Turtle avec Python

1 Environnement de programmation Python

Programmer en Python consiste à écrire des instructions qui respectent les contraintes du langage Python. En pratique, ce programme est écris dans un fichier texte (pas au format Word !! plutôt au format txt).

Nous allons utiliser un environnement de programmation pour programmer en Python : Spyder. L'environnement de programmation offre des possibilités d'édition amélioré d'un programme :

- La coloration syntaxique (les commentaires, les constantes, les mots clés sont en couleur)
- La complétion automatique
- L'intégration du code et de son exécution (utiliser la touche F5)

Pour faire un programme informatique (j'insiste sur le terme pour différencier de certaines pratiques, notamment en statistiques, que la morale informatique réprouve!), vous devez concevoir **un programme comme un tout dans un fichier** : il ne s'agit pas de ligne de codes en vrac !!

Dans la suite, vous allez programmer le robot `turtle`, qui va généré de magnifiques dessins !!

2 Le robot Turtle

Dans cette partie, on s'intéresse à un robot un peu particulier : le robot *turtle*. Ce robot dessine en même temps qu'il se déplace dans son terratorium. Ce robot peut être commandé par les instructions suivantes :

- `left(D)` : tourne la tortue sur elle même de D degrés vers la gauche
- `right(D)` : tourne la tortue sur elle même de D degrés vers la gauche
- `forward(D)` : la tortue avance de D unités
- `pendown()` (resp. `penup()`) : active (resp. désactive) le dessin de trait pendant les déplacements

Voici un exemple complet de programme :

```
import turtle           # import des fonctionnalités de la tortue

silly = turtle.Turtle()  # Création d'une tortue nommée 'silly'

silly.forward(50)   # Avancer de 50 unités
silly.right(90)     # Rotation 90° dans les sens des aiguilles d'une montre

silly.forward(50)
silly.right(90)

silly.forward(50)
silly.right(90)

silly.forward(50)
silly.right(90)

turtle.done()       # fin du dessin
```

Exercice 1 - Écrire, tester et modifier le programme

Q.1) Recopier le programme ci-dessus dans un nouveau programme Python (à quoi servent les '#')

Q.2) Exécuter ce programme pour observer le résultat (utiliser la touche F5). Si des erreurs s'affiche, corriger votre programme.

Q.3) L'exemple ci-dessous illustre comment répéter 100 fois des opérations, recopier et tester

cet exemple. Puis adapter le pour simplifier le programme précédent de la tortue !

```
print(‘punition :’)
for i in range(100) :
        print(‘Je n’ai pas été sage !’)
        print(‘Je dois recopier cette phrase 100 fois’)
```

Exercice 2 - Expérimentations avec Turtle

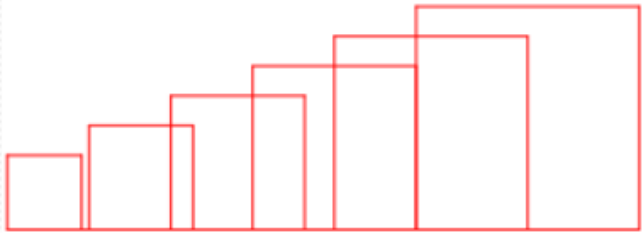
Q.1) Tester le programme exemple `turtle_ex1.py` *de la tortue*

Q.2) Supprimer la partie du programme dessinant le trait de droite, puis modifier la taille des carrés en les passant à une taille de 10

Q.3) On souhaite maintenant répéter l'opération de création d'un carré ***plusieurs fois****, mais en décalant la tortue de 36° vers la droite. Dans l'exemple ci-dessous, 3 rectangles ont été créé, vous le ferez pour 10 (soit 360° pour 1 tour complet)*

Q.4) Revenir au programme de la question 2 et modifier le pour afficher 6 rectangles de taille 10 (séparé de 5) à la suite comme illustré ci-dessous :

Q.5) Modifier ensuite votre programme pour avoir un dessin ressemblant à celui-ci



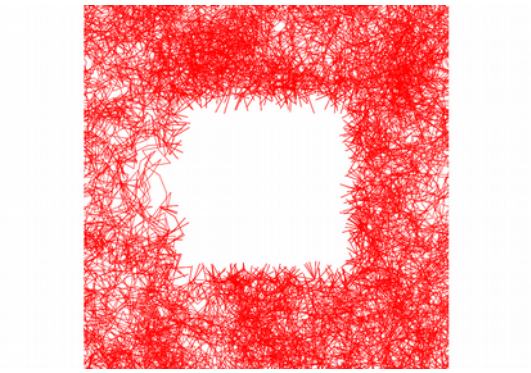
Exercice 3 - Marches aléatoires (mouvement Brownien)

L'instruction ci-dessous donne aléatoirement un nombre entre 0 et 360 (distribution uniforme). Ceci permet de tirer aléatoirement une direction (un angle) pour la tortue :

```
dir = random.randint(1, 360)
```

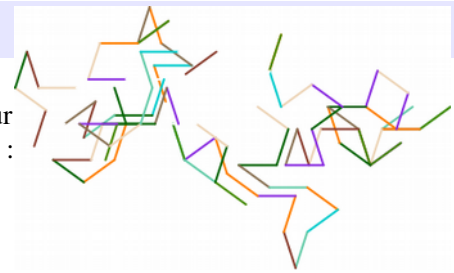
Q.1) Écrire alors un programme qui permette à la tortue de se déplacer « aléatoirement » en suivant le principe suivant de mouvement brownien : la tortue avance de 5 unités de distance, puis elle change aléatoirement de direction. Ceci sera alors répété, par exemple, 1000 fois.

Q.2) (difficile) On souhaite maintenant contraindre la tortue à éviter la région centrale de la zone centrale (cf Illustration ci-dessous)



Pour obtenir les coordonnées `(x,y)` de la tortue, vous pourrez utiliser les instructions ci-dessous :

```
x=silly.xcor()
y=silly.ycor()
```



Vous trouverez finalement un exemple d'utilisation de turtle pour réaliser des Pi-Piquant tel qu'illustré ci-contre (en savoir plus : http://images.math.cnrs.fr/Geometrie-dans-les-spasmes.html)