Info 4 - Fonctions

L'objectif principal de ce second TP est d'utiliser de manipuler les fonctions. On aborde aussi la notion de liste.

On demande de rédiger un compte-rendu au format *.doc* ou *.odt* à transférer en fin d'activité par l'intermédiaire de l'onglet ***Mon Compte*** du site https://nsibranly.fr en utilisant le code : **tp4** . Ce compte-rendu contiendra :

- les réponses aux différentes questions posées,
- les captures d'écran **des morceaux de codes** écrits **et** celles **des résultats des exécutions**. Pour faire ces captures, utiliser *l'Outil Capture d'écran* de Windows (touches clavier *windows+Shift+s*)

Attention à repérer correctement les titres de paragraphe.

1- FONCTION QUI CALCULE MON AGE :

Une fonction nommée *monAge()* renvoie l'âge en fonction de l'année de naissance.

```
# Programme principal
a = monAge(2007)
print(a)
a = monAge(2025)
print(a)
print(monAge(2000))
```

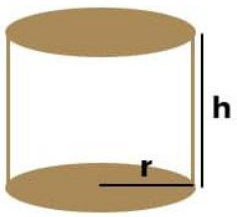
En exécutant le code ci-contre, on obtient dans la console :

```
>>> (executing file "tp2023.py")
16
pas encore né
23
```

⇨ Créer le script de cette fonction *monAge()* et tester l'ensemble.

2- FONCTION QUI CALCULE UN VOLUME :

Le volume d'un cylindre est donné par la relation ci-contre avec $\pi \approx 3.14$.



volume = π x r² x h

Une fonction nommée *volumeCylindre()* renvoie le volume d'un cylindre, en fonction de son rayon et de sa hauteur mis en arguments.

En exécutant le code ci-contre, on obtient dans la console :

```
>>> (executing file "tp2023.py")
6.28 m3
les valeurs doivent être positives
les valeurs doivent être positives
```

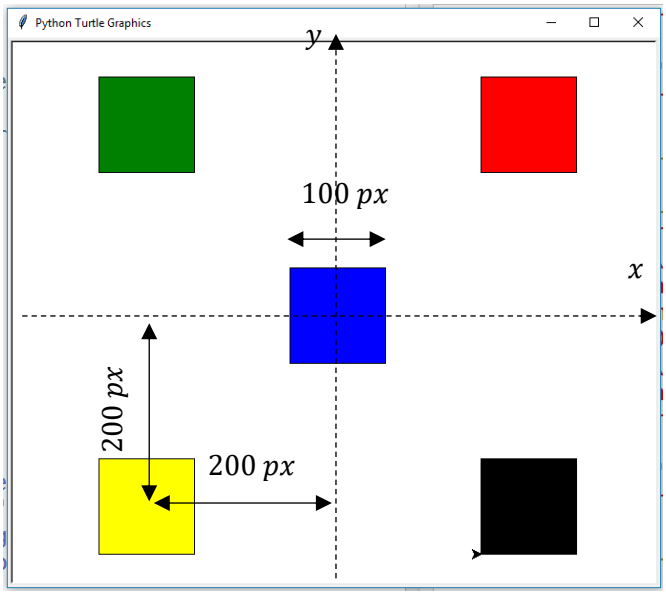
```
# Programme principal
vol = volumeCylindre(1,2)
print(vol)

sortie = volumeCylindre(-1,2)
print(sortie)

message =  volumeCylindre(1,-2)
print(message)
```

⇨ Créer le script de cette fonction *volumeCylindre ()* et tester l'ensemble.

3- FONCTION QUI TRACE DES CARRES DE COULEUR :

Une fonction nommée *carre()* permet de tracer avec la bibliothèque *turtle,* un carré en fonction de la longueur en pixel de son coté, des coordonnées x, y **du centre** du carré (en pixels) et de la couleur.

En exécutant le code ci-contre, on obtient la fenêtre graphique ci-dessous :

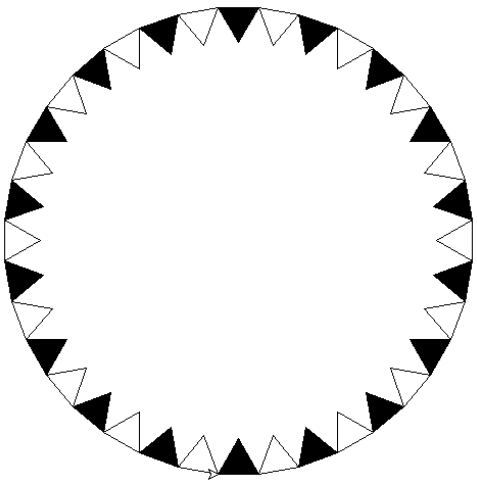



```
# Programme principal
from turtle import *
carre(100,0,0,"blue")
carre(100,200,200,"red")
carre(100,-200,-200,"yellow")
carre(100,-200,200,"green")
carre(100,200,-200,"black")

mainloop()
```

⇨ Créer le script de cette fonction *carre()* et tester l'ensemble.

4- FONCTION QUI TRACE DES TRIANGLES NOIRS OU BLANCS :

Une fonction nommée *triangle()* prend en argument la longueur du côté d'un triangle équilatéral et les 3 coefficients de la couleur de remplissage (nombres compris entre 0 et 1). En exécutant le code ci-contre, on obtient la fenêtre graphique ci-contre.



⇨ Créer le script de cette fonction *triangle()* et tester l'ensemble.

```
# Programme principal
from turtle import *
speed(100)
up()
goto(0,-200)
down()
for i in range(36) :
    if i%2 == 0 :
        c = 0
    else : c = 1
    triangle(40,c,c,c)
    left(10)

mainloop()
```

5- FONCTIONS LIEES A DES EVENEMENTS :

Un script python va toujours s'exécuter en étant lu de haut en bas. Une fois la lecture terminée, l'exécution s'arrête. En informatique, la plupart des applications sont à l'écoute de ce qui se passe et réagissent lorsque des évènements prédéfinis se produisent. On voit ce nouveau mode de fonctionnement dans cet exercice :

```
#Fonctions
def avance():
    forward(10)

def tourneG():
    left(90)

def tourneD():
    left(-90)

def direct(x, y):
    goto(x,y)

# Programme principal
from turtle import *
shape('turtle')  #apparence de la tortue

onkey(tourneG, 'Left') #évènement clavier
onkey(tourneD, 'Right') #évènement clavier
onkey(avance, 'space') #évènement clavier

onscreenclick(direct) #évènement click souris

#permet l'écoute des évènements
listen()
mainloop()
```

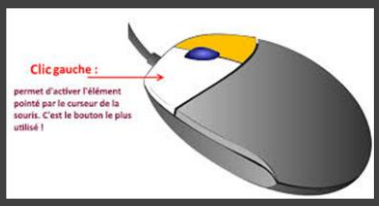
Le code ci-contre est composée de 4 fonctions.

Dans le programme principal, on conditionne l'exécution de ces fonctions à des évènements claviers ou souris. On utilise pour cela des fonctions de *turtle* qui sont *onkey()* pour un évènement clavier et *onscreenclick()* pour un évènement souris.

Si on actionne la touche ← du clavier, la fonction tourneG() est exécutée

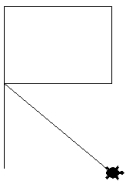
Attention : ***pas de parenthèses à tourneG ET 'Left' avec L majuscule***

Après un click souris sur la fenêtre turtle, la fonction direct() est exécutée



⇨ Saisir ce code et tester l'ensemble.

⇨ Ecrire avec la tortue la

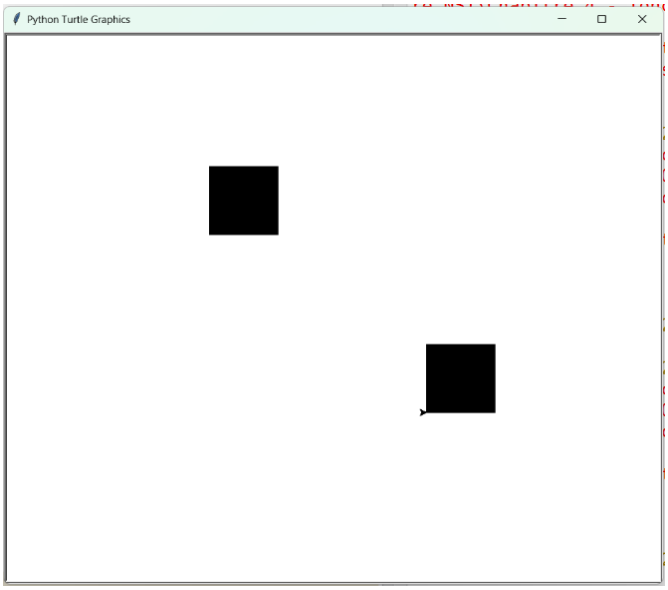


6- TOUJOURS PLUS LOIN AVEC LES EVENEMENTS :

Le code ci-contre est mal indenté.

⇨ Copier-coller ce code dans l'éditeur de pyzo et enregistrer le fichier.

⇨ Modifier et compléter le code afin qu'il donne à l'exécution une fenêtre graphique dans laquelle on crée un carré noir centré sur le point cliqué.




```
#Fonctions
def carre(a,x,y,r,g,b) :
speed(100)
up()
fillcolor(r,g,b)
x = x - a/2
y = y - a/2
goto(x,y)
down()
begin_fill()
for i in range(4) :
forward(a)
left(90)
end_fill()


def abracadabra(x, y):
a = 100
r = 0
g = 0
b = 0
carre(a,x,y,r,g,b)

# Programme principal
from turtle import *

onscreenclick(abracadabra)

listen()
mainloop()
```

7- TOUJOURS PLUS LOIN AVEC LES EVENEMENTS :

⇨ Modifier le code précédent afin de pouvoir cette fois-ci créer des carrés de taille et de couleur aléatoires à chaque click de souris.

8- FONCTION QUI PERMET DE SAVOIR SU UNE ANNEE EST BISSEXTILE :

Une fonction nommée *bissextile()* prend en argument une numéro d'année. Elle renvoie un message qui indique si cette année est bissextile.

Info tirée de Wikipédia :

> Depuis l'ajustement du calendrier grégorien, l'année n'est bissextile (elle aura 366 jours)[1] que dans l'un des deux cas suivants :
>
> 1. si l'année est divisible par 4 et non divisible par 100 ;
> 2. si l'année est divisible par 400.
>
> Dans un autre cas, l'année n'est pas bissextile : elle a la durée habituelle de 365 jours.

En exécutant le code ci-contre, on obtient dans la console :

```
>>> (executing file "tp2023.py")
Année : 2024
Année Bissextile
```

```
# Programme principal
n = int(input("Année : "))
message = bissextile(n)
print(message)
```

⇨ Créer le script de cette fonction *bissextile()* et tester l'ensemble.

9- ECRITURE DANS LA FENETRE TURTLE :

Le code ci-contre est mal indenté.

⇨ Copier-coller ce code dans l'éditeur de pyzo et enregistrer le fichier.

⇨ Modifier et compléter le code afin qu'il donne à l'exécution une fenêtre graphique dans laquelle on affiche les coordonnées du point cliqué.

```
#Fonctions
def test(x, y):
up()
goto(x,y)
dot(20,'red')
texte = '('+str(x)+","+str(y)+')'
write(texte, align="right" , font=("Arial",15, "normal"))

# Programme principal
from turtle import *
onscreenclick(test)

listen()
mainloop()
```

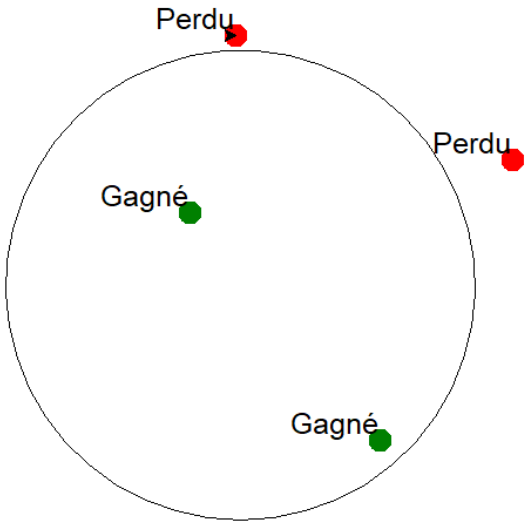
10- ON JOUE AUX FLECHETTES ? :

Le code ci-contre est mal indenté et surtout **incomplet** pour la fonction test().

⇨ Copier-coller ce code dans l'éditeur de pyzo et enregistrer le fichier.

⇨ Modifier et compléter le code afin qu'il donne à l'exécution une fenêtre graphique dans laquelle on affiche :

- « Gagné » et un point vert si on clique dans le cercle,
- « Perdu » et un point rouge si on clique en dehors.




```
#Fonctions
def traceCercle(r) :
up()
goto(0,-r)
down()
circle(r)

def test(x, y):
up()
goto(x,y)
print(x,y)
d = sqrt(x**2 + y**2)




# Programme principal
from turtle import *
from math import sqrt
speed(100)
traceCercle(200)
onscreenclick(test)

listen()
mainloop()
```