LA RÉCURSIVITÉ : SYNTHÈSE DE COURS

1) Définition

> Une fonction récursive est une fonction qui s'appelle elle-même jusqu'à atteindre une condition d'arrêt (appelée aussi « cas de base » par opposition au « cas récursif »).

Ex : La suite de Fibonacci (0, 1, 1, 2, 3, 5, 8,… $u_n = u_{n-1} + u_{n-2}$,…)

```
def fibonacci(n):
    if n < 2:
        return n
    else:
        return fibonacci(n-1) + fibonacci(n-2)
```

Intérêt : Une fonction récursive permet donc, comme une boucle, de répéter des instructions. Ainsi, de nombreux algorithmes peuvent se programmer de façon itérative (avec des boucles) ou de façon récursive. Certains problèmes sont de nature récursive et se codent de façon récursive avec beaucoup de simplicité et d'élégance (Hanoï, utilisation de structures de données récursives...)
La programmation récursive est parfois gourmande en ressource car, à chaque appel récursif, il faut enregistrer le contexte d'exécution de la fonction en cours (valeurs des registres et des variables) dans une pile dont la taille est limitée.

2) Arbre des appels récursifs

Pour déterminer la valeur renvoyée par une fonction récursive, il est commode de dessiner l'arbre des appels récursifs :