Le module Turtle

La bibliothèque **Turtle** permet de réaliser des figures géométriques de la même manière qu'avec **Scratch**, mais au lieu de manipuler des blocs, on écrit un code Python qui d'ailleurs s'organise aussi en blocs...

On utilise les fonctions de ce module dans un programme après un : from turtle import *
Pour vérifier que le module est installé sur votre système, saisir puis exécuter le programme suivant, qui normalement doit afficher le résultat sur la droite.

```
from turtle import *

forward(100)
left(120)
forward(100)
left(120)
forward(100)
```

Il est possible d'utiliser le module Turtle en ligne, par exemple à l'adresse :

`http://fe.fil.univ-lille1.fr/apl/2018/turtle.html`

Liste des commandes les plus utiles avec Turtle

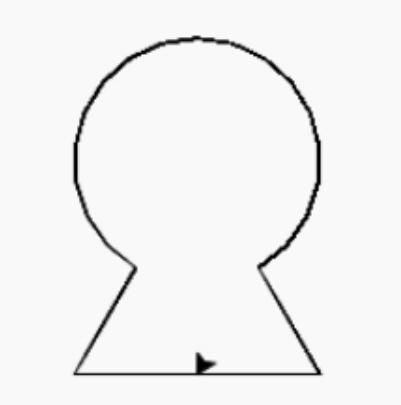
instruction	description
goto(x, y)	aller au point de coordonnées (x, y)
forward(d)	avancer de la distance d
backward(d)	reculer de la distance d
left(a)	pivoter à gauche de l'angle a
right(a)	pivoter à droite de l'angle a
setheading(a)	fixer la direction avec l'angle a
circle(r , a)	tracer un cercle d'angle a et de rayon r
dot(r)	tracer un point de rayon r
up()	relever le crayon (et interrompre le dessin)
down()	redescendre le crayon (et interrompre le dessin)
width(e)	fixe à e l'épaisseur du trait
color(c)	sélectionner la couleur c pour les traits
begin_fill()	activer le mode remplissage
end_fill()	désactiver le mode remplissage
fillcolor(c)	sélectionner la couleur c pour le remplissage
write('texte')	écrit un texte
clear()	efface tout ce qui est tracé
reset()	tout effacer et recommencer
speed(s)	définir la vitesse de déplacement
setup(L, l)	définit la longueur et la hauteur de la fenêtre.

Au départ, la tortue est en $(0,0)$, orientée à 0^o, c'est à dire vers la droite. La fenêtre par défaut fait $950 = 2 \times 475$ pixels de large, et $800 = 2 \times 400$ pixels de haut. Le point $(0,0)$ est au centre de l'écran. Il est possible de modifier cette fenêtre avec **setup** et **clear**.

Les couleurs sont notamment "black", "blue" , "red", "green" ... ou bien en RGB comme (0.3,0.3,0.3) pour un gris foncé.

Exercice 1 - Pour commencer.

Proposer un algorithme qui donne à peu près la figure suivante :



Exercice 2 - Quelques polygones réguliers.

1. Proposer un algorithme qui trace un hexagone régulier.
2. Proposer un algorithme qui trace un octogone régulier.
3. Expliquer comment généraliser la méthode : écrire pour cela une fonction `polygone_reg(n,d)` qui, à partir du nombre de côtés n et de la longueur d'un côté d trace le polygone régulier.

Exercice 3 - Une spirale.

```
from turtle import *

circle(4,180)
circle(9,180)
circle(25,180)
```

1. Tester le code donné.
2. Proposer un algorithme qui permet de poursuivre la spirale automatiquement.

Exercice 4 - La balade de Rantanplan.

Dans la cours de la prison, trois gardiens, placés en triangle, appellent le chien Rantanplan qui creuse un trou.
A chaque fois que Rantanplan cesse de creuser :

1. il lève la tête, et choisit un gardien au hasard ;
2. il se dirige vers le gardien ;
3. à mi-chemin vers lui, il s'arrête, creuse un trou, puis recommence.

Programmer, avec **Turtle**, un algorithme qui permet visualiser, sur un grand nombre de répétitions, où sont les trous creusés par Rantanplan.

- la position des trois gardiens est un choix arbitraire ;

- au départ, on pourra placer le chien au milieu des deux premiers gardiens ;
- on pourra utiliser `dot(r)` pour représenter un trou ;
- pour accélérer la visualisation, on peut changer la vitesse avec `speed(v)` .