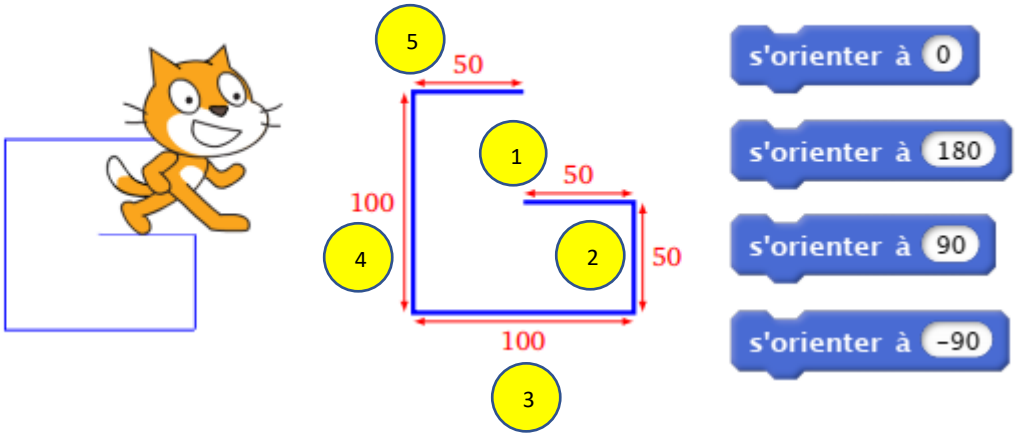
De scratch à python : 1

Exercice 1 :

Trace la figure suivante représentant la lettre « G ».



Avec scratch, les instructions « s'orienter à » permettent de définir la direction vers laquelle le lutin va se déplacer.

S'orienter à 0 → s'orienter vers le haut (Nord)

S'orienter à 90 → s'orienter vers la droite (Est)

S'orienter à 180 → s'orienter vers le bas (Sud)

S'orienter à -90 ou 270 → s'orienter vers la gauche (Ouest)

Avec Python

Pour commencer

Python est constitué d'un ensemble de modules, contenant des données (objets) et des méthodes (fonctions permettant de manipuler les objets).

Afin de ne pas alourdir Python bien des modules ne sont pas chargés par défaut. Mais ils sont déjà installés sur votre ordinateur. Il vous suffit de les activer en les **important** dans votre espace de programmation.

On peut importer l'ensemble des objets et méthodes d'un module. Tout ce qu'il contient devient alors utilisable par notre programme.

On peut aussi n'importer qu'une partie d'un module.

Pour importer l'ensemble d'un module, il y a deux façons de faire.

- **from nom_module import *** : cette façon de faire, très souvent utilisée dans les exemples que l'on peut trouver, est à bannir lorsque l'on est débutant. Elle est dangereuse si l'on ne comprend pas bien le mécanisme mis en place par cette instruction et elle ne permet pas aux environnements modernes permettant de créer des programmes python, de les exécuter et de les tester tel que Spyder, de vérifier dès l'écriture du programme la bonne utilisation des fonctions importées et de signaler les erreurs avant même de tester.

- **import nom_module** : Pour utiliser une méthode du module importé par **import nom_module**, nous devons écrire **nom_module.nom_méthode()**.

  Pour notre programme, nous pouvons redéfinir le **nom** du module, afin par exemple de le raccourcir : **import nom_module as nm**

  Utilisation d'une méthode du module importé sera alors : **nm.nom_méthode()**

Pour n'importer qu'une partie d'un module on écrira :

- **from nom_module import nom_ à importer, nom_ à importer, …**

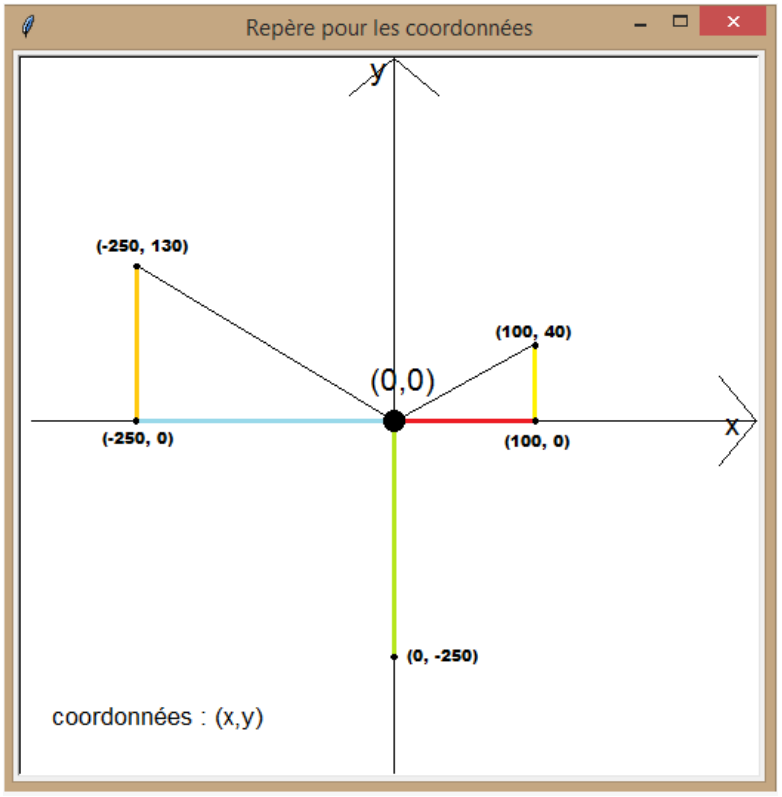
  Pour utiliser le nom de ce qui a été importé, il n'est plus besoin de préfixer ce nom par le nom du module. On utilise ce nom directement.

La tortue python

Python dispose d'une « tortue » qui permet de faire les mêmes dessins que scratch.

Cette tortue provient d'un ancien langage LOGO créé en 1966 par Wally Feurzig and Seymour Papert.

Cette tortue travaille dans un espace, dont nous pouvons donner les dimensions, et qui possède un repère à deux dimensions.



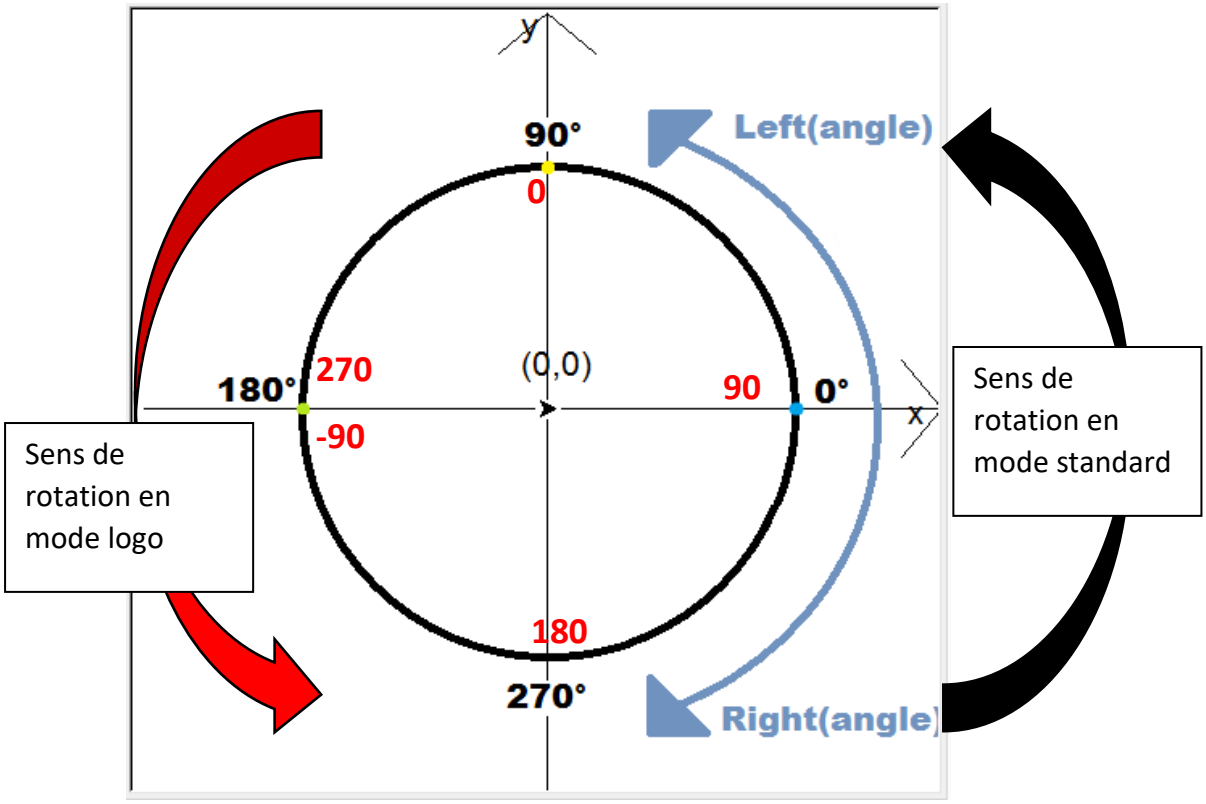
La tortue de python peut utiliser deux modes différents pour orienter la tortue, c'est-à-dire l'inclinaison du trait laissé par la tourtue.

En **mode standard**, on tourne dans le sens contraire des aiguilles d'une montre, et le point de départ est à droite. On part donc de la droite, on monte vers le haut, puis on redescend à gauche, puis on descend vers le bas, et enfin on remonte à droite.

En **mode logo**, on tourne dans le sens des aiguilles d'une montre, et le point de départ est en haut. On part donc du haut, puis on redescend à droite, puis on descend vers le bas, et enfin on remonte à gauche.

Sur le schéma ci-dessous, les angles et la flèche en noir sont ceux du mode standard et ceux en rouge sont ceux du mode logo.

Le **mode logo**, **correspond à la façon dont Scratch oriente les lutins**. Nous choisirons donc ce mode logo



Voici le programme :

import turtle as tu → Module définissant la tortue. Nous le renommons **tu**.

tu.mode('logo') → nous indiquons ici que nous voulons utiliser la façon dont Logo oriente la tortue.

tu.goto(0,0) → 

tu.clear() → 

tu.down() → 

tu.setheading(90) → 

tu.forward(50) → 

tu.setheading(180) #s'orienter vers le sud (vers le bas)→ 

tu.forward(50) #avancer de 50

tu.setheading(-90) #s'orienter vers l'ouest (à gauche) → 

tu.forward(100) #avancer de 100

tu.setheading(0) #s'orienter vers le nord (en haut)→ 

tu.forward(100) #avancer de 100

tu.setheading(90) #s'orienter vers l'est (à droite) → 

tu.forward(100) #avancer de 100

tu.mainloop() #Indique où s'arrête la boucle d'exécution.

tu.bye() #Ferme la fenêtre de la tortue proprement

Au démarrage de votre programme avec le module « turtle » (pour scratch : clic sur le drapeau vert), python met en place une boucle d'exécution et un « écouteur d'évènements » rudimentaire, chargé de détecter si l'utilisateur a cliqué sur le bouton de fermeture de la fenêtre : bouton X en haut à droite (pour scratch : clic sur le drapeau rouge).

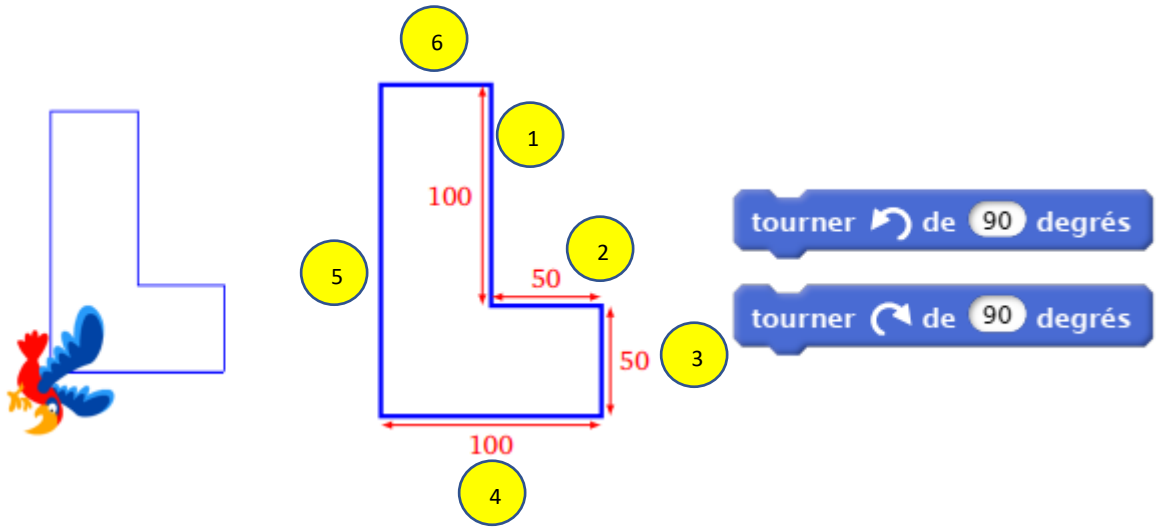
Cet évènement peut arriver n'importe quand, même si la tortue n'a pas fini son travail.

Si l'utilisateur clique sur ce bouton, l'« écouteur d'événement » s'arrête, la boucle d'exécution s'arrête (mainloop()) et le programme exécute ce qui suit tu.mainloop() : ici il exécute tu.bye() qui ferme la fenêtre d'exécution de la tortue en libérant tous les objets créés.

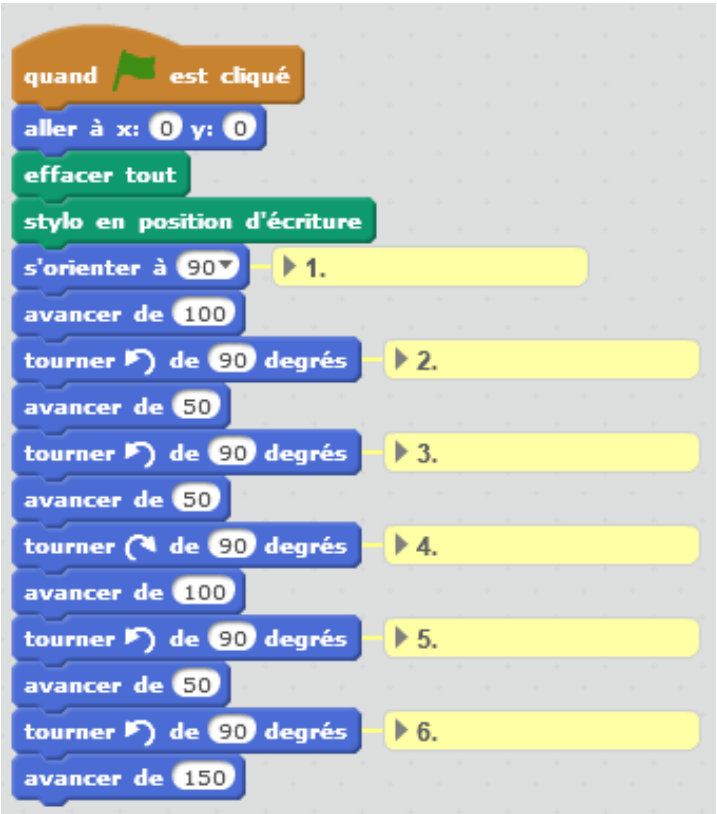
Nous reviendrons plus à fond sur cette boucle d'exécution, dans un autre tutoriel.

Exercice 2 :

Trace la figure suivante représentant un L



Utilise seulement le bloc « Avancer » et le bloc « Tourner vers la droite de 90° » pour tourner d'un quart de tour à droite, ou le bloc « Tourner vers la gauche de 90° » pour tourner d'un quart de tour à gauche.



Avec Python

tu.left(90) #tourner à gauche de 90° → 

tu.right(90) #tourner à droite de 90° → 

Le programme :

```
import turtle as tu
tu.mode('logo')
tu.goto(0,0)          # aller à la position (0,0)
tu.clear()            #effacer tout
tu.down()             #poser le crayon
#1.-------------------------------------------------
tu.setheading(90)     #orientation de la tortue vers l'Est
tu.forward(100)       #avancer de 100
#2.-------------------------------------------------
tu.left(90)           #tourner à gauche de 90°
tu.forward(50)        #favancer de 50
#3.-------------------------------------------------
tu.left(90)           #tourner à gauche de 90°
tu.forward(50)        #avancer de 50
#4.-------------------------------------------------
tu.right(90)          #tourner à droite de 90°
tu.forward(100)       #avancer de 100
#5.-------------------------------------------------
tu.left(90)           #tourner à gauche de 90°
tu.forward(50)        #avancer de 50
#6.-------------------------------------------------
tu.left(90)           #tourner à gauche de 90°
tu.forward(150)       #avancer de 150
#-------------------------------------------------
```

```
tu.mainloop()
tu.bye()
```