

Python sur Trinket – Leçon 3

Review


- La librairie Python pour les commandes Tortues
- Commenter
- Gestions des erreurs

Plan de la leçon

- Guider les élèves dans la création de programmes en commençant par la création d'un flocon de neige pour ensuite introduire les boucles, les variables et les nombres entiers.
 1. Dendrites (bras) d'un flocon de neige
 2. Flocon de neige complet
 - Démontrer les changements pour améliorer le programme : ajouter des boucles, des couleurs et des nombres entiers aléatoires
 3. Amener les élèves à utiliser les boucles et les variables afin de créer des formes uniques

Leçon

Initialiser le programme
<code>import turtle as t</code>
<code>import random</code>
<code>t.shape("turtle")</code>
<code>t.speed(speed)</code>

1. Dendrites (Bras) du flocon de neige	
<pre> t.forward(50) t.stamp() t.backward(20) t.left(45) t.forward(20) t.backward(20) t.right(90) t.forward(20) t.backward(20) t.left(45) t.backward(30) </pre>	

Montrer les changements suivants afin d'améliorer la fonctionnalité de notre programme

- Introduire la notion de boucles et de variable (afin que les élèves puissent faire un flocon avec plusieurs dendrites)
 - Les boucles nous permettent d'exécuter un code à plusieurs reprises (pour la création d'un flocon de neige, nous ne voulons pas réécrire les mêmes lignes de codes pour chaque dendrite. En utilisant une boucle, nous pouvons ainsi indiquer le nombre de fois que nous voulons répéter les lignes de code choisies.
 - Si l'on veut utiliser une valeur plus d'une fois, il est possible de la sauvegarder en tant que variable. Ainsi, si nous voulons modifier la valeur, nous n'avons qu'à modifier la ligne où nous précisons la nature de la variable (au lieu de modifier toutes les lignes où la valeur est écrite).

```
for count in range(10):  
    ligne de code...  
    ligne de code...  
    t.left(360/10)
```

Si nous voulons 15 dendrites au lieu de 10, nous aurions à trouver et à modifier toutes les lignes de code où le nombre 10 apparaît. Or, avec la valeur transformée variable, nous n'avons qu'à modifier la ligne où nous définissons la variable.

```
num_arms = 10  
for count in range(num_arms):  
    lines of code...  
    lines of code...  
    t.left(360/num_arms)
```


```
num_arms = 15    # seulement un changement de ligne  
for count in range(num_arms):  
    lines of code...  
    lines of code...  
    t.left(360/num_arms)
```

- for count in range (10) **vs.** for count in range (num_arms)
- t.left(360/10) **vs.** t.left(360/num_arms)

- Nombres entiers aléatoires (afin de créer des dendrites aux multiples couleurs)

```
''' Get a random value for red, green, and blue within the
numeric range of colours (0-255) '''
# t.color(r, g, b)
t.color(random.randint(0, 255), random.randint(0, 255), random.randint(0, 255))
```

- Angles – après avoir créé une dendrite, tourner à $(360^\circ / \text{nombres de dendrites})$ pour avoir une distance égale entre les dendrites.

2a. Flocon de neige (+ boucles, valeur aléatoire, rotation)	
<pre>for count in range(10): t.color(random.randint(0, 255), random.randint(0, 255), random.randint(0, 255)) t.forward(50) t.stamp() t.backward(20) t.left(45) t.forward(20) t.backward(20) t.right(90) t.forward(20) t.backward(20) t.left(45) t.backward(30) t.left(36)</pre>	

2b. Flocon de neige (+ variable)	
<pre>size = 10 for count in range(size): t.color(random.randint(0, 255), random.randint(0, 255), random.randint(0, 255)) t.forward(50) t.stamp() t.backward(20) t.left(45) t.forward(20) t.backward(20) t.right(90) t.forward(20) t.backward(20) t.left(45) t.backward(30) t.left(360/size)</pre>	