Ce travail s'ajoute à celui donné dans le tp1. Les réponses sont à donner dans un fichier texte .odt ou .doc qui contiendra les copies d'écran des morceaux de codes et des fenêtres d'exécutions. Ce fichier sera uploadé sur le site *nsibranly.fr* avec le code **tp1** toujours.

## 1. Se familiariser avec les fonctions de la bibliothèque *turtle*

Pour utiliser le module *turtle*, commencer le code avec l'instruction : *from turtle import **

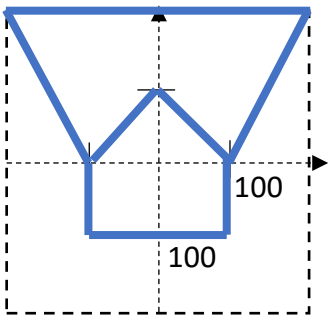
La dernière instruction du code sera : *mainloop()* afin de pouvoir maintenir la fenêtre graphique ouverte.

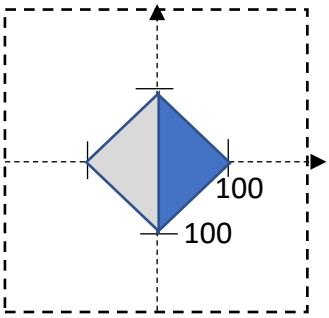
Le site https://docs.python.org/fr/3/library/turtle.html indique toutes les fonctions mises à disposition dans cette bibliothèque.

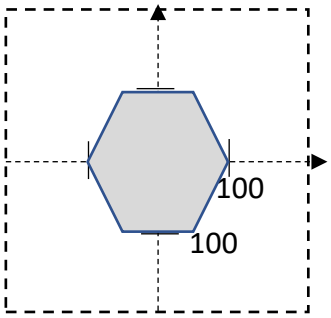
Parmi celles-ci, les principales que l'on utilisera seront :

- *goto(x,y)* pour se déplacer sur un point de coordonnées (x,y) ,
- *forward(d)* pour se déplacer d'une distance d (en pixels) dans l'axe de la tortue,
- left($a$) pour faire tourner la tortue sur elle-même de l'angle $a$ donné en degré,
- up() pour lever le crayon,
- down() pour abaisser le crayon,
- pencolor(''red'') pour que la couleur du trait tracé soit rouge par exemple,
- begin_fill() à appeler avant de dessiner une forme fermée à remplir de couleur,
- end_fill() permet de remplir la forme dessinée après l'appel de begin_fill() ,
- fillcolor(''red'') pour définir à rouge la couleur de remplissage.

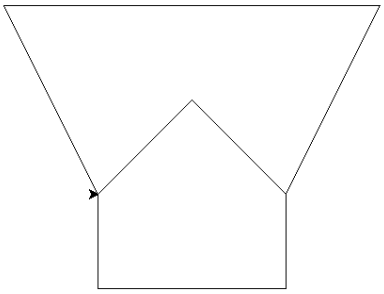
⇨ Reproduire au mieux, avec *turtle* les dessins suivants. Faire une copie d'écran de la figure et du code correspondant.



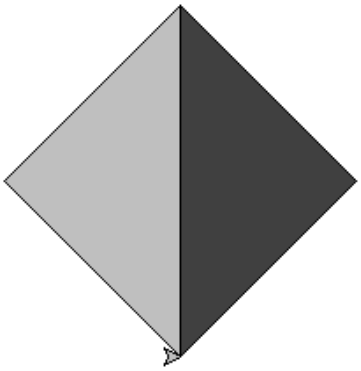



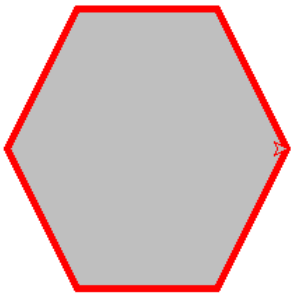
```
from turtle import *
# positionnement initial du crayon
up()
goto(100,0)
# tracé partie haute
down()
goto(200,200)
goto(-200,200)
goto(-100,0)
goto(0,100)
goto(100,0)
# tracé partie basse
goto(100,-100)
goto(-100,-100)
goto(-100,0)

# pour fermer proprement la fenêtre
exitonclick()

```



```
from turtle import *
# positionnement initial du crayon


# tracé partie droite

fillcolor(0.25,0.25,0.25)
begin_fill()
goto(0,100)
goto(100,0)
goto(0,-100)
end_fill()

# tracé partie gauche

fillcolor(0.75,0.75,0.75)
begin_fill()
goto(-100,0)
goto(0,100)
goto(0,-100)
end_fill()

# pour fermer proprement la fenêtre
exitonclick()
```



```
1  from turtle import *
2  # positionnement crayon
3  up()
4  goto(100,0)
5  # tracé héxagone
6  down()
7  fillcolor(0.75,0.75,0.75)
8  pencolor("red")
9  width(5)
10 begin_fill()
11 goto(50,100)
12 goto(-50,100)
13 goto(-100,0)
14 goto(-50,-100)
15 goto(50,-100)
16 goto(100,0)
17
18 end_fill()
19
20 # pour fermer proprement la fenêtre
21 exitonclick()
```



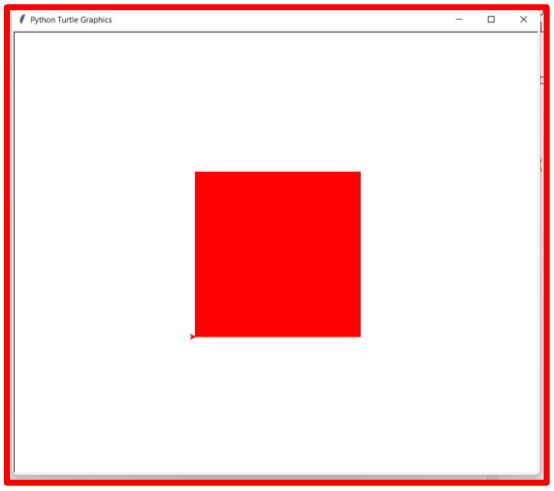
## 2. Utiliser l'équivalent de la fonction input() avec turtle

Les fonctions *numinput()* et *textinput()* de la bibliothèque *turtle* permettent de saisir respectivement des nombres et des textes par l'intermédiaire d'une fenêtre de dialogue.

Exercice 1 : Copier et exécuter le code ci-dessous :

```
from turtle import *
a = numinput("dimension", "Saisis le coté du carré")
coul = textinput("couleur", "Saisis la couleur")
pencolor(coul)
fillcolor(coul)
up()
goto(-a//2,-a//2)
begin_fill()
down()
goto(a//2,-a//2)
goto(a//2,a//2)
goto(-a//2,a//2)
goto(-a//2,-a//2)
end_fill()

mainloop()
```

1- Joindre un exemple de l'exécution de ce code dans le compte-rendu.




2- En utilisant la fonction *type()* native de python, donner le type de la variable *a* qui renvoyée par la fonction *numinput()* ?

```
>>> type(a)
<class 'float'>
```

3- Même question pour la variable *coul* renvoyée par *textinput()* ?

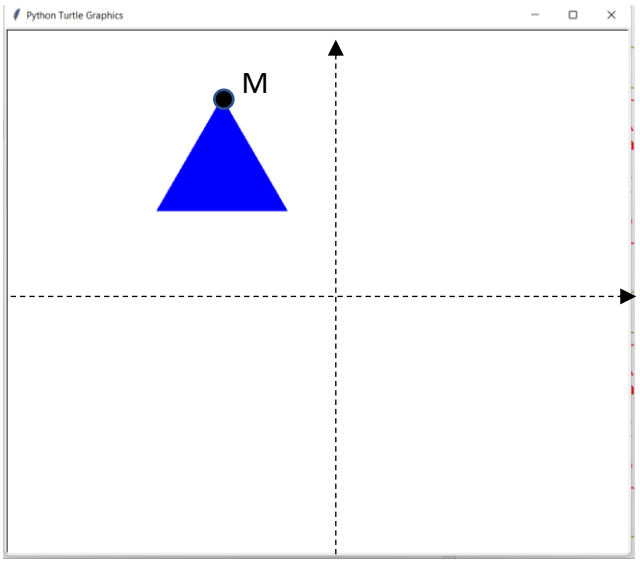
```
>>> type(coul)
<class 'str'>
```

Exercice 2 :

Ecrire un code qui permet de tracer un triangle équilatéral n'importe où dans la fenêtre graphique.

Ce code saisit la coordonnée en $x$ du point M, sa coordonnée en $y$, la longueur $a$ du côté du triangle et la couleur de remplissage.

Après saisie, le triangle est tracé avec son sommet positionné sur le point M.




Aide :

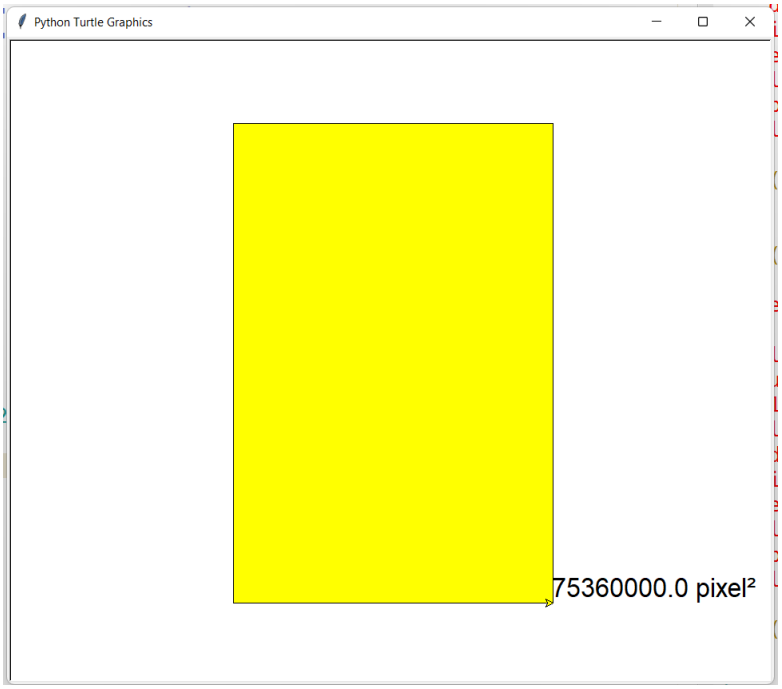
- Les trois cotés du triangle équilatéral ont la même longueur
- Les 3 angles internes du triangle sont de 60°

```
from turtle import *
a = numinput("dimension", "Saisis le coté du triangle")
x = numinput("dimension", "Saisis l'abscisse du sommet")
y = numinput("dimension", "Saisis l'ordonnée du sommet")
coul = textinput("couleur", "Saisis la couleur")
pencolor(coul)
fillcolor(coul)
up()
goto(x,y)
begin_fill()
down()
right(60)
forward(a)
right(120)
forward(a)
right(120)
forward(a)
end_fill()

mainloop()
```

Exercice 3 :

Ecrire un code qui saisit le diamètre $d$ et la hauteur $h$ d'un cylindre. Ce code permet alors :

- De calculer l'aire de ce cylindre en pixel² (on prendra $\pi \approx 3.14$)
- De tracer ce cylindre en couleur, avec les dimensions saisies et positionné au centre de la fenêtre
- D'afficher l'aire calculée sur le coin bas droit du cylindre




Aide :

- Pour afficher le texte dans la fenêtre, on utilisera la fonction *write()* de *turtle*. Par exemple `write(aire,font=("Arial", 20, "normal"))` permet d'afficher le texte *aire = str(75360000.0)+' pixel²'* a coté du pointeur de la tortue.
- Pour arrondir un nombre réel à un chiffre après la virgule, on utilise la fonction *round()* native de python. Par exemple, *round(1.12345 , 2)* renvoie *1.1*

```
from turtle import *
d = numinput("dimension", "Saisis le diamètre du cylindre")
h = numinput("dimension", "Saisis la hauteur du cylindre")
aire = 3.14*d**2/4*h
aire = round(aire,1)
aire = str(aire)+" pixel²"
fillcolor("yellow")
up()
goto(d//2,-h//2)
down()
begin_fill()
goto(d//2,h//2)
goto(-d//2,h//2)
goto(-d//2,-h//2)
goto(d//2,-h//2)
end_fill()

write(aire,font=("Arial", 20, "normal"))
mainloop()
```