

Activité 1 : Les figures dynamiques

Dans cette activité, nous allons réaliser de figures dynamiques en utilisant le langage de programmation Python et particulièrement sa bibliothèque turtle. Ce module permet de déplacer un point dans un espace 2D (ce point est souvent vu comme une tortue), et ainsi de réaliser des dessins. Les commandes de bases de la bibliothèque Turtle sont les suivantes :

up() : lève le crayon	width(e) : définit l'épaisseur du trait
down() : baisse le crayon	speed("texte") : définit la vitesse d'exécution
forward(n) : avance de n	write("texte") : écrit le texte
left(d) : tourne vers la gauche de d degrés	color("couleur") : définit la couleur du trait
right(d) : tourne vers la droite de d degrés	bgcolor("couleur") : définit la couleur de fond
goto(x,y) : se déplace vers le point de coordonnées (x,y)	reset() : efface tout
circle(r) : dessine un cercle de rayon r	done() : arrête le dessin

Dans les instructions color() et bgcolor() précédentes, les couleurs suivantes peuvent être choisies : blue, red, black, green, . . .

Pour la vitesse, on peut choisir (du plus rapide au plus lent) : slowest, slow, normal, fast et fastest.

Le début de chaque programme doit faire appel à la bibliothèque de fonctions Turtle en tapant la directive `from turtle import *`

1. Prise en main du logiciel

- **Créer** un fichier nommé first.py et contenant le code ci-contre.
- **Exécuter** ce programme et **analyser** le résultat afin d'identifier le rôle de chaque instruction
- **Modifier** ce programme afin de tester les fonctions suivantes : right, circle, width et color.

Ma première figure

- x **Dessiner** un carré rouge de côté 100 et un autre jaune deux fois plus petit et centré sur le premier
- x **Dessiner** un carré rouge de côté 100 et un cercle jaune de rayon 80 centré sur le carré

```
from turtle import *
longueur = 120
angle = 90
reset()
forward(longueur)
left(angle)
color('red')
forward(longueur-40)
done()
```

2. Booster ses programmes

Les répétitions

Nous nous rendons compte que le programme précédent comporte plusieurs fois les mêmes instructions. Il est possible de simplifier le code en répétant plusieurs fois le groupe d'instruction à l'aide d'une boucle for. La syntaxe de de cette structure répétitive est la suivante :

Instructions à répéter.
Elles doivent décalées
d'une tabulation

```
for i in range(2):  
    Instruction 1  
    Instruction 2
```

Nombre de
répétitions

En utilisant cette boucle répétitive, le code pour dessiner un carré de 50 de côté devient :

x **Dessiner** un triangle équilatéral jaune de 70 de côté en utilisant une boucle for pour répéter 3 fois le dessin d'un côté du triangle

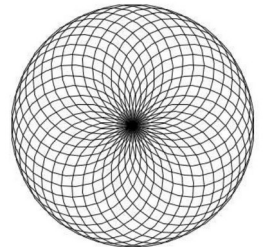
x **Réaliser** le dessin de la figure suivante constituée de dix carrés de côté 20 avec un espace de 5 entre chacun d'eux.

```
from turtle import *  
reset()  
for i in range(4) :  
    forward(50)  
    Left(90)  
done()
```



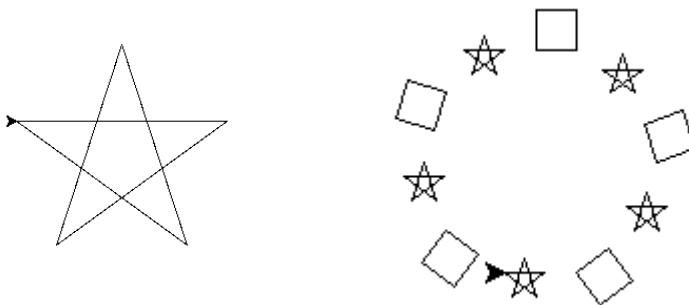
x **Ecrire** un programme réalisant le dessin ci-contre.

Pour obtenir ce dessin, on peut observer qu'il est constitué de cercles de même rayon (80 dans notre cas), avec un décalage de 10 degrés entre deux cercles successifs (soit 36 cercles au total).



3. Pour aller plus loin

x **Écrire** un programme, qui trace les figures suivantes :



4. Jouons avec les couleurs

La fonction `color()` modifie la couleur du tracé en cours. En conditionnant l'appel de cette fonction à certaines conditions il est alors possible d'introduire des couleurs dans les figures.

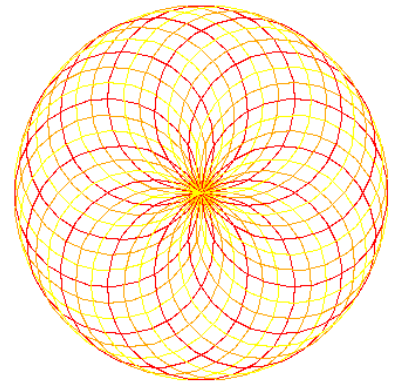
Des conditions peuvent être implémentées dans le programme avec l'instruction `if`. Cette instruction exécute une suite d'instructions si l'expression associée est vraie :

Instructions à effectuer. Elles doivent être décalées d'une tabulation

```
if expression :  
    Instruction 1  
    Instruction 2
```

Expression associée de test

x **Modifier** les deux programmes précédents afin d'obtenir les figures suivantes :



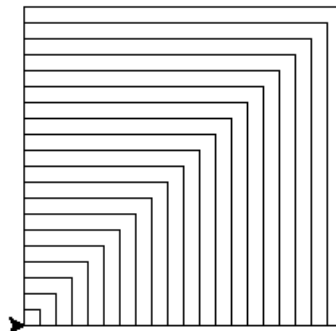
5. Les variables

Les variables sont l'un des concepts qui se retrouvent dans la totalité des langages de programmation. Le principal intérêt d'une variable réside dans le fait qu'il est possible de la faire évoluer durant l'exécution du programme. Par exemple le code ci-contre permet de dessiner deux carrés l'un dans l'autre.

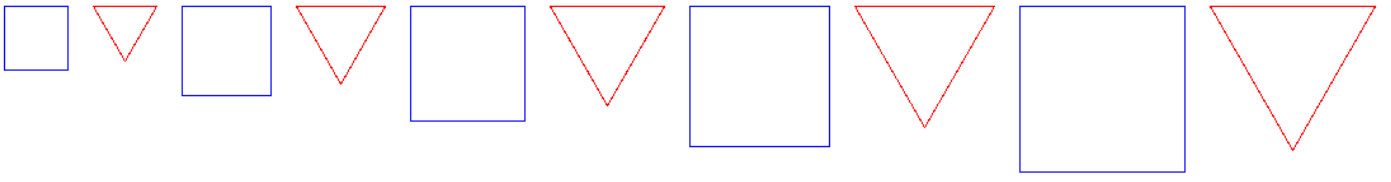
x **Analyser** ce code et **déterminer** la longueur des côtés des deux carrés

```
from turtle import *  
reset()  
speed('fastest')  
down()  
lg = 80  
for i in range(2) :  
    for j in range(4) :  
        forward(lg)  
        left(90)  
    lg = lg + 200
```

x **Modifier** le programme précédent afin d'obtenir la figure suivante. On peut noter qu'elle est constituée de 20 carrés dont le plus petit a des côtés de 10 et le dernier de 200



x **Écrire** un programme, qui trace un carré puis un triangle, qui grossissent au fur et à mesure.



6. A vous de dessiner

x Voici des figures dont vous devez trouver le code :

