LE CONCEPT

On dit qu'un programme, une procédure ou une fonction est récursif s'il **s'appelle lui-même**. Ainsi pour résoudre un problème ou effectuer un calcul, on se ramène à la résolution d'un problème similaire mais de **complexité moindre.** On recommence ainsi jusqu'à obtenir **un problème élémentaire** que l'on sait résoudre.

Ex :

```
def somme2(n):
    """renvoie la somme des
    n premiers entiers"""
    if n == 1:
        return 1
    else:
        return n + somme2(n-1)
```

Inconvénient de la récursivité : Le plus gros inconvénient de la récursivité est qu'une fois la technique implémentée dans un langage de programmation, elle est très gourmande en ressource mémoire. Du fait que l'on empile les appels récursifs, **des débordements de capacité** peuvent se produire lorsque cette pile est pleine.
Python limite explicitement à 1000 le nombre d'appels récursifs dans une fonction.

DIFFÉRENTES RÉCURSIVITÉS

- **La récursivité croisée** fait référence **à une fonction qui appelle une autre fonction qui appelle elle-même la première.**
  Ex :

```
def pair(n):
    if n == 0:
        return True
    else:
        return impair(n-1)

def impair(n):
    if n == 0:
        return False
    else:
        return pair(n-1)
```

- Il existe des situations où la fonction **s'appelle plusieurs fois, on parle alors de récursivité multiple**.
  Ex :

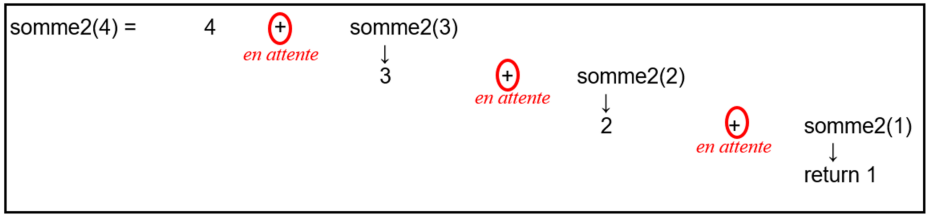
```
def fibonacci(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fibonacci(n-2) + fibonacci(n-1)
```

PRINCIPES GÉNÉRAUX

- Une fonction récursive doit **contenir une ou des conditions d'arrêt**. Sinon le programme boucle indéfiniment.
- **Les valeurs passées en paramètres dans les appels récursifs doivent être différentes**. Sinon la fonction s'exécute à chaque appel de manière identique et continue donc indéfiniment.
- Après **un nombre fini d'appels,** la ou les valeurs passées en paramètres doivent **satisfaire à la condition d'arrêt.**

Lorsqu'on souhaite écrire une fonction récursive, il faut donc avoir en tête la ou les conditions d'arrêt ainsi que la formule de récursivité.
Pour représenter la situation, on utilise **un arbre d'appels :**

Exemple avec `somme2(4)` :



LA RÉCURSIVITÉ
(Programmation S3)