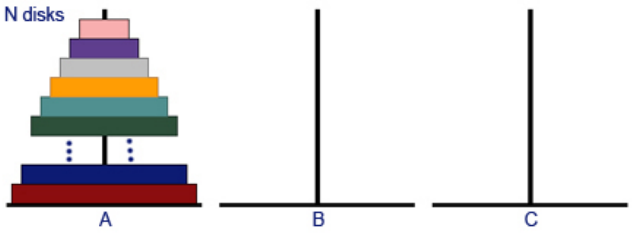
LABORATOIRE 14 | *Informatique 1*
RECURSIVITE

Hes·so VALAIS WALLIS
Haute Ecole d'Ingénierie
Hochschule für Ingenieurwissenschaften

But du laboratoire (6 périodes)

1. Le but de ce laboratoire est de vous permettre de vous exercer avec le concept de récursivité et d'illustrer comment il peut être très utile, d'abord dans un cas simple puis dans un cas plus complet avec des graphiques.
2. La durée estimée pour réaliser ce laboratoire est de **six périodes**. Si le temps imparti ne devait pas suffire, vous êtes invités à terminer le travail en dehors des heures de cours.
3. Vous pouvez trouver cette donnée sous forme électronique sur *http://inf1.begincoding.net* dans le thème à la rubrique *Labo 14*. Vous y trouverez également le corrigé de ce labo dans quelque temps.

Partie 1 – Tours de Hanoï



Le jeu des Tours de Hanoï consiste à déplacer n disques d'un poteau A vers un poteau C en s'aidant d'un poteau B. Les règles sont les suivantes :

- On ne peut bouger qu'un disque à la fois
- On ne peut que déplacer les disques se trouvant au sommet d'une pile
- On ne doit jamais mettre un disque plus grand sur un disque plus petit

C'est un exemple typique de problème pouvant se résoudre à l'aide de récursion. L'idée de la solution est la suivante si l'on souhaite déplacer les disques de A (départ) vers C (final) :

- S'il n'y a aucun disque à déplacer, on arrête la récursion. Sinon :
  - On déplace récursivement les n – 1 disques du poteau A vers le poteau B (nommé auxiliaire) en s'aidant de C.
  - On déplace le dernier disque de A (départ) vers C (final)
  - On déplace récursivement les n – 1 disques du poteau B vers le poteau C en s'aidant de A

Tâche 1

1) Implémentez le code des tours de Hanoi en texte d'abord. Le prototype de votre fonction est le suivant :

```
public static void hanoi(int n, char start, char aux, char end)
```

2) Vérifiez le fonctionnement. Pour 3 disques de A vers C en s'aidant de B, on obtient

```
Moving disk 1 from a->c
Moving disk 2 from a->b
Moving disk 1 from c->b
Moving disk 3 from a->c
Moving disk 1 from b->a
Moving disk 2 from b->c
Moving disk 1 from a->c
```

Partie 2 - Prise en mains de Logo

Nous introduisons maintenant un nouveau paquetage graphique que vous utiliserez dans la dernière partie de ce laboratoire. Pour ce faire, vous allez utiliser une classe nommée `TurtleGraphics`.

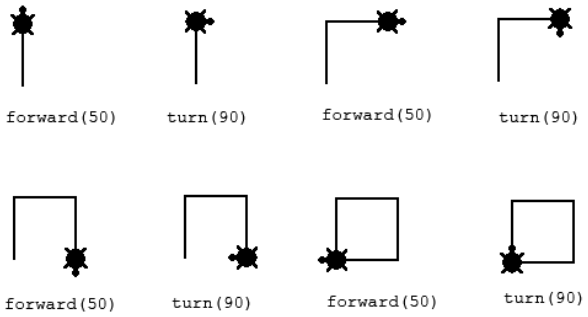


Figure 1 - Mouvement de la tortue (dessin adapté de *The Logo Foundation*)

Comme illustré ci-dessus, cette classe vous offre une abstraction correspondant à une petite tortue qui se déplace sur l'écran et qui peut dessiner sur son chemin. En plus des méthodes de la classe `FunGraphics`, la classe `TurtleGraphics` vous offre les méthodes suivantes :

- `forward(double n)` : fait avancer la tortue de n unités dans la direction actuelle
- `turn(double alpha)` : fait tourner la tortue d'un angle alpha
- `penDown()` : baisse le crayon
- `penUp()` : lève le crayon
- `changeColor(Color)` : change la couleur du trait
- `jump(int, int)` : saute à position X, Y sans dessiner dans l'intervalle

En plus de ces méthodes de base, vous pouvez également récupérer les données de la tortue avec les méthodes :

- `getPosition()` : position actuelle de la tortue
- `getTurtleAngle()` : angle actuel de la tortue
- `setAngle(double)` : tourner la tortue dans une direction donnée

Selon l'application, vous pourrez également utiliser les versions spécifiant les angles en radiants plutôt qu'en degrés :

- `getTurtleAngleRad()` : obtenir l'angle actuel en radians
- `turnRad(double)` : tourner de n radians
- `setAngleRad(double)` : mettre l'angle actuel à n radians

Notez que pour cette partie, il n'y a pas besoin d'utiliser la récursivité !

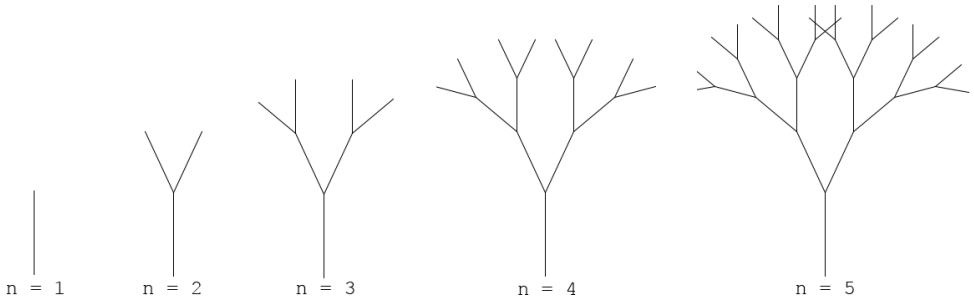
Tâche 2

1) Pour vous faire la main avec cette classe, créez une classe nommée `TestLogo`.
2) Dans cette classe, dessinez un triangle équilatéral avec les méthodes `forward` et `turn`.
3) Rajoutez une méthode permettant de réaliser le dessin d'un **cercle** réalisé avec la tortue. Cette méthode doit accepter un paramètre permettant de spécifier la grandeur du cercle. Essayer de trouver à partir du tableau suivant comment faire :

Triangle	Square	Pentagon
`forward 100`	`forward 90`	`forward 80`
`right 120`	`right 90`	`right 72`
`forward 100`	`forward 90`	`forward 80`
`right 120`	`right 90`	`right 72`
`forward 100`	`forward 90`	`forward 80`
`right 120`	`right 90`	`right 72`
	`forward 90`	`forward 80`
	`right 90`	`right 72`
		`forward 80`
		`right 72`

Partie 3 - Arbres récursifs

Dans cette partie le but est de dessiner des arbres avec des méthodes récursives. Le résultat attendu est le suivant :



Pour réaliser de tels arbres, vous devez écrire une méthode récursive dont le pseudo-code est le suivant (`n` correspond au niveau de récursivité et `length` à la taille du sous-arbre) :

```
public void drawTree(int n, double length)
   if(n > 1){
         drawBranch(length);

         Save turtle position and angle

         // Left subtree
         Draw the left tree, with n = n - 1

         Restore position and angle of turtle

         // Right subtree
         Draw the right tree, with n = n - 1
   }
   else{
         drawBranch(length)
   }
}
```

Tâche 3 - Dessin d'arbre récursif de base

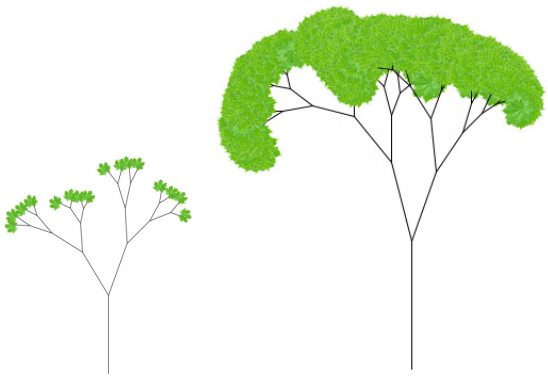
1) Écrivez la méthode `void drawBranch(double length)` qui sera utilisée pour dessiner une branche.
2) Écrivez la méthode `void drawTree`.
3) Testez votre méthode.

Tâche 4 - Dessin avancé

Dans cette tâche, nous vous proposons de rendre le dessin de votre arbre un peu moins "synthétique". Pour ce faire, plusieurs améliorations sont possibles :

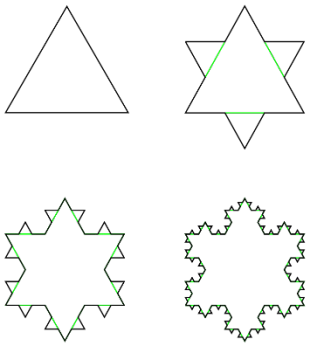
1) Rendre la longueur des segments aléatoires
2) Rendre l'angle de rotation des branches aléatoire
3) Dessiner des feuilles
4) Prenez des captures d'écran et mettez-les sur Teams dans le canal *Trees*, cela fera une jolie galerie !

Vous pouvez choisir de réaliser une ou plusieurs de ces améliorations. Un exemple possible de réalisation est montré sur la figure ci-dessous :



Partie 4 – Un peu de neige

Dans cette tâche, vous devrez développer vous-même l'algorithme pour dessiner un flocon de neige. Graphiquement, voici les différentes étapes nécessaires :



L'idée est la suivante : vous devez dessiner trois segments d'un triangle équilatéral. Pour chaque niveau de détail supplémentaire, divisez le segment en trois parties égales. La partie centrale du segment ainsi découpé devient la base d'un nouveau triangle équilatéral pointant vers l'extérieur. Quelques remarques pour vous aider :

1) Commencez uniquement par dessiner l'un des côtés du triangle (avec la récursivité). Cela donnera pour le niveau 1 une ligne, pour le niveau 2 une ligne avec un triangle sans base au milieu etc…
2) Répétez l'étape une en tournant à chaque fois de 60°. Vous aurez ainsi les dessins faits ci-dessus.
3) Dessinez le flocon pour un niveau 5.