Algorithmes récursifs

1. Définition

Un **algorithme récursif** résout un problème en calculant des solutions plus petites du même problème. L'approche récursive est un concept de base en informatique. Un algorithme récursif est un algorithme qui s'appelle lui-même.

Les premiers langages de programmation qui ont autorisé l'emploi de la récursivité sont **LISP** et **Algol 60**. Depuis, tous les langages de programmation généraux permettent la récursivité.

On oppose les algorithmes récursifs aux **algorithmes itératifs** qui s'exécutent sans s'appeler eux-mêmes.

2. Exemples

Exemple1 : la factorielle

Celle-ci se définit pour des entiers naturels de la fonction suivante :

$$n! = 1 \times 2 \times 3 \times \ldots \times (n-1) \times n$$
$$0! = 1$$

Autrement dit : $n\,! = (n-1)\,! \times n$ pour $n\,! \neq 0$ et $0! = 1$

Cette définition de la factorielle peut se traduire par le programme suivant en pseudo-code :

```
factorielle(n):
  si (n = 0) alors 1
  sinon n * factorielle(n-1)
```

Préciser que **`factorielle(0) = 1`** est fondamental : sans cela la fonction ne serait pas définie et l'algorithme ne se termine pas. Le cas $n = 0$ est appelé **cas de base**.

Exemple2 : suite de Fibonacci

Dans une suite de Fibonacci, chaque terme est la somme des deux précédents.

Version 1

```
fonction fibonacci(n, a, b)
     si n = 0 alors renvoyer 0
     sinon si n = 1 alors renvoyer 1
     sinon renvoyer fibo(n - 1) + fibo(n - 2)
```

Version 2

```
fonction fibonacci(n, a, b)
      si n = 0 alors retourner a
      sinon si n = 1 alors retourner b
      sinon retourner fibonacci(n-1, b, a+b)
```

Exemple2 : suites de Syracuse

```
syracuse(n):
  si (n = 0) ou (n = 1) alors 1
  sinon si (n mod 2 = 0) alors syracuse(n/2)
  sinon syracuse(3*n + 1)
```

3. Fractales

Visionnez la vidéo consacrée au flocon de Koch :