TP2 : Optimisation des scripts, boucles for, input et fonction

Maintenant qu'on a fait quelques dessins, il est temps de rendre notre code un peu plus "pro". On va utiliser des boucles et des entrées utilisateurs pour le rendre plus efficace.

1. Optimisation avec boucle for

Voici le code pour optimiser notre script rectangle et qu'il prenne moins de place :

Sans boucle for : ce que tu as dû faire	Avec boucle for (à tester)

`from turtle import *`
 	
`forward(100)`
`left(90)`
`forward(50)`
`left(90)`
`forward(100)`
`left(90)`
`forward(50)`
 	
`hideturtle()`
	`from turtle import *`
	
	`for i in range(0, 4):`
	`    forward(100)`
	`    left(90)`
	`    forward(50)`
	`    left(90)`

Exercice :
Modifie les scripts du triangle et du carré sans coin pour utiliser la boucle `for i in range(…,…)`pour le triangle et pour le carré sans coin.

2. Utilisation de input() pour personnaliser les dimensions

Et si vous pouviez demander à l'utilisateur de choisir la longueur et la largeur du rectangle, plutôt que de les définir vous-même ? C'est là qu'intervient `input()` !

Exemple à recopier et tester:

```
from turtle import *

longueur = int(input("Entrez la longueur du rectangle : "))
largeur = int(input("Entrez la largeur du rectangle : "))
couleur = input("De quelle couleur veux-tu ton rectangle ? "))

for i in range(0, 4):
    forward(100)
    left(90)
    forward(50)
    left(90)

hideturtle()
```

Analyse du script:
`longueur = int(input("Entrez la longueur du rectangle : "))` : demande à l'utilisateur « Entrez la longueur du triangle » et stock sa réponse sous la variable longueur.

Remarque importante !!!!
le `int(…)`, permet de convertir la réponse de l'input qui par défaut est une chaîne de caractères, en un entier (integer) utilisable par le script (voir le TP Découverte des types de données en Python)
Exercice : Ajoute des entrées utilisateur dans tes scripts du triangle et du carré sans coin, pour que les utilisateurs puissent définir la taille des figures.

3. Création de fonctions avec def

Python te permet de créer des fonctions pour réutiliser du code. Voici un exemple pour le rectangle. Crée un fichier nommé programme_rectangle.py et implémente une fonction qui trace un rectangle avec des dimensions personnalisées.

```
from turtle import *

def rectangle(longueur, largeur):
    for i in range(0, 4):
        forward(100)
        left(90)
        forward(50)
        left(90)
```

Appuyer sur run pour sauvegarder ce programme. Pour l'appeler, va sur la console, tape : `rectangle(150, 70)`, puis entrée, et hop…

Remarque :
On peut mettre d'autres paramètres pour gérer les couleurs.

```
from turtle import *

def rectangle(longueur, largeur, couleur):
    color(couleur)
    for i in range(0, 4):
        forward(100)
        left(90)
        forward(50)
        left(90)
```

Exercice :

Crée deux nouveaux fichiers : `triangle.py` et `carre_sans_coin.py`, et implémente des fonctions pour tracer le triangle équilatéral et le carré sans coin avec les dimensions souhaitées par l'utilisateur.