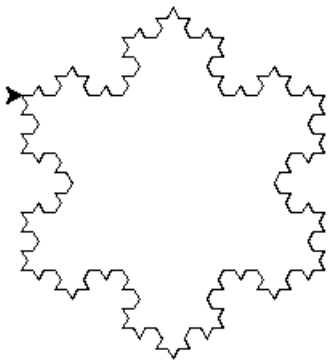
TP 11
Algorithmes récursifs

Dans la première partie le but est de tracer le **flocon de Von Koch**, une courbe qui est obtenue par étapes successives en partant d'une ligne droite puis en la coupant en trois segments égaux et en remplaçant le segment central par un triangle équilatéral : d'abord on trace la **courbe de Von Koch** à l'étape n
Étape $n = 0$:

Étape $n = 1$:

Étape $n = 2$:

Puis le flocon complet :



I Préliminaire : la tortue

Nous allons apprendre à tracer quelques graphiques intéressants.
Pour commencer on chargera le module `turtle` avec cette ligne **au tout début du fichier** et à exécuter une seule fois :

```
import turtle
```

Tester ensuite directement (mode script, dans une cellule, puis exécuter) les lignes suivantes pour vérifier que tout fonctionne correctement :

```
##
turtle.reset()
turtle.forward(200)
turtle.done()
##
```

Explications : le module `turtle` (*tortue* en français) permet de tracer des graphiques de la façon suivante. On imagine que les commandes du programme sont en fait des ordres donnés à une tortue qui tient un crayon, et à qui on peut dire d'avancer d'un certain nombre de pas, de tourner à droite ou à gauche d'un certain angle, etc. Cela est souvent utilisé pour apprendre la programmation. Le bout de programme ci-dessus demande d'avancer de 200 pas (ajuster ce nombre chacun sur son ordinateur selon la taille de la fenêtre) puis d'afficher la fenêtre.
Les commandes suivantes sont disponibles, aussi décrites dans la documentation officielle (pour les courageux) ou bien dans cet aide-mémoire (pour les enfants) :

- `turtle.reset()` : ré-initialise la fenêtre et la tortue à son point de départ, première instruction du programme
- `turtle.done()` : affiche la fenêtre et attend que l'utilisateur la ferme, dernière instruction du programme
- `turtle.forward(n)` : avance de `n` pas
- `turtle.left(r)` : tourne à gauche de l'angle `r` exprimé en degrés
- `turtle.right(r)` : idem mais tourne à droite
- … c'est tout ce dont nous aurons besoin, à vous de jouer à changer la couleur et l'épaisseur du trait etc.

D'autres commandes permettent notamment de « lever le pinceau » (et donc avancer sans tracer de trait) et le rabaisser et changer tout plein de paramètres graphiques.

Exercice 1. Échauffement : tracer un carré.

Exercice 2. Tracer un triangle équilatéral.

Exercice 3. Tracer la première étape ($n = 1$) d'une courbe de Von Koch.

II Le flocon de Von Koch

Pour tracer un flocon de Von Koch, on va d'abord tracer la courbe de Von Koch et écrire une fonction récursive qui prend deux arguments :

1. `n` : le numéro de l'étape qu'on est en train de tracer. Pour $n = 0$ la courbe est une simple ligne droite, pour $n = 1$ c'est la ligne polygonale de 4 morceaux tracée précédemment.
2. `L` : un paramètre qui indique la longueur du morceau que l'on est en train de tracer, et qu'il faut diviser par 3 dans les appels récursifs.

Exercice 4. Écrire une fonction `vonkoch(n, L)` récursive qui trace la courbe de Von Koch : pour $n = 0$ elle trace une ligne droite de longueur `L`, et sinon elle s'appelle récursivement en divisant la longueur par 3 et entre les appels récursifs la tortue doit tourner comme dans la section précédente.

Pour l'appeler proprement, on écrira alors (directement dans une cellule à part) :

```
##
turtle.reset()
vonkoch(n, L)
turtle.done()
##
```

Pour dessiner le flocon complet sous forme hexagonale, il suffit… d'appeler plusieurs fois la fonction précédente, en tournant entre temps d'un angle approprié.

Exercice 5. Écrire une fonction `flocon(n, L)` qui trace le flocon de Von Koch complet avec n étapes (ceci n'est plus une fonction récursive).

Pour que ce soit plus joli, il faut lire la documentation !

Exercice 6. Lire la documentation ou l'aide mémoire pour ajuster la couleur, la vitesse du tracé, le titre de la fenêtre, et pour bien centrer la figure.

III Reprise du TP précédent

Reprendre les exercices du TP précédént, notamment :

Exercice 7. On rappelle la formule de Pascal pour les coefficients binomiaux :

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k} \tag{1}$$

démontrée par le calcul sous la condition $1 \leqslant k \leqslant n-1$ (sinon, l'un des deux termes voire les deux sont nuls, et il est facile de vérifier que la relation est quand même valable).
Écrire une fonction récursive `binome(n)` qui renvoie pour tout $n \geqslant 0$ la *liste* des $n+1$ coefficients binomiaux $\binom{n}{k}$ pour $0 \leqslant k \leqslant n$. Autrement dit c'est la fonction qui calcule d'un coup toute une ligne du triangle de Pascal en fonction uniquement de la ligne précédente. Attention aux bords du triangle !

Exercice 8. Écrire une fonction récursive `parties(n)` qui retourne la liste de toutes les parties de l'ensemble $[\![1, n]\!]$ (une liste de listes, donc), en formulant d'abord le problème correctement de façon récursive.