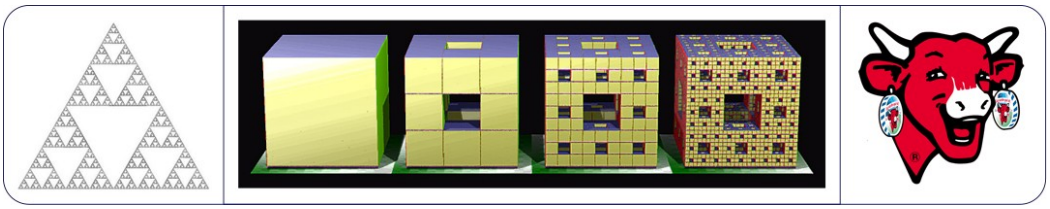
Fonctions récursives

On dit qu'un programme est récursif si pour parvenir au résultat voulu il se réemploie lui-même. Cela peut s'illustrer par les images suivantes :



1. Principe de base :

Une fonction récursive simple s'écrit sous la forme :

```
def fonctionRecursive(args)
        if condition arret:
                return valeur
        fonctionRecursive(argument)
```

Pour écrire une fonction récursive, il faut :

- ➔ déterminer le type de données à renvoyer
- ➔ écrire la condition d'arrêt (appelé aussi cas de base) pour interdire une récursivité infinie
- ➔ écrire l'appel récursif

2.Intérêt de la récursivité

La récursivité fournit des algorithmes concis et élégants. Il s'agit à la fois d'un style de programmation mais également d'une technique pour définir des concepts et résoudre certains problèmes qu'il n'est parfois pas facile de traiter en programmant uniquement avec des boucles.
Il faut cependant ce méfier de la complexité, de plus il faut bien s'assurer de définir un cas d'arrêt pour interdire une récursivité infinie

3.Gestion de la mémoire lors d'un appel récursif

Pour chaque appel de fonction, les données (arguments de la fonction, place pour la valeur de retour, adresse de retour, les variables locales) sont stockés dans un espace réservé de la mémoire appelé la « pile ». Cette pile est une structure de données linéaire ayant pour maxime « dernier entré, premier sorti » (last in, first out – LIFO)

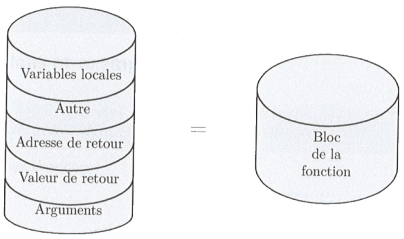
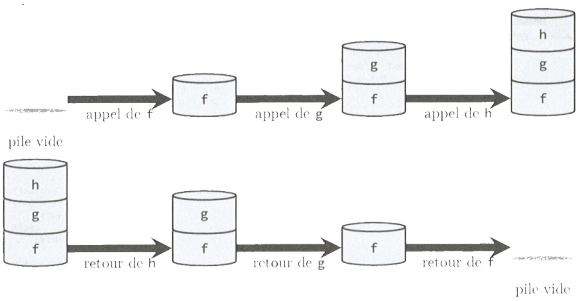


Figure 1: Pile LIFO

Gestion des appels de fonctions

Considérons l'appel des fonctions suivantes :

```
def h(x):
        return x+1
def g(x):
        return h(x)+2
def f(x):
        return g(x)+1
```



L'appel de f(5) entraine le mécanisme suivant dans la pile

Pile d'appel sous Python

Lors de l'appel d'une fonction récursive une pile est utilisée. En Python, la taille de la pile est limitée à 1000 par défaut. En cas de dépassement le message d'erreur suivant s'affiche :

```
RuntimError : maximum recursion depth exceeded
```

Types de récursivité

Il existe plusieurs types de récursivité :

- ➔ **récursivité simple** : un algorithme récursif est simple ou linéaire si chaque cas se résout en au plus un appel récursif. Le calcul de la factorielle en est un exemple.
- ➔ **récursivité multiple** : un algorithme récursif est multiple si l'un des cas qu'il distingue se résout avec plusieurs appels récursifs. Si "plusieurs=2" on parle de récursivité binaire.
- ➔ **récursivité terminale** : un algorithme est récursif terminal si l'appel récursif est la dernière instruction de la fonction. La valeur retournée est directement obtenue par un appel récursif.