**Python for Loop**

In computer programming, loops are used to repeat a block of code.For example, if we want to show a message **100** times, then we can use a loop. It's just a simple example; you can achieve much more with loops.

There are 2 types of loops in Python:

* [for loop](https://www.programiz.com/python-programming/for-loop)
* [while loop](https://www.programiz.com/python-programming/while-loop)

**Python for Loop**

In Python, a for loop is used to iterate over sequences such as [lists](https://www.programiz.com/python-programming/list), [tuples](https://www.programiz.com/python-programming/tuple), [string](https://www.programiz.com/python-programming/string), etc. For example,

languages = ['Swift', l'Python', 'Go', 'JavaScript']

# run a loop for each item of the list

for language in languages:

print(language)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Swift

Python

Go

JavaScript

In the above example, we have created a list called languages.

Initially, the value of language is set to the first element of the array,i.e. Swift, so the print statement inside the loop is executed.

language is updated with the next element of the list, and the print statement is executed again. This way, the loop runs until the last element of the list is accessed.

**for Loop Syntax**

The syntax of a for loop is:

for val in sequence:

# statement(s)

Here, val accesses each item of sequence on each iteration. The loop continues until we reach the last item in the sequence.

**Flowchart of Python for Loop**

Working of Python for loop

**Example: Loop Through a String**

for x in 'Python':

print(x)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

P

y

t

h

o

n

**Python for Loop with Python range()**

A [range](https://www.programiz.com/python-programming/methods/built-in/range) is a series of values between two numeric intervals.

We use Python's built-in function range() to define a range of values. For example,

values = range(4)

Here, **4** inside range() defines a range containing values **0, 1, 2, 3.**

In Python, we can use for loop to iterate over a range. For example,

# use of range() to define a range of values

values = range(4)

# iterate from i = 0 to i = 3

for i in values:

print(i)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

0

1

2

3

In the above example, we have used the for loop to iterate over a range from **0** to **3**.

The value of i is set to **0** and it is updated to the next number of the range on each iteration. This process continues until **3** is reached.

|  |  |  |
| --- | --- | --- |
| Iteration | Condition | Action |
| 1st | True | 0 is printed. i is increased to **1**. |
| 2nd | True | 1 is printed. i is increased to **2**. |
| 3rd | True | 2 is printed. i is increased to **3**. |
| 4th | True | 3 is printed. i is increased to **4**. |
| 5th | False | The loop is terminated |

**Note**: To learn more about the use of for loop with range, visit [Python range()](https://www.programiz.com/python-programming/methods/built-in/range).

**Using a for Loop Without Accessing Items**

It is not mandatory to use items of a sequence within a for loop. For example,

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

for language in languages:

print('Hello')

print('Hi')

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Hello

Hi

Hello

Hi

Hello

Hi

Here, the loop runs three times because our list has three items. In each iteration, the loop body prints 'Hello' and 'Hi'. The items of the list are not used within the loop.

If we do not intend to use items of a sequence within the loop, we can write the loop like this:

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

for \_ in languages:

print('Hello')

print('Hi')

[Run Code](https://www.programiz.com/python-programming/online-compiler)

The \_ symbol is used to denote that the elements of a sequence will not be used within the loop body.

**Python for loop with else**

A for loop can have an optional else block. The else part is executed when the loop is exhausted (after the loop iterates through every item of a sequence). For example,

digits = [0, 1, 5]

for i in digits:

print(i)

else:

print("No items left.")

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

0

1

5

No items left.

Here, the for loop prints all the items of the digits list. When the loop finishes, it executes the else block and prints No items left.

**Note**: The else block will not execute if the for loop is stopped by a [break](https://www.programiz.com/python-programming/break-continue) statement.