Une **fonction récursive** en informatique est une fonction qui s'appelle elle-même.
On propose dans ce TP plusieurs exercices faisant intervenir des fonctions récursives.

1) La fonction factorielle

Exemple par excellence de fonction récursive, on peut définir la factorielle d'un entier n à partir de la factorielle de l'entier précédent : $\forall n \in \mathbf{N}^\star,\ n! = n \times (n-1)!$
Il faut donc définir le *cas de base* : $0\,! = 1$, puis appeler la fonction factorielle en l'entier $n-1$.

Compléter le script suivant, qui définit la fonction factorielle :

```
def factorielle(n) :
      if n == 0 : return ........
      return n * factorielle(........)
```

Écrire une fonction arithm de paramètre u0, n et R qui renvoie le nième terme d'une suite arithmétique de premier terme u0 et de raison R. On écrira trois fonctions différentes (une directe, une itérative et une récursive).
Réaliser le même exercice pour une suite géométrique.

2) Une suite récurrente d'ordre 1

On peut définir de façon récursive toute suite (u_n) telle que : $\begin{cases} u_0 \quad \text{est donné} \\ u_{n+1} = f(u_n) \quad \text{avec } f \text{ une fonction donnée} \end{cases}$

Écrire une fonction `U` qui renvoie de façon récursive la valeur u_n de la suite définie par : $\begin{cases} u_0 = 3 \\ \forall n \geqslant 0,\ u_{n+1} = 5 + \dfrac{3}{u_n} \end{cases}$

Indication : la relation de récurrence s'écrit aussi $\forall n \geqslant 1,\ u_n = 5 + \dfrac{3}{u_{n-1}}$.

3) Une suite récurrente d'ordre 2 : la suite de Fibonacci

Pour une suite récurrente d'ordre 2, le cas de base englobe deux valeurs de n.

Écrire une fonction renvoyant le terme d'indice n de la suite de Fibonacci : $\begin{cases} f_0 = 0,\ f_1 = 1 \\ \forall n \geqslant 0,\ f_{n+2} = f_{n+1} + f_n \end{cases}$

4) L'algorithme de Syracuse

Il est possible que l'appel à la fonction récursive soit soumis à un test.
On rappelle l'algorithme de Syracuse : si $n \in \mathbf{N}^\star$ est pair, on le divise par 2, et sinon on renvoie $3n+1$.
La conjecture de Syracuse énonce que pour tout $n \in \mathbf{N}^\star$, après un nombre fini d'étapes, on obtient 1.

Écrire une fonction d'argument $n \in \mathbf{N}^\star$ sur lequel opère de façon répétée l'algorithme de Syracuse, et qui renvoie la liste de toutes les valeurs prises jusqu'à obtenir la valeur 1.

Par exemple `>>> syracuse(7)` devra renvoyer la liste : $[7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1]$.

5) Une fonction récursive à 2 variables : le PGCD

Le plus grand diviseur commun (PGCD) de deux entiers $p, q \in \mathbf{N}$ s'obtient grâce à l'algorithme suivant :

- si $p = 0$, alors $PGCD(p, q) = q$,
- si $q = 0$, alors $PGCD(p, q) = p$,
- si $p \geqslant q$, alors $PGCD(p, q) = PGCD(p-q, q)$,
- si $q > p$, alors $PGCD(p, q) = PGCD(q-p, p)$.

Écrire une fonction `PGCD` de deux variables p, q et renvoyant le $PGCD$ de p et q.

6) Des permutations

On considère une liste L contenant des lettres distinctes, par exemple : `L = ['a', 'b', 'c']`.
On veut écrire une fonction renvoyant une liste formée par tous les mots obtenus en permutant les lettres de `L` : ici `['abc', 'acb', 'bac', 'bca', 'cab', 'cba']`.
Le cas de base correspond alors à une liste `L` de longueur 1 : on renvoie simplement la liste `L`.
Si `len(L) > 1`, on constitue une liste `M` formée par toutes les permutations obtenues à partir de la liste `L` privée de son dernier élément (par l'appel récursif à une fonction), puis on insère à toutes les places possibles cette dernière lettre dans les mots de la liste `M`.

Écrire une fonction répondant au problème (la liste obtenue ne sera pas forcément classée par ordre alphabétique).

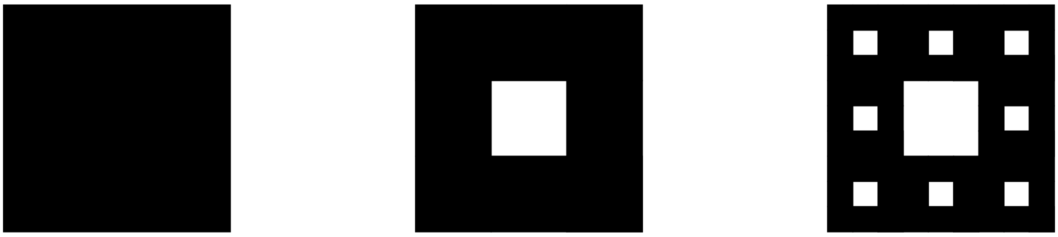
7) Engendrer des mots de longueur donnée

Écrire de façon récursive une fonction prenant pour arguments un entier n et une liste de lettres L, et renvoyant tous les mots de longueur n utilisant les lettres de la liste L.
Par exemple, cette fonction appliquée à $n = 2$ et à la liste `L = ['a', 'b', 'c']` devra renvoyer la liste :
`['aa', 'ab', 'ac', 'ba', 'bb', 'bc', 'ca', 'cb', 'cc']`

8) Une figure fractale

Le tapis de Sierpiński (1882-1969) est une figure obtenue, à l'ordre $n \in \mathbf{N}^\star$, en découpant un carré en 9 carrés égaux, en supprimant le carré central, et en réitérant $(n-1)$ fois le procédé aux 8 carrés restants.

$n = 0$ $\quad$ $n = 1$ $\quad$ $n = 2$



On peut créer une variable `PIL` définissant le tapis de Sierpiński d'ordre n en partant pour $n = 0$ d'une image formée d'un unique pixel noir, puis à l'ordre suivant en définissant une image contenant 3 fois plus de colonnes et de lignes, et en recopiant l'image précédente 8 fois (partout sauf dans le carré central).

Écrire une fonction répondant au problème.

Indication : le carré central est caractérisé, si d est la dimension de l'image précédente, par le fait que les indices i, j définissant la position d'un pixel vérifient : `i//d == 1 and j//d == 1`.