

La boucle for

L'objectif des boucles est de **répéter plusieurs fois** un ensemble d'instructions. Chaque répétition de ces instructions est appelée **itération**.

Il existe deux types de boucle, pour le moment nous allons nous intéresser uniquement à la boucle inconditionnelle **for** (pour en français). Celle-ci permet de répéter des plusieurs instructions une **nombre fini de fois**, connu dès la première itération.

La syntaxe d'une boucle **for** est la suivante :

```
for variable_de_boucle in range(...):  
    # Ici les instructions a répéter.  
    # Elles sont indentées par rapport à l'instruction for.  
# les instructions ici ne sont pas dans la boucle
```

Deux informations sont présentes dans l'instruction **for** :

- La **variable de boucle** est une variable dont la valeur va changer à chaque itération. On lui donne généralement les noms **i**, **j**, ou **k**. Certains algorithmes n'ont pas besoin de cette variable et dans ces cas là, on peut l'anonymiser et la nommer **_** (tiré du bas, underscore).
- Le **nombre de répétitions** est renseigné au moyen des parenthèses de **range**. Il y existe trois appels possibles :

- Avec **range(fin)**, la boucle va effectuer **fin - 1** itérations et la variable de boucle va prendre les valeurs entières de **0** à **fin - 1**.

```
for i in range(3):  
    print(i)          # Affiche successivement 0, 1, 2
```

- Avec **range(debut, fin)**, la boucle va effectuer **fin - debut - 1** itérations et la variable de boucle va prendre les valeurs entières de **debut** à **fin - 1**.

```
for j in range(2, 4):  
    print(j)          # Affiche successivement 2, 3
```

- Avec **range(debut, fin, pas)**, la variable de boucle va prendre les valeurs de **debut** à **fin - 1** en évoluant de **pas** en **pas**.

```
for i in range(1, 10, 2):  
    print(i)          # Affiche successivement 1, 3, 5, 7, 9
```

```
for j in range(-5, -10, -1):  
    print(j)          # Affiche successivement -5, -6, -7, -8, -9
```

```
for k in range(10, 0, -2):  
    print(k)          # Affiche successivement 10, 8, 6, 4, 2
```

Exercice 1 : Prédiction, run, investigate

Sans exécuter, qu'affichent les codes suivants ? Vérifier en exécutant le code.

<pre># Code 1 for _ in range(4): print("bonjour")</pre>	<pre># Code 2 for _ in range(4, 9): print("hello")</pre>
<pre># Code 3 for i in range(10, 7, -1): print(100+i)</pre>	<pre># Code 4 s = 0 for i in range(4): s = s + i print(s)</pre>
<pre># Code 5 for _ in range(3): for _ in range(3): print("hello")</pre>	<pre># Code 6 for i in range(3): for j in range(3, 6): print(i, j)</pre>

Exercice 2 : Modify

1. En modifiant le code 3 de l'exercice 1, écrire un programme qui affiche la table de 2 (de 1 à 10) avec une boucle for. Chaque ligne doit être de la forme $2 * \dots = \dots$.
2. En modifiant le code 3 de l'exercice 1, écrire un programme qui demande à l'utilisateur un entier n affiche la table de n (de 1 à 10) avec une boucle for. Chaque ligne doit être de la forme $n * \dots = \dots$.
3. En modifiant le code 4 de l'exercice 1, écrire un programme qui calcule la moyenne des entiers de 1 à 100 inclus puis affiche le résultat.

Exercice 3 : Make

1. Écrivez un programme avec une boucle qui affiche pour chaque entier entre 1 et 10 si celui-ci est pair ou impair.
2. Ana achète une voiture d'une valeur de 60 000€. La valeur de la voiture baisse de 15% par an. Écrire un programme qui à l'aide d'une boucle for affiche la valeur de la voiture au bout de 10 ans.
3. Écrivez un programme avec une boucle qui affiche sur plusieurs lignes le décompte 5, 4, 3, 2, 1, BOOM. On pourra s'aider de la fonction Python `time.sleep(secs)` où `secs` représente le nombre de secondes à attendre. Pour l'utiliser, il faut ajouter la ligne `import time` en haut de votre code.
4. La somme des carrés des entiers impairs de 1 à 10 est $1^2+3^2+5^2+7^2+9^2 = 165$. Écrire un programme qui affiche la somme des carrés des entiers impairs de 1 à 10000.
5. Écrire un programme qui demande à l'utilisateur un entier n qui retourne le nombre de diviseurs de n . Par exemple, 6 à 4 diviseurs : 1, 2, 3, 6.
6. a) Écrivez un programme qui affiche tous les nombres entre 1 et 100 avec les exceptions suivantes :
 - Il affiche "Fizz" à la place du nombre si celui-ci est divisible par 3.
 - Il affiche "Buzz" à la place du nombre si celui-ci est divisible par 5 et non par 3.b) Ensuite, améliorez votre programme pour qu'il affiche "FizzBuzz" à la place des nombres divisibles à la fois par 3 et par 5.

7. Bob achète un appartement d'une valeur de 112 000€. La valeur de l'appartement augmente de 2% par an. Écrire un programme en python qui à l'aide d'une boucle for affiche la valeur de l'appartement au bout de 15 ans.
8. On prend une feuille de papier de 0,1 mm d'épaisseur. Écrire un programme qui à l'aide d'une boucle for affiche l'épaisseur au bout de 20 pliages. Le résultat sera donné en mètre.
9. Écrire un programme en python pour calculer la moyenne de n notes entrées par l'utilisateur, le nombre n de notes étant choisi par l'utilisateur.
10. Écrire les programmes permettant d'obtenir les sorties suivantes avec des boucles.

Sortie 1

```
*  
**  
***  
****  
*****
```

Sortie 2

```
*****  
****  
***  
**  
*
```

Code d'aide

```
txt=""  
for _ in range(...):  
    txt = txt + "**"
```