

Tortue (Scratch avec Python)

Le module turtle permet de tracer facilement des dessins en Python. Il s'agit de commander une tortue à l'aide d'instructions simples comme « avancer », « tourner »... C'est le même principe qu'avec Scratch, avec toutefois des différences : tu ne déplaces plus des blocs, mais tu écris les instructions ; et en plus les instructions sont en anglais !

[Vidéo ■ Tortue - partie 1](#)

[Vidéo ■ Tortue - partie 2](#)

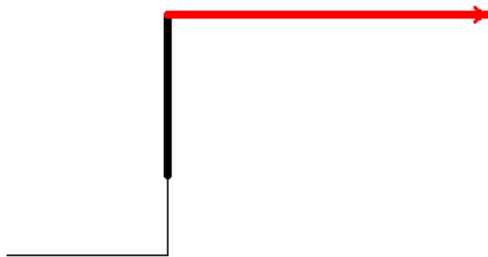
Cours 1 (La tortue Python).

La tortue c'est l'ancêtre de *Scratch* ! En quelques lignes tu peux faire de beaux dessins.

```
from turtle import *

forward(100)  # On avance
left(90)     # 90 degrés à gauche
forward(50)
width(5)     # Epaisseur du trait
forward(100)
color('red')
right(90)
forward(200)

exitonclick()
```



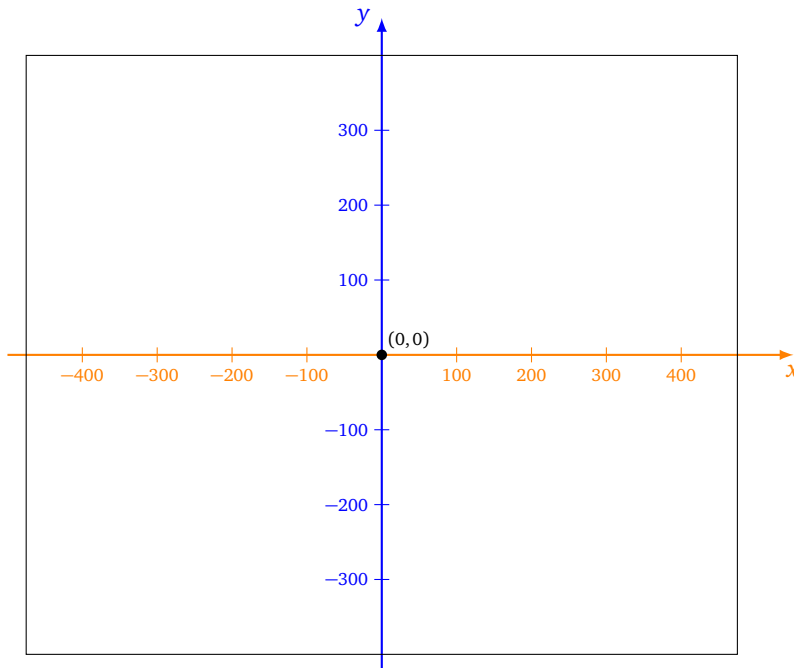
Voici une liste des principales commandes, accessibles après avoir écrit :

```
from turtle import *
```

- `forward(longueur)` avance d'un certain nombre de pas
- `backward(longueur)` recule
- `right(angle)` tourne vers la droite (sans avancer) selon un angle donné en degrés

- `left(angle)` tourne vers la gauche
- `setheading(direction)` s'oriente dans une direction (0 = droite, 90 = haut, -90 = bas, 180 = gauche)
- `goto(x,y)` se déplace jusqu'au point (x,y)
- `setx(newx)` change la valeur de l'abscisse
- `sety(newy)` change la valeur de l'ordonnée
- `down()` abaisse le stylo
- `up()` relève le stylo
- `width(epaisseur)` change l'épaisseur du trait
- `color(couleur)` change la couleur : "red", "green", "blue", "orange", "purple",...
- `position()` renvoie la position (x,y) de la tortue
- `heading()` renvoie la direction angle vers laquelle pointe la tortue
- `towards(x,y)` renvoie l'angle entre l'horizontale et le segment commençant à la tortue et finissant au point (x,y)
- `exitonclick()` termine le programme dès que l'on clique

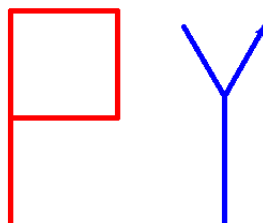
Les coordonnées de l'écran par défaut vont de -475 à +475 pour les x et de -400 à +400 pour les y ; (0,0) est au centre de l'écran.



Activité 1 (Premiers pas).

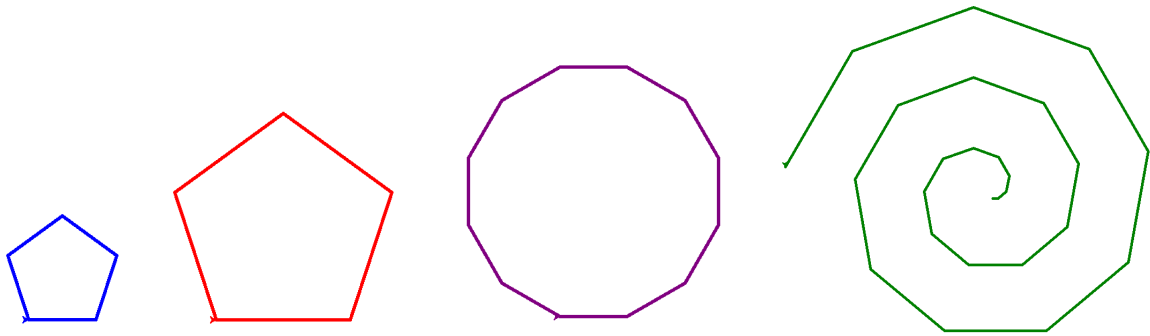
Objectifs : tracer tes premiers dessins.

Trace les premières lettres de Python, par exemple comme ci-dessous.



Activité 2 (Figures).

Objectifs : tracer des figures géométriques.



1. **Pentagone.** Trace un premier pentagone (en bleu). Tu dois répéter 5 fois : avancer de 100 pas, tourner de 72 degrés.

Indication. Pour construire une boucle utilise

```
for i in range(5):
```

(même si tu n'utilises pas ensuite la variable i).

2. **Pentagone (bis).** Définis une variable longueur qui vaut 200 et une variable angle qui vaut 72 degrés. Trace un second pentagone (en rouge) en avançant cette fois de longueur et en tournant de angle.

3. **Dodécagone.** Trace un polygone à 12 côtés (en violet).

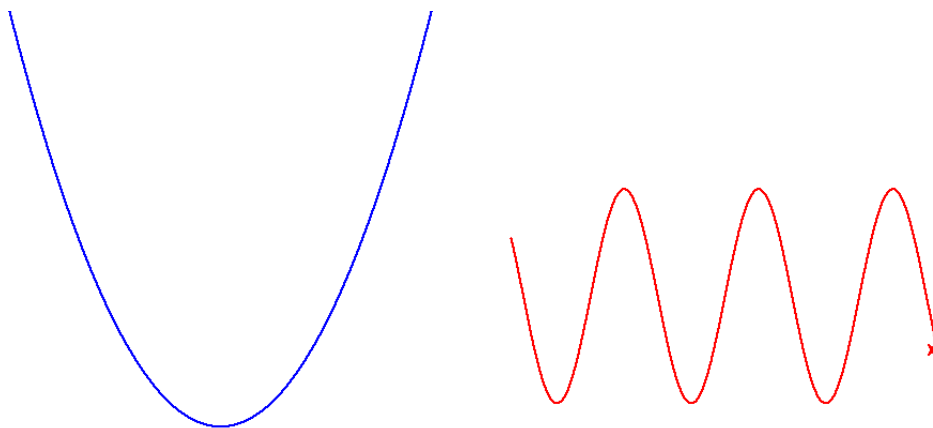
Indication. Pour tracer un polygone à n côtés, il faut tourner d'un angle de $360/n$ degrés.

4. **Spirale.** Trace une spirale (en vert).

Indication. Construis une boucle, dans laquelle tu tournes toujours du même angle, mais par contre tu avances d'une longueur qui augmente à chaque étape.

Activité 3 (Graphe de fonctions).

Objectifs : tracer le graphe d'une fonction.



Trace le graphe de la fonction carré et de la fonction sinus.

Afin d'obtenir une courbe dans la fenêtre de la tortue, répète pour x variant de -200 à $+200$:

- poser $y = \frac{1}{100}x^2$,

- aller à (x, y) .

Pour la sinusoïde, tu peux utiliser la formule

$$y = 100 \sin\left(\frac{1}{20}x\right).$$

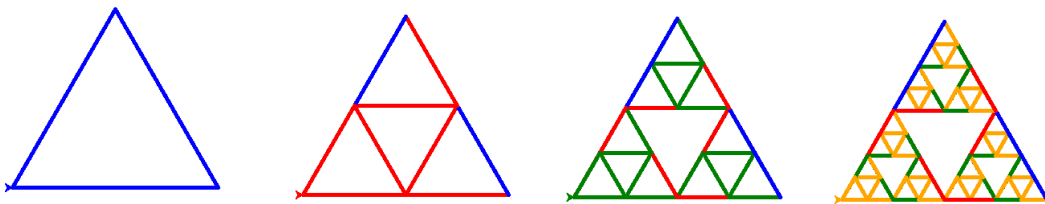
Par défaut Python ne connaît pas la fonction sinus, pour utiliser `sin()` il faut d'abord importer le module `math` :

```
from math import *
```

Pour que la tortue avance plus vite, tu peux utiliser la commande `speed("fastest")`.

Activité 4 (Triangle de Sierpinski).

Objectifs : tracer le début de la fractale de Sierpinski en imbriquant des boucles.



Voici comment tracer le second dessin. Analyse l'imbrication des boucles et trace les dessins suivants.

```
for i in range(3):
    color("blue")
    forward(256)
    left(120)

for i in range(3):
    color("red")
    forward(128)
    left(120)
```

Activité 5 (Le cœur des tables de multiplication).

Objectifs : dessiner les tables de multiplication. Pour une introduction en vidéo voir [la face cachée des tables de multiplication](#) sur Youtube par Micmaths.

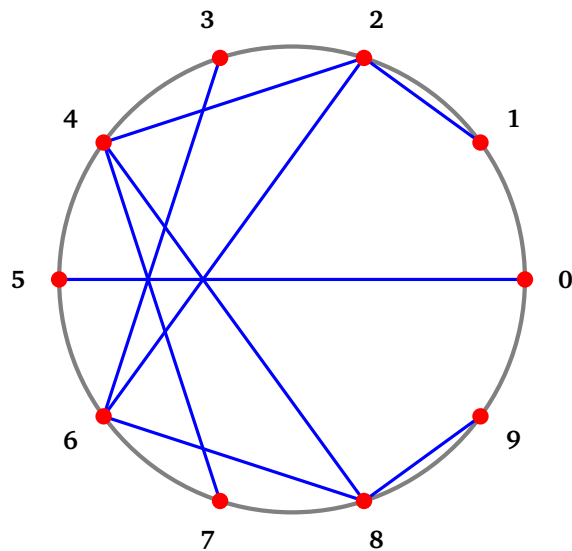
On fixe un entier n . On étudie la table de 2, c'est-à-dire que l'on calcule 2×0 , 2×1 , 2×2 , jusqu'à $2 \times (n-1)$. En plus les calculs se feront modulo n . On calcule donc

$$2 \times k \pmod{n} \quad \text{pour } k = 0, 1, \dots, n-1$$

Comment dessiner cette table ?

On place sur un cercle n points numérotés de 0 à $n-1$. Pour chaque $k \in \{0, \dots, n-1\}$, on relie le point numéro k et le point numéro $2 \times k \pmod{n}$ par un segment.

Voici le tracé, de la table de 2, modulo $n = 10$.

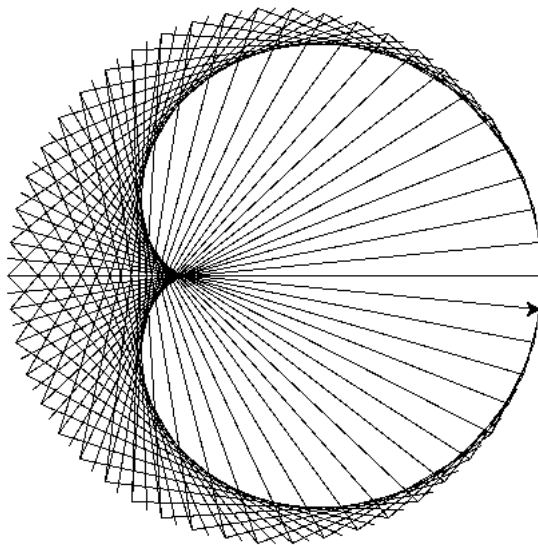


Par exemple :

- le point 3 est relié au point 6, car $2 \times 3 = 6$;
- le point 4 est relié au point 8, car $2 \times 4 = 8$;
- le point 7 est relié au point 4, car $2 \times 7 = 14 = 4 \pmod{10}$.

Trace la table de 2 modulo n , pour différentes valeurs de n .

Voici ce que cela donne pour $n = 100$.



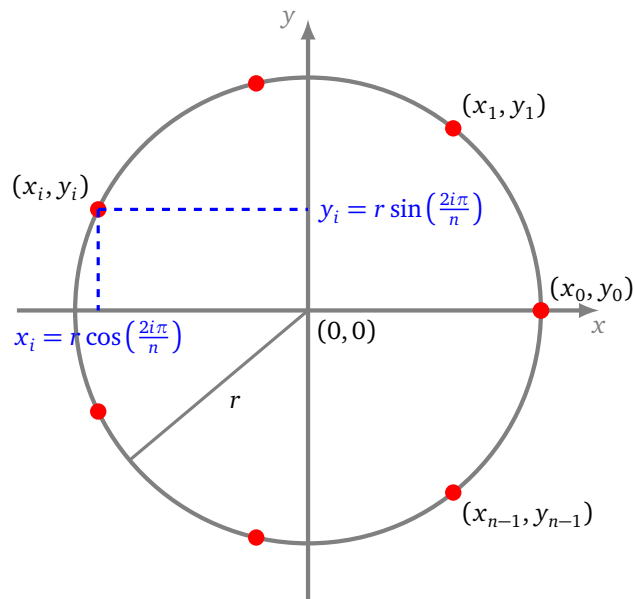
Indications.

Pour les calculs modulo n , utilise l'expression $(2*k) \% n$.

Voici comment obtenir les coordonnées des sommets. Cela se fait avec les fonctions sinus et cosinus (disponibles à partir du module `math`). Les coordonnées (x_i, y_i) du sommet numéro i , peuvent être calculées par la formule :

$$x_i = r \cos\left(\frac{2i\pi}{n}\right) \quad \text{et} \quad y_i = r \sin\left(\frac{2i\pi}{n}\right)$$

Ces sommets seront situés sur le cercle de rayon r , centré en $(0,0)$. Tu devras choisir r assez grand (par exemple $r = 200$).



Cours 2 (Plusieurs tortues).

On peut définir plusieurs tortues qui se déplacent de façon indépendante chacune de leur côté. Voici comment définir deux tortues (une rouge et une bleue) et les déplacer.

```
tortue1 = Turtle()    # Avec un 'T' majuscule !
tortue2 = Turtle()

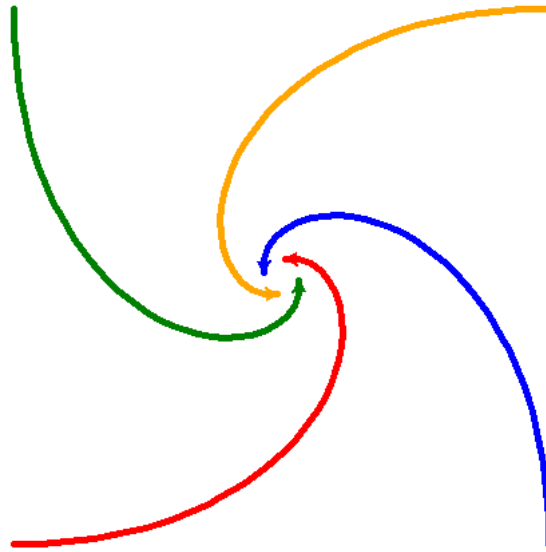
tortue1.color('red')
tortue2.color('blue')

tortue1.forward(100)
tortue2.left(90)
tortue2.forward(100)
```

Activité 6 (La poursuite des tortues).

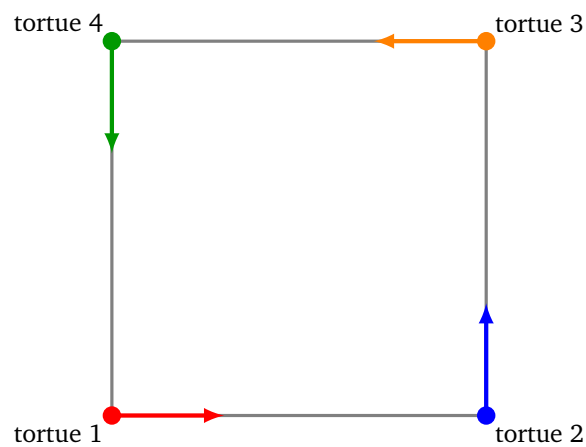
Objectifs : tracer des courbes de poursuite.

Programme quatre tortues qui courent les unes après les autres :



- la tortue 1 court après la tortue 2,
- la tortue 2 court après la tortue 3,
- la tortue 3 court après la tortue 4,
- la tortue 4 court après la tortue 1.

Voici les positions et orientations de départ :



Indications. Utilise le bout de code suivant :

```
position1 = tortue1.position()
position2 = tortue2.position()
angle1 = tortue1.towards(position2)
tortue1.setheading(angle1)
```

- Tu places les tortues aux quatre coins d'un carré, par exemple en $(-200, -200)$, $(200, -200)$, $(200, 200)$ et $(-200, 200)$.
- Tu récupères la position de la première tortue par `position1 = tortue1.position()`. Idem pour les autres tortues.
- Tu calcules l'angle entre la tortue 1 et la tortue 2 par la commande `angle1 = tortue1.towards(position2)`.

- Tu orientes la tortue 1 selon cet angle : `tortue1.setheading(angle1)`.
- Tu avances la tortue 1 de 10 pas.

Améliore ton programme en traçant à chaque fois un segment entre la tortue poursuivante et la tortue poursuivie.

