Le module **turtle** est un ensemble d'outils permettant de dessiner à l'aide d'instructions simples.

Imaginez un robot se déplaçant dans un plan muni d'un repère orthonormé. Il est positionné au départ en (0 ; 0) et dirigé vers l'est. Il peut avancer ou reculer d'une certaine distance, exprimée en **pixel** ; il peut tourner vers la droite ou la gauche d'un certain angle, exprimé en **degré**.

Étudions un premier exemple : taper le script ci-dessous dans la console (sans le commantaire).

```
from turtle import*
# on importe l'ensemble des fonctions du module turtle
forward(150)
left(120)
color('red')
forward(150)
color('blue')
goto(0, 0)
```

Une fois le module **turtle** importé, on peut utiliser toutes les « fonctions » (plus précisément, méthodes) définies dans ce module, dont les principales sont détaillées ci-dessous. Bien entendu, si l'on désire parcourir l'ensemble des méthodes disponibles, on peut aller voir la documentation officielle accessible sur le site : https://docs.python.org/fr/3.6/library/turtle.html.

Mouvements de la tortue

Méthode	Description
`forward(d)`	Avancer d'une distance `d` (en pixels).
`backward(d)`	Reculer d'une distance `d` (en pixels).
`left(a)`	Faire pivoter la tortue d'un angle de mesure `a` degrés vers la gauche.
`right(a)`	Faire pivoter la tortue d'un angle de mesure `a` degrés vers la droite.
`goto(x, y)`	Positionner la tortue au point de coordonnées `(x,y)` tout en conservant sa direction.
`home()`	Replacer la tortue au point (0 ; 0) dirigée vers l'est.
`circle(r)`	Tracer un cercle de rayon `r` à partir de la position de la tortue.
`circle(r, s)`	Tracer un arc de cercle de rayon `r` correspondant à `s` degrés.

Contrôle du stylo

Méthode	Description
`up()`	Lever le crayon (pour déplacer la tortue sans dessiner).
`down()`	Baisser le crayon (pour recommencer à dessiner).
`width(n)`	Fixer la largeur du trait du crayon à `n` pixels.
`color(arg)`	Changer la couleur du crayon par `arg` où `arg` est : • soit une chaîne de caractères représentant une couleur, par exemple : `'red'`, `'blue'`, `'green'`, `'yellow'` ... • soit un triplet de flottants correspondant aux composantes d'une couleur dans l'espace colorimétrique RVB, par exemple `(1, 0, 0)` pour désigner le rouge, `(0, 0, 1)` pour le bleu, etc.
`fillcolor(arg)`	Remplir une figure fermée à l'aide de la couleur définie par `arg` (chaîne de caractères ou triplet de flottants).

Les balises `begin_fill()` et `end_fill()` permettent de commencer et de terminer le remplissage d'une figure géométrique.

La méthode `reset()` nettoie la fenêtre de dessin et réinitialise la tortue.

On peut aussi redimmensionner la fenêtre de la tortue à l'aide de l'instruction `setup(largeur, hauteur)`, où `largeur` et `hauteur` sont exprimées en pixels. Par exemple `setup(800, 600)` affiche une fenêtre de dimensions 800×600.

En outre, la vitesse de la tortue peut être modifiée à l'aide de la commande `speed(s)`, où `s` est un entier variant de 1 (vitesse la plus lente) à 10 (vitesse la plus rapide).

Enfin, il peut être utile dans certains cas de convertir des radians en degrés et réciproquement. On utilise alors respectivement les fonctions `degrees(a)` et `radians(a)`.

Exercice 1

1. Recopier et compléter dans l'éditeur la fonction `carre` ayant pour argument un entier naturel `l`.

```
def carre(l):
    """trace un carre de longueur l pixels"""
    for i in range(...):
        ...
        ...
```

2. Écrire un script permettant de reproduire la figure ci-dessous.