SNT - Langage Python et module turtle

Nous allons découvrir comment dessiner des figures géométriques en langage Python à l'aide du module turtle. Ce module est un peu comme une bibliothèque d'instructions ou de commandes utilisables pour exécuter des opérations.

Liste des commandes les plus communes est la suivante :

Commande	Effet
reset()	On efface tout et on recommence
goto(x, y)	Aller à l'endroit de coordonnées x, y
forward(distance)	Avancer d'une distance donnée
backward(distance)	Reculer
up()	Relever le crayon (pour pouvoir avancer sans dessiner)
down()	Abaisser le crayon (pour recommencer à dessiner)
color(couleur)	couleur peut être 'red' , 'blue', etc.
left(angle)	Tourner à gauche d'un angle donné (exprimé en degrés)
right(angle)	Tourner à droite
width(épaisseur)	Choisir l'épaisseur du tracé
begin_fill()	Début de la zone fermée à colorier
end_fill()	Fin de la zone fermée à colorier
write(texte)	texte doit être une chaîne de caractères
done() ou mainloop()	Attend que l'utilisateur ferme la fenêtre

Pour commencer à programmer en langage Python, ouvrons tout d'abord l'interface de développement intégrée IDLE.
Une fenêtre appelée **Console** ou **Interpréteur** s'ouvre avec trois chevrons >>> et un prompt, comme ci-dessous :

```
Python 3.13.7 (v3.13.7:bcee1c32211, Aug 14 2025, 19:10:51) [Clang 16.0.0 (clang-1600.0.26.6)] on darwin
Enter "help" below or click "Help" above for more information.
>>>
```

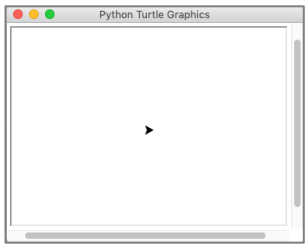
La **Console** permet de communiquer **à la volée** en langage python avec l'ordinateur, ce qui signifie que chaque instruction écrite derrière les chevrons >>> est exécutée dès validation à l'aide de la touche ENTER. La console ne permet pas d'exécuter plusieurs lignes de code et d'écrire des programmes complexes.

Ensuite, cliquer sur New File ou Nouveau Fichier.
Une fenêtre s'ouvre. Il s'agit de ce que l'on appelle l'**Éditeur**. C'est dans cette fenêtre que nous allons écrire notre premier programme, code ou script avec turtle.
Sélectionner ensuite Save ou Enregistrer, puis enregister le fichier sous le nom : turtle_votrenom.py

Un script est un programme ou code qui sera exécuté par l'interpréteur Python, la machine capable de comprendre ou d'interpréter le script écrit. En mode écrire, il est possible d'écrire de nombreuses lignes de code et de lancer (RUN) leur exécution instantanément.

```
"""
Commentaires :
Ce programme est mon premier programme avec turtle
Date : Le vendredi 2 octobre 2026
Auteur :
"""
import turtle    # Il faut importer le module pour pouvoir utiliser ses instructions
turtle.reset()   # La fonction reset() réinitialise la fenêtre de dessin utilisée
turtle.down()    # La fonction down() abaisse la tortue pour que celle-ci dessine
turtle.done()    # La fonction done() maintient ouverte la fenêtre contenant le dessin fait.
```

Une fois le programme ci-dessus écrit dans la fenêtre, cliquer sur RUN pour exécuter le script. Une fenêtre dans laquelle apparaît un curseur s'affiche comme ci-dessous :

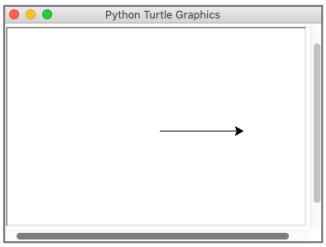


Le script proposé commande la réinitialisation d'une fenêtre de dessin, un abaissement de la tortue et un maintien de l'ouverture de la fenêtre.

Améliorons le script en demandant à la tortue de se déplacer vers l'avant de 100 pixels.

```
import turtle
turtle.reset()
turtle.down()
turtle.forward(100) # L'instruction forward(100) commande un déplacement de 100 pixels vers l'avant.
turtle.done()
```

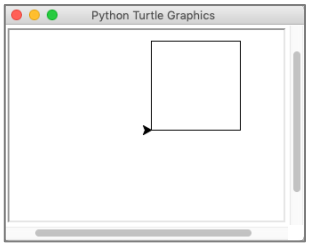
On obtient l'affichage :



En utilisant les commandes de la liste ci-avant, on peut écrire des scripts très simples pour tracer, par exemple, un carré ou un hexagone. Néanmoins, dorénavant, nous ne ferons plus référence au module turtle via son nom complet mais via le raccourci t, bien plus pratique et concis.

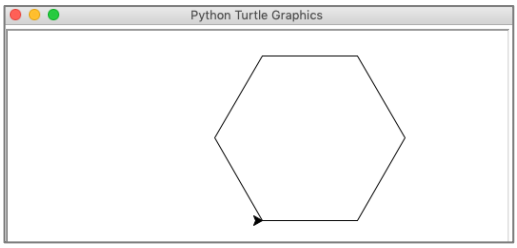
• Script du tracé d'un carré et figure réalisée

```
import turtle as t
t.reset()
t.down()
t.forward(100)
t.left(90)
t.forward(100)
t.left(90)
t.forward(100)
t.left(90)
t.forward(100)
t.left(90)
t.done()
```



• Script du tracé d'un hexagone et figure réalisée
Les instructions forward( ) et left( ) sont remplacées par les raccourcis fd( ) et lt( ).

```
import turtle as t
t.reset()
t.down()
t.fd(100)
t.lt(60)
t.fd(100)
t.lt(60)
t.fd(100)
t.lt(60)
t.fd(100)
t.lt(60)
t.fd(100)
t.lt(60)
t.fd(100)
t.lt(60)
t.done()
```



• Amélioration du script du tracé d'un carré
En utilisant l'instruction de bouclage `for i in range(4):` suivie d'un bloc de code, on réduit sensiblement le nombre de lignes de code du script.

```
import turtle as t
t.reset()
t.down()
for i in range(4):
    t.fd(100)
    t.lt(90)
t.done()
```

