

## Dessins de fractales avec Bob la tortue

Le but de ce TP est de dessiner des fractales dans le plan. On considérera que le plan est orienté de façon usuelle comme indiqué sur la Figure 1.

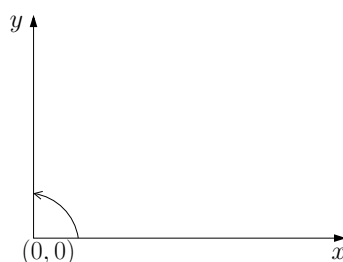


Figure 1: Orientation du plan

Pour effectuer des tracés dans le plan, on dispose de Bob : une tortue munie d'un crayon scotché sous son ventre, la pointe du crayon vers le sol. A tout moment, Bob est caractérisée par son abscisse, son ordonnée, sa direction (c'est à dire l'angle dans le sens trigonométrique entre la direction de l'axe des abscisses (de 0 vers  $\infty$ )) et la direction dans laquelle elle regarde, et la position du crayon (le crayon peut toucher le sol ou être relevé). On peut manipuler Bob grâce aux méthodes suivantes :

- `Turtle bob = new Turtle();` //créer Bob en position  $(0, 0)$ , avec l'angle 0 (Bob regarde donc vers la droite), et le crayon qui touche le sol (et ouvre une fenêtre graphique)
- `bob.forward(double d)` //fait avancer Bob vers là où elle regarde d'une distance  $d$
- `bob.left(double a)` //fait tourner Bob sur elle-même de  $a$  degrés vers la gauche
- `bob.right(double a)` //fait tourner Bob sur elle-même de  $a$  degrés vers la droite
- `bob.down()` //fait descendre le crayon pour qu'il touche le sol
- `bob.up()` //fait remonter le crayon pour qu'il ne touche plus le sol
- `bob.clear()` //efface l'écran
- `bob.speed(int v)` //modifie la vitesse de Bob (attention, plus  $v$  est grand, plus bob est lente)

Dans tout ce TP, les spécifications sont volontairement incomplètes : à vous de les compléter pour éviter les mauvaises surprises (en particulier, définir un prérequis sur la direction de Bob avant de commencer le tracé, et une spécification sur sa direction après le tracé).

Pour tous les exercices, il est interdit d'écrire une méthode auxiliaire.

### Exercice 1. Echauffement

#### Question 1.1.

Nous allons commencer par simplement dessiner un carré. Placez dans un même dossier les fichiers `Turtle.java` et `DessinFractale.java` disponibles sur l'ent. Tout le travail de ce TP se fera dans `DessinFractale.java`. Complétez la méthode `public void carre(double l)` pour qu'elle dessine un carré de côté  $l$  (cette spécification est volontairement incomplète, décidez vous même où le carré doit être tracé!).

## Exercice 2. Flocon

### Question 2.1.

Ecrire une méthode `public void vonKoch(double l, int n)` qui trace un flocon de Von Koch de taille  $l$  et d'ordre  $n \geq 0$  (voir Figure 2).



Figure 2: Exemples de flocons de Von Koch de taille  $l$  et d'ordre 0 (à gauche), 1 (au milieu) et 2 (à droite).

## Exercice 3. Arbre

### Question 3.1.

Ecrire une méthode pour tracer des arbres semblables à ceux dessinés Figure 3. Essayez de varier l'angle des branches!

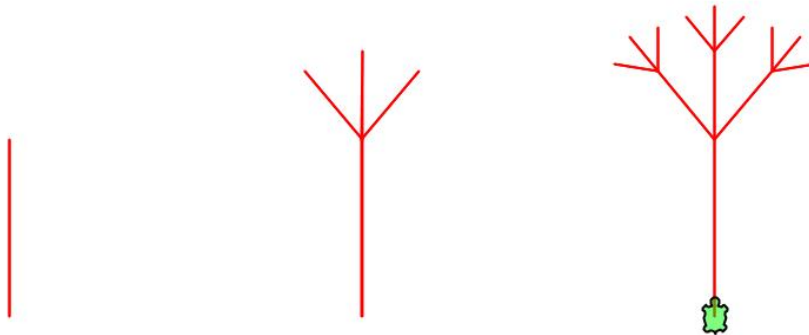


Figure 3: Exemples d'arbre d'ordre 0 (à gauche), 1 (au milieu) et 2 (à droite).

## Exercice 4. Triforce (ou presque) (Sierpinski)

### Question 4.1.

Ecrire une méthode pour tracer des figures semblables à celles dessinées Figure 4 (la taille totale du triangle englobant ne varie pas).

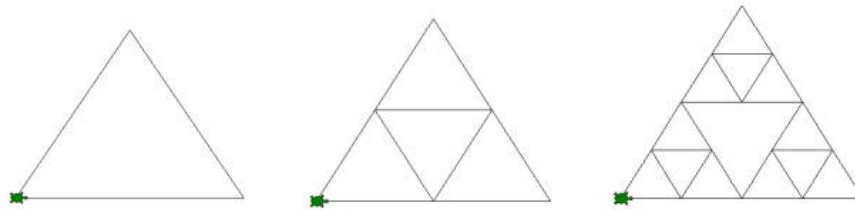


Figure 4: Exemples de Sierpinski à l'ordre 0, 1 et 2.

## Exercices bonus

### Exercice 5. Dragon

#### Question 5.1.

Ecrire une méthode pour tracer des dragons à ceux dessinés Figure 5 (la taille des tracés n'est pas importante ici). Vous pouvez utiliser des méthodes auxiliaires, ou ayant des paramètres supplémentaires.

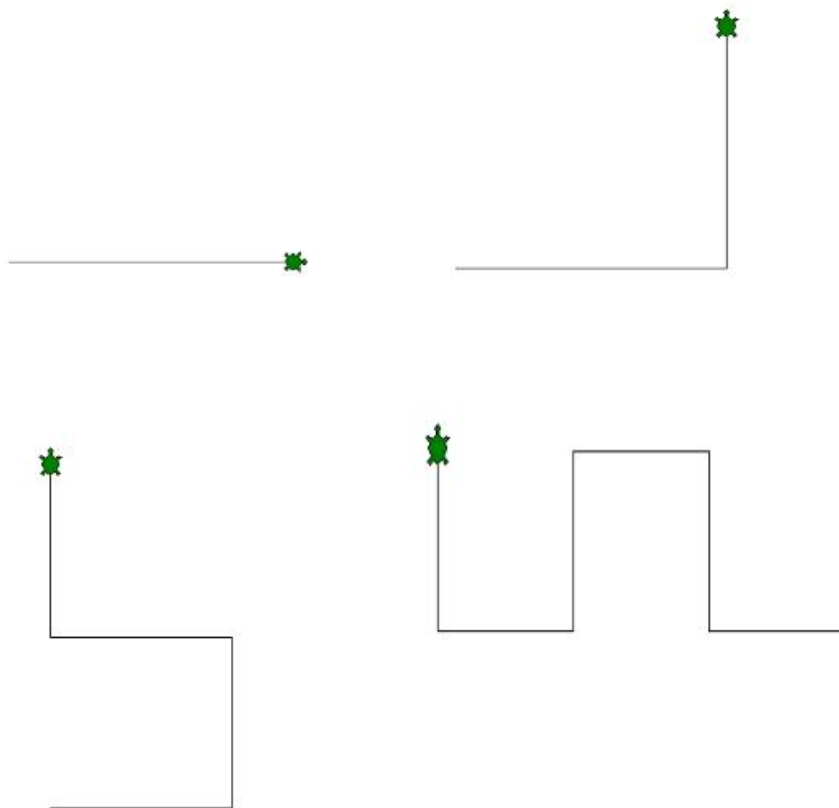


Figure 5: Exemples de dragons d'ordre 0, 1, 2 et 3.