

# Exercice : Boucles en Python

---

## Exercice 1

- 1) Donner dans chaque cas les valeurs renvoyées par la boucle **for**.

a)

```
for m in range(5):  
    print(m)
```

b)

```
for k in range(4,8):  
    print(k)
```

c)

```
for nombre in range(1,19,3):  
    print(nombre)
```

d)

```
for i in range(5):  
    print(i*10-1)
```

- 2) Écrire une boucle **for** pour obtenir la suite de nombres (sans les virgules) dans chaque cas :

- a) 5, 6, 7, 8, 9
- b) 0, 2, 4, 6, 8, 10, 12, 14, 16, 18
- c) -1, -2, -3, -4, -5
- d) 4, 9, 16, 25, 36

## Exercice 2

Écrire chacune des boucles **for** de l'exercice 1 avec des boucles **while** lorsque cela est possible.

## Exercice 3

- 1) Créer une boucle **FOR** qui affiche 4 fois le message "Bienvenue"
- 2) Modifier votre programme pour que le message soit affiché autant de fois qu'une valeur saisie avant la boucle.
- 3) Modifier enfin votre code pour que le message soit numéroté :  
Bienvenue 1 fois  
Bienvenue 2 fois  
etc.

## Exercice 4

La fonction **print** affiche un contenu et passe à la ligne. On peut modifier ce comportement en ajoutant un paramètre avec le message.

Par exemple : `print("Bienvenue", end="-")`

Dans ce cas, à la fin de mot affiché Bienvenue, on a un tiret et on reste sur la même ligne pour l'affichage suivant.

- 1) Créer une boucle **FOR** qui affiche sur une même ligne les nombres de 0 à 9 séparés par un tiret : 0 - 1 - 2 - ...
- 2) Modifier votre boucle pour qu'elle affiche les nombres en commençant à 5 : 5 - 6 - 7 - ...
- 3) Modifier votre boucle qu'elle affiche les nombres progressant de 3 en 3 : 0 - 3 - 6 - ...
- 4) Modifier votre boucle pour afficher les nombres dans l'ordre décroissant de 9 à 0 : 9 - 8 - 7 - 6 - ...

## Exercice 5

- 1) Écrire un programme qui affiche les nombres de 0 à 99 sur 10 lignes, chaque nombre étant séparé par un tiret.
- 2) Proposer une seconde version afin qu'il n'y ait pas de tiret à la fin de chaque ligne.

## Exercice 6

- 1) On souhaite réaliser le motif en "X" ci-dessous. Il est composé de 10 lignes et 10 colonnes.

```
XXXXXXXXXX
XXXXXXXXXX
XXXXXXXXXX
XXXXXXXXXX
XXXXXXXXXX
XXXXXXXXXX
XXXXXXXXXX
XXXXXXXXXX
XXXXXXXXXX
XXXXXXXXXX
```

- a) Créer une première boucle qui permet de créer une première ligne. Pour afficher du texte sur une même ligne, il faut ajouter le paramètre **end** à la fonction **print** : `print(message, end="")`.
- b) Créer une boucle qui va répéter 10 fois la boucle précédente. On dit que ce sont des boucles imbriquées.
- 2) Modifier votre code pour que le nombre de lignes et colonnes soit saisi par l'utilisateur.
- 3) Réaliser les motifs suivants de 10 lignes et 10 colonnes :

a)

```
XXXXXXXXXX
X        X
X      X
X      X
X      X
X      X
X      X
X      X
X      X
X      X
X      X
XXXXXXXXXX
```

b)

```
HIHIHIHIHIHIHIHIHIHI
IHIHIHIHIHIHIHIHIHI
HIHIHIHIHIHIHIHIHIHI
IHIHIHIHIHIHIHIHIHI
HIHIHIHIHIHIHIHIHIHI
IHIHIHIHIHIHIHIHIHI
HIHIHIHIHIHIHIHIHIHI
IHIHIHIHIHIHIHIHIHI
HIHIHIHIHIHIHIHIHIHI
IHIHIHIHIHIHIHIHIHI
HIHIHIHIHIHIHIHIHIHI
```

c)

```
XXXXXXXXXX
X      X
X      X
X      X
XX
XX
X      X
X      X
X      X
X      X
XXXXXXXXXX
```

## Exercice 7

- 1) Écrire un premier programme qui calcule et affiche  $1 + 2 + 3 + \dots + 100$ .
- 2) Écrire un second programme qui calcule et affiche  $1 \times 2 \times \dots \times 100$ .
- 3) Modifier les deux programmes précédents pour calculer et afficher les résultats jusqu'à un nombre  $N$  saisi par l'utilisateur.

## Exercice 8

Les programmes seront appliqués sur la chaîne de caractères suivante :

Le langage python a été créé par Guido Van Rossum.

- 1) Écrire un programme qui affiche les voyelles contenues dans la chaîne de caractères.
- 2) Écrire un programme qui affiche la chaîne de caractères en lettres capitales.
- 3) Écrire un programme qui affiche la chaîne de caractères écrite à l'envers (lettre par lettre).

## Exercice 9

- 1) Saisir dans l'interpréteur la fonction `bin(9)`. Quel est le résultat affiché ?
- 2) L'objectif est de créer une boucle **WHILE** qui donne l'écriture binaire d'un nombre entier positif en utilisant les divisions successives.
- a) On utilisera les variables **n** pour le nombre entier positif, **b** pour l'écriture binaire obtenue qui sera de type "string" et **r** pour les restes des divisions entières.
- b) On calculera les divisions en mémorisant le reste dans une variable **r** et le nouveau quotient dans **n**.
- c) Le nombre **n** est divisé par 2 jusqu'à devenir nul. On fait les calculs tant que **n** est différent de 0.
- d) On construit le nombre binaire **b** comme chaîne de caractères constituée des restes.
- 3) On pourra faire une vérification avec la fonction `int(chaine,base)`.