TP 7 Récursivité

D'après wikipedia : Un algorithme récursif est un algorithme qui résout un problème en calculant des solutions d'instances plus petites du même problème. L'approche récursive est un des concepts de base en informatique. ... Pour répéter des opérations, typiquement, un algorithme récursif s'appelle lui-même.

On a déjà vu que lorsqu'on avait définit une fonction f, on pouvait appeler f à l'infini que ce soit dans la console ou bien dans d'autres fonctions. Dans ce TP, on va programmer des fonctions qui s'appelent elles meme.

Définition : Une fonction récursive est une fonction qui s'appelle elle-même.

Exercice 1 (Un premier exemple mathématique)

1. Calculer 5!, 6! et 7!
2. Rappeler la formule qui lie $(n+1)!$ à $n!$.
3. Ecrire une fonction prenant en entrée un entier n et renvoyant l'entier $n!$. On utilisera ni boucle for, ni boucle while mais on écrira une fonction récursive.

Comment ça marche pour $n = 4$?

Appel à fact(4) .

4*fact(3) = ?

Appel à fact(3)

3*fact(2) = ?

Appel à fact(2)

2*fact(1) = ?

Appel à fact(1)

1*fact(0) = ?

Appel à fact(0)

Retour de la valeur 1

1*1

Retour de la valeur 1

2*1

Retour de la valeur 2

3*2 .

Retour de la valeur 6

4*6

Retour de la valeur 24

Exercice 2 (Un peu de dessin aplphanumérique)

1. Ecrire une fonction ligne prenant en entrée un entier n et qui dessine (avec un print) une ligne de n symboles * .
2. Ecrire une fonction récursive prenant en entrée un entier n et qui dessine un triangle de la forme :

```
* * * * *
* * * *
* * *
* *
*
```

Exercice 3 (L'exponentiation rapide)

Rappel : L'algorithme d'exponentiation rapide est un algorithme extrêmement efficace permettant de calculer x^n. Il est basé sur la formule :

$$x^n = \begin{cases} x^p.x^p & \text{si } n = 2p \\ x^p.x^p.x & \text{si } n = 2p+1 \\ 1 & \text{si } n = 0 \end{cases}$$

Utiliser cette méthode afin d'écrire une fonction récursive qui prend en entrée un nombre x et un entier n et qui renvoie le nombre x^n.

Exercice 4 Une fractale

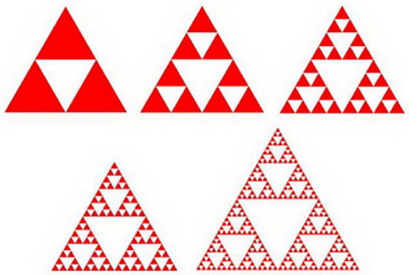
Vous pouvez importer le module turtle en tapant le code suivant :

```
from turtle import *
```

On controle un curseur via différentes fonctions, qui dessine un trait là où il passe. Les fonctions dont nous aurons besoin sont :

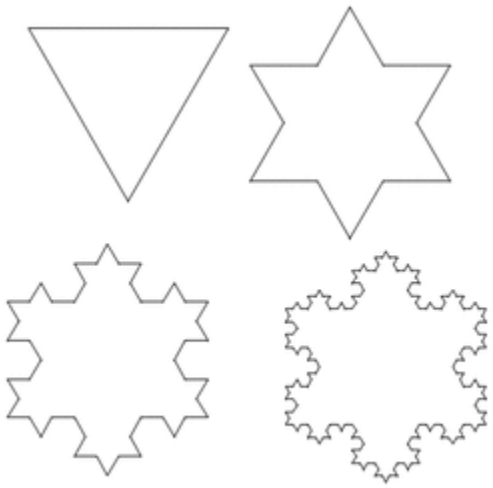
- forward (longueur) : le curseur avant de longueur (dans la direction déterminée par son angle)
- left(angle) : tourne vers la gauche de angle (en degré)
- right(angle) : tourne vers la gauche de angle (en degré).

1. Ecrire une fonction prenant en entrée un entier l et qui dessine un triangle équilatéral de coté l.
2. Ecrire une fonction récursive (prenant en entrée l et n) qui dessine le triangle de Sierpinsky à l'étape n. Ci joint les premières etapes de la construction (on ne cherche pas à colorier l'intérieur des triangles mais juste à tracer les contours) :



Exercice 5 : Fractale de Von Koch

Ecrire une fonction récursive qui prend en entrée une longueur l et un entier n et qui dessine le flocon de Vonkoch à l'étape n :



Exercice 6 (pour aller plus loin) : le tri par fusion

Le principe du tri par fusion est simple.

- Si une liste est vide ou contient un seul élément alors elle est triée.
- Sinon on trie la première moitié de la liste, la seconde moitié et on recolle les deux morceaux triés.

1. Ecrire une fonction qui prend en entrée deux listes triées l_1 et l_2 et renvoie la liste triée contenant les valeurs de l_1 et de l_2 fusionnées.[1]
2. Déduire un algorithme récursif permettant de trier une liste.
3. Evaluer la complexité d'un tel programme. Pourquoi est-il meilleur que tous les algorithmes de tri que nous avons déjà vus ?

[1] une même valeur peut apparaitre plusieurs fois dans les listes de départ où d'arrivée