**3 – Récursivité : principes et exemples**

1. **Prérequis**

Algorithme : Un algorithme est une suite finie et non ambiguë d’opérations ou d'instructions permettant de résoudre un problème ou d'obtenir un résultat. Il est dit correct lorsque, pour chaque instance du problème, il se termine en produisant la bonne sortie, c'est-à-dire qu'il résout le problème posé.

Fonction en programmation : En informatique, une fonction est une portion de code représentant un sous-programme, qui effectue une tâche ou un calcul relativement indépendant du reste du programme. Une fonction à une entrée (les arguments, ou paramètres qu'on lui donne), exécute un travail sur ces arguments, et, généralement, retourne une ou plusieurs valeurs (la sortie).

Une fonction peut donc comporter du code définissant un algorithme.

1. **Le principe de la récursivité**

Tout objet est dit récursif s’il se définit à partir de lui-même

1. Les structures récursives

Une structure est récursive si un de ses attributs est une autre instance de cette structure.

Exemple : les arbres

1. Les fonctions récursives

Ainsi, une fonction est dite récursive si elle comporte, dans son corps, au moins un appel à elle-même.

Condition(s) terminale(s) : elle doit se placer en début de fonction, c’est une condition particulière de la fonction dans laquelle il n’y aura pas d’appelle récursif, la chaine d’appels s’arrêtera.

Ex : fonction factorielle

Preuve qu’un algorithme récursif se termine, comme pour une boucle while, il faut trouver un variant ou convergent qui décroit fortement jusqu'à ce que la condition terminal soit vérifié. Exemple avec Syracuse qui se termine mais la preuve n’existe pas.

Il n’existe pas de programme vérifiant qu’un algorithme se termine sans le faire tourner.

Exemple d’un petit algo avec le chemin d’exécution/ pile et retour (fibo)

1. **Exemples**

Exemple d’algo plus efficace en itératif qu’en récursif : fibo + récursif efficace

Exemple d’algo plus facile à écrire en récursif : parcours en profondeur d’un arbre, Quicksort