3]  
        t.append(4)  
        return t  
  
        t = cree_tab()  
        print(t)  
  
[1, 2, 3, 4]
```

Fonctions avec parcours de tableaux

Très souvent les fonctions parcourent le tableau avec une boucle.

Exemples

- pour calculer une somme :

```
In [3]: def somme_tab(t):  
        somme = 0  
        i=0  
        while i < len(t):  
            somme += t[i]
```

```

        i+=1
    return somme

print(t)
print(somme_tab(t))

[1, 2, 3, 4]
10

```

- pour modifier le tableau :

```

In [4]: def mise_a_zero(t):
        i=0
        while i < len(t):
            t[i] = 0
            i+=1

        t = [1,1,1]
        mise_a_zero(t)
        print(t)

[0, 0, 0]

```

Dans une fonction, l'affectation globale d'un tableau crée une variable locale, détruite à la fin de l'exécution de la fonction :

```

In [1]: def affectation(t):
        t = [0,0,0]
        print("local : " + str(t))

        t = [1,1,1]
        print("avant l'appel : " + str(t))
        affectation(t)
        print("après l'appel : " + str(t))

avant l'appel : [1, 1, 1]
local : [0, 0, 0]
après l'appel : [1, 1, 1]

```

Le tableau passé en paramètre n'est donc pas modifié.