**Python break and continue**

* break exits the loop entirely
* continue skips the current iteration and proceeds to the next one

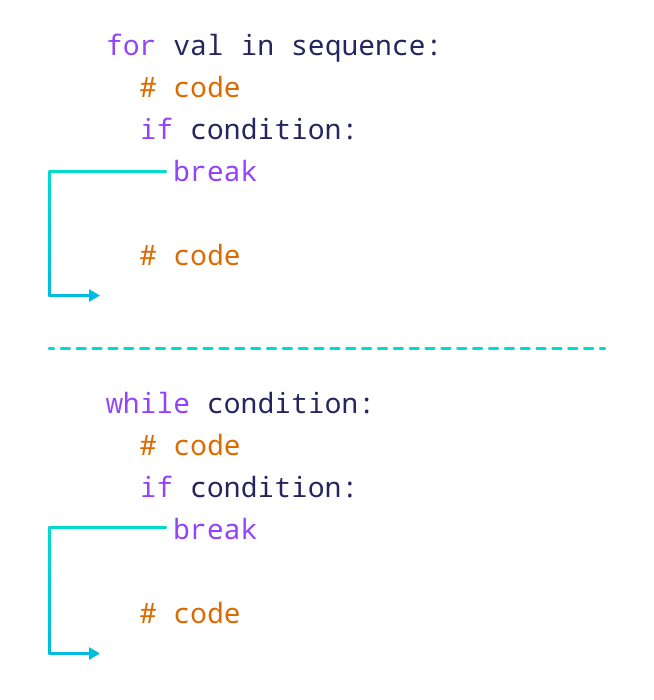
**Python break Statement**

The break statement terminates the loop immediately when it's encountered.

**Syntax**

break

**Working of Python break Statement**



**Example: break Statement with for Loop**

We can use the break statement with the for loop to terminate the loop when a certain condition is met. For example,

for i in range(5):

if i == 3:

break

print(i)

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

**Output**

0

1

2

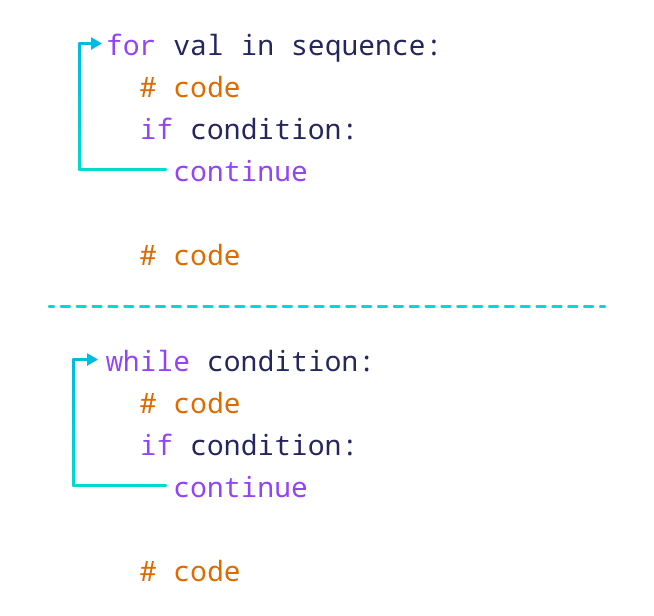
**Python continue Statement**

The continue statement skips the current iteration of the loop and the control flow of the program goes to the next iteration.

**Syntax**

continue

**Working of continue Statement in Python**



**Example: continue Statement with for Loop**

We can use the continue statement with the for loop to skip the current iteration of the loop and jump to the next iteration. For example,

for i in range(5):

if i == 3:

continue

print(i)

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

**Output**

0

1

2

4

**Note:** We can also use the continue statement with a while loop.

**Creating a Function**

In Python a function is defined using the def keyword:

**def** my\_function():  
  print("Hello from a function")

**Calling a Function**

To call a function, use the function name followed by parenthesis:

**my\_function()**

**Arguments**

Information can be passed into functions as arguments.

def my\_function(**fname**):  
  print(fname + "Roy")  
  
my\_function(**"Emil"**)  
my\_function(**"Tobias"**)  
my\_function(**"Linus"**)

**Return Values**

To let a function return a value, use the return statement:

def my\_function(x):  
  **return 5 \* x**

print(my\_function(9))