TP2-Introduction à la programmation à l'aide du python

Activité 1

Écrire le code suivant dans Thonny et l'enregistrer dans votre dossier de travail sous le nom « carre1.py »

```
from turtle import *
title("Activité 1")
setup(1050,600)
up()
goto(-450,-100)
down()
shape("classic")
pensize(4)
fd(180)
lt(90)
fd(180)
lt(90)
fd(180)
lt(90)
fd(180)
lt(90)
```

1- À quoi servent les méthodes

- title : permet d'ajouter un titre dans la barre des titres
- shape : permet de choisir la forme de la tortue
- setup : permet de définir les dimensions de la fenêtre d'exécution
- shapesize : permet de changer les dimensions de la tortue
- pensize : permet de changer l'épaisseur du crayon
- goto : permet de déplacer le curseur dans un endroit voulu
- up et down : lever / baisser le crayon

2- On remarque que les instructions 9 et 10 se répètent 4 fois, on parle d'un traitement itératif ou dit répétitif

Et puisque le nombre de répétitions est connu à l'avance, en programmation, on peut utiliser une structure qui permet à l'ordinateur de répéter un traitement plusieurs fois. Cette instruction est dite structure itérative complète pour (for)(on connu le nombre de répétition en avance)

Et qui peut être écrite, sous python, sous la forme

```
for compteur in range (nombre d'itération) :
        traitement
```

En utilisant la structure for, simplifier le code ci-dessus : les espaces avant le traitement sont dites "indentation" (obligatoire pour mentionner à l'ordinateur que le traitement dans la boucle pour.

```
for i in range(4) :
        fd(180)
        lt(90)
```

3- En utilisant la méthodes color et begin_fill() et end_fill() colorer le résultat par une couleur de votre choix.

```
color('black', 'sky blue')
begin_fill()
for i in range(4) :
        fd(180)
        lt(90)
end_fill()
```

4- On donne la liste des couleurs suivantes : "indian red", "firebrick", "red", "red2", "red3" et "red4" et en utilisant l'instruction :

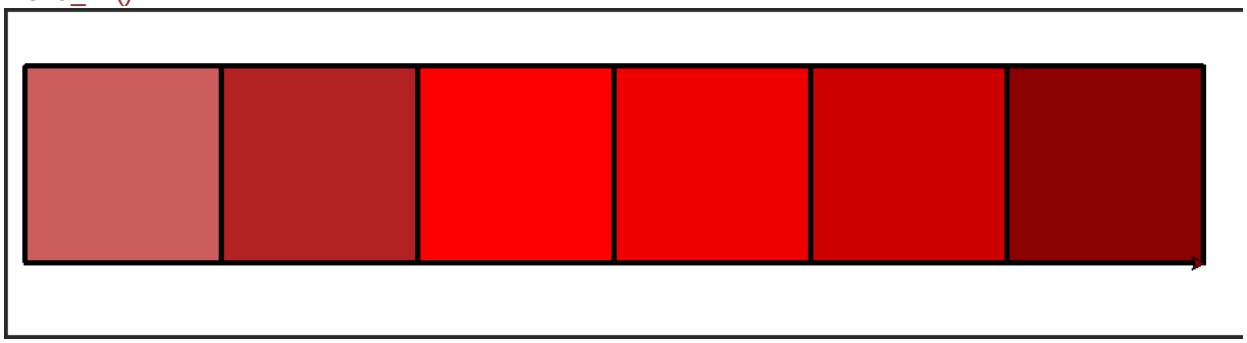
```
couleur = ["indian red","firebrick","red","red2","red3","red4"]
```

qui permet de définir dans python une liste (un tableau) transformer le code pour tracer 5 carrés colorés par les couleurs de la liste couleur.

Constatation :

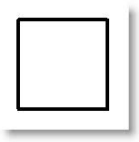
Une structure répétitive est : une structure qui permet à l'ordinateur de répéter un certain traitement plusieurs fois.
Une liste est : un objet qui permet d'enregistrer plusieurs valeur dans une même structure

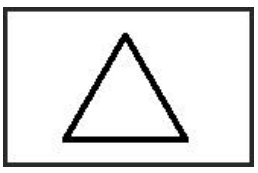
```
from turtle import *
title("Activité 1")
setup(1050,600)
up()
goto(-450,-100)
down()
shape("classic")
pensize(4)
couleur = ['indian red',"firebrick","red","red2","red3","red4"]
for i in range(6):
    color('black',couleur[i])
    begin_fill()
    for j in range(4):
        fd(150)
        lt(90)
    end_fill()
```

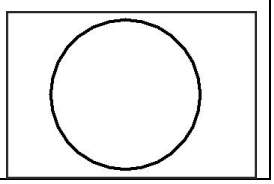


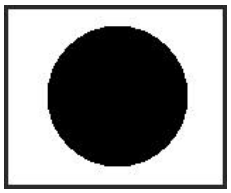
Activité 2

En utilisant Thonny, écrire les codes qui permettent de tracer les formes suivantes :

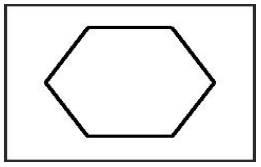
Un carré de coté 150	`for i in range(4) :`
	`  fd(150)`
	`  lt(90)`
		

Un triangle équilatérale de coté 80	`for i in range (3) :`
	`  fd(80)`
	`  lt(120)`
		

Un cercle de rayon 50	`circle(50)`	

Un disque de diamètre 120	dot(120)	

Une étoile de 5 branches	for i in range(5) :
	  fd(100)
	  lt(144)
		

Un polygone de 6 cotés de langueurs 80	for i in range(6) :
	  fd(80)
	  lt(60)
		

Activité 3

Écrire un code python permettant de dessiner le drapeau de la Tunisie 

```
from turtle import *
bgcolor('red')
pensize(3)

color('white')
penup()
goto(0,-120)
pendown()
begin_fill()
circle(150)
end_fill()

penup()
goto(0,-90)
pendown()
pendown()
color('red')
begin_fill()
circle(120)
end_fill()
```

```
penup()
goto(29,-80)
pendown()
pendown()
color('white')
begin_fill()
circle(100)
end_fill()

penup()
setheading(30)
goto(10,-25)
color('red')
begin_fill()
down()
for i in range(5):
   forward(100)
   lt(144)

end_fill()
hideturtle()
```

Activité 4

En se basant de tout ce qu'on a vu et en utilisant https://docs.python.org/fr/3/library/turtle.html et à l'aide de « TP2.py », remplir le tableau suivant :

Annexe (méthodes utiles Turtle)

Méthode (paramètres)	Description
Méthodes agissant sur la fenêtre d'exécution	
setup (largeur, hauteur)	Définir les dimensions de la fenêtre
bgcolor('couleur')	Définir la couleur du font
clear()	Efface le contenue de la fenêtre
time.sleep(durée)	Il faut importe la bibliothèque time permet d'arrêter l'exécution du programme pendant une durée
exitonclick()	Fermer la fenêtre lors d'un click de la souris
reset()	Efface la fenêtre et remettre la tortue au milieu
exit()	Fermer la fenêtre
Méthodes agissants sur le comportement de la tortue	
shape('forme')	Change la forme de la tortue
up()	Lever le crayon
down()	Baisser le crayon
hideturtle()	Masquer la tortue
showturtle()	Montrer la tortue
speed(n)	Définir la vitesse (de 1 à 10)
pensize(n)	Définir l'épaisseur du crayon (de1 à 10)
width(n)	Définir l'épaisseur du crayon (de 1 à 10)
shapesize(n)	Définir la taille de la tortue
Méthodes agissant sur le déplacement de la tortue	
forward()	Ou fd() pour avancer d'un distance en pixel
back()	Ou bd() pour reculer d'une distance en pixel
goto(x,y)	Déplacer vers la point (x,y)
home()	Déplacer vers le point de centre
left()	Ou lt() Tourner la tortue vers la gauche
right	Ou rt() Tourne la tortue vers la droite
setheading(angle)	Tourne la tortue vers soit 0 , 90, 180 ou 270 degré
Méthodes agissant sur les couleurs	
couleur('couleur')	Définir la couleur du crayon et les formes
couleur('couleur1','couleur2')	Définir la couleur du crayon(couleur1) et au formes(couleur2)
begin_fill()	Commencement du remplissage avec la couleur
end_fill()	Fin du remplissage avec la couleur
Méthodes pour tracer des formes spécifiques	
circle(rayon,[angle])	Dessiner un cercle ou un arc du cercle
dot(diamètre, ['couleur'])	Dessine un disque noir ou avec une couleur = couleur
write('message',[,font=("police", taille, "style")])	Permet d'écrire des message dans la fenêtres