16_python_for_loops

November 27, 2023

1 Python For loops

En programmation, les boucles sont utilisées pour répéter un bloc de code autant de fois que nécessaire.

En Python, nous avons deux types de boucles: - `for` loop, qui est utilisé pour itérer sur une séquence (liste, tuple, dictionnaire, ensemble, chaîne) - `while` loop, qui est utilisé pour exécuter un bloc de code tant qu'une condition est vraie

[ ]:
```
languages = ['Swift', 'Python', 'Go', 'JavaScript']

# run a loop for each item of the list
for language in languages:
    language = language.upper()
    print(language)
```

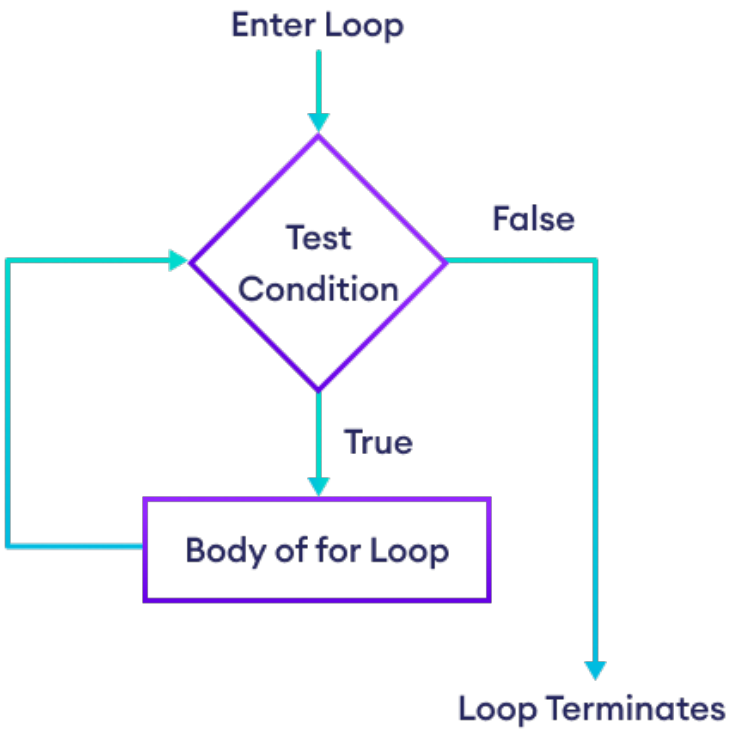
Dans l'exemple ci-dessus, nous avons utilisé une boucle `for` pour imprimer chaque élément de la liste `languages`.

La syntaxe de la boucle `for` est la suivante:

```
for element in sequence:
    # do something with element
```

- `element` est une variable qui contient l'élément de la séquence à chaque itération (l'élément courrant)

1.1 Flowchart de la boucle `for`



1.1.1 Exemple 1: boucler a travers une chaîne de caractères

[ ]:
```
for x in '           ':
    print(x, end=" ")
```

2 Python for loop with range()

La fonction `range()` est utilisée pour générer une séquence de nombres.

La syntaxe de la fonction `range()` est la suivante:

`range(start, stop, step)`

- `start` est un nombre entier qui indique le début de la séquence (par défaut 0)
- `stop` est un nombre entier qui indique la fin de la séquence (non inclus)
- `step` est un nombre entier qui indique le pas de la séquence (par défaut 1)

[ ]:
```
students = ["Nicolas", "Geogios", "Homan", "Elena"]
for i in range(len(students)):
    print(students[i])

for student in (students):
    print(student)
```

3 Python for loop sans accéder aux éléments

Parfois, nous n'avons pas besoin d'accéder aux éléments de la séquence, nous voulons juste répéter un bloc de code un certain nombre de fois.

Dans ce cas, nous pouvons utiliser la fonction `range()` avec la boucle `for` comme ceci:

```
for _ in range(5):
    # do something
```

[ ]:
```
languages = ['Swift', 'Python', 'Go']

for _ in True:
    print('Hello')
    print('Hi')
    print(_)
```

4 Utilisation de la fonction enumerate()

La fonction `enumerate()` est utilisée pour obtenir l'index et l'élément de la séquence à chaque itération. La syntaxe de la fonction `enumerate()` est la suivante:

```
for index, element in enumerate(sequence):
    # do something with index and element
```

- `index` est un nombre entier qui indique l'index de l'élément dans la séquence
- `element` est l'élément courrant de la séquence

[ ]:
```
#print(languages)
print(enumerate(languages))
for i, language in enumerate(languages):
    print(i, language)
```

5 Python for loop with else

La clause `else` est facultative et sera exécutée si la boucle se termine sans interruption.

[ ]:
```
digits = [0, 1, 5]

for i in digits:
    print(i)
    if i == 1:
        break
else:
    print("No items left.")
```

Note: La clause `else` ne sera pas exécutée si la boucle se termine avec `break`.