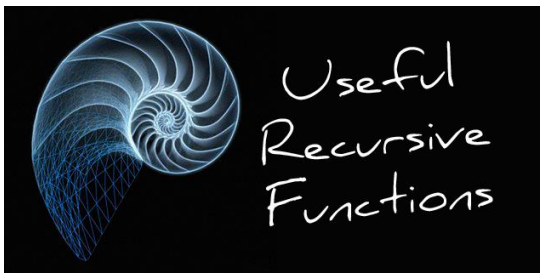
1 Récursivité : Introduction



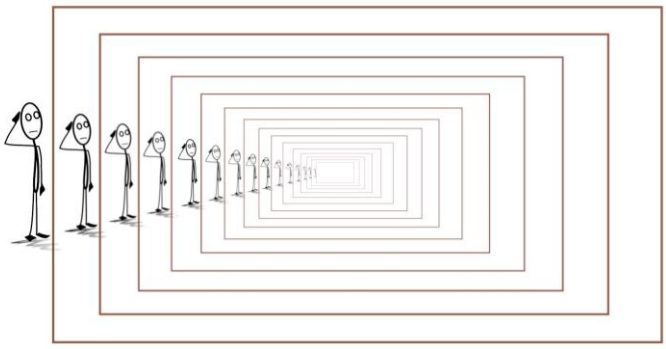
Visionner la vidéo d'introduction :

L'approche **récursive** est un des concepts de base en informatique. Les premiers langages de programmation qui ont autorisé l'emploi de la **récursivité** sont LISP et Algol 60.

Cette approche de programmation permet de résoudre parfois élégamment et en peu de lignes des problèmes qui pourraient être complexes en version itérative.

2 Définitions, déroulement d'un algorithme récursif

Le principe d'un algorithme récursif, pour résoudre un problème, est de le ramener à un problème similaire mais de complexité moindre. On recommence ainsi jusqu'à arriver à un problème élémentaire que l'on sait résoudre.



Pour réaliser cela, **une fonction récursive va s'appeler elle-même**, avec un paramètre "plus petit". Cet appel en induira un autre, puis un autre, … et ainsi de suite jusqu'à que le problème soit immédiatement résolvable.

Définitions : Une fonction (ou algorithme) est dite **récursive si elle s'appelle elle-même**.

Important : Une fonction (ou algorithme) récursive **doit toujours avoir une condition d'arrêt**, sinon elle va s'appeler une infinité de fois.

Remarques :

- La plupart des algorithmes itératifs peuvent se transformer en algorithmes récursifs
- La méthode récursive est **gourmande en mémoire**. En effet, tous les appels successifs sont stockés en mémoire (empilés), un débordement de mémoire peut survenir et engendrer un arrêt de l'exécution du programme lorsque cette pile est pleine.
- Python limite la profondeur de récursivité (nombre d'appels récursifs), pour connaitre le nombre maximum d'appels :

```
import sys
print (sys.getrecursionlimit())
```

Exemple : La fonction mathématique factorielle : 5! = 1*2*3*4*5 = 120

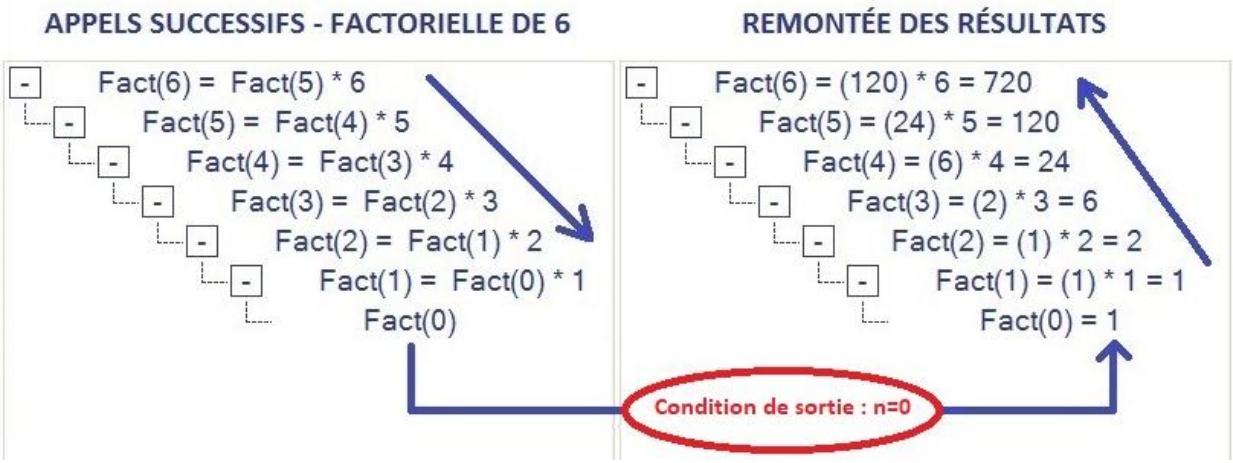
Version itérative

```
def factorielle(n) :
    resultat = 1
    for i in range (1,n+1) :
        resultat*=i
    return resultat
```

Version récursive

```
def factorielle(n) :
    if n==0 :
        return 1          } condition d'arrêt
    else :
        return n*factorielle(n-1)   } appels récursifs
```

Tester ces programmes ici : *factorielle.py*



3 Complexité d'un algorithme récursif

La complexité d'un algorithme récursif est soit linéaire soit exponentielle. Par exemple, la complexité de l'algorithme factoriel ci-dessus est linéaire : en O(n) (notation Landau).

Vous pouvez tenter de le redémontrer… (Ce n'est pas compliqué).