Python break and continue:

What is the use of break and continue in Python?

In Python, break and continue statements can alter the flow of a normal loop.

Loops iterate over a block of code until test expression is false, but sometimes we wish to terminate the current iteration or even the whole loop without checking test expression.

The break and continue statements are used in these cases

Python break statement

The break statement terminates the loop containing it.

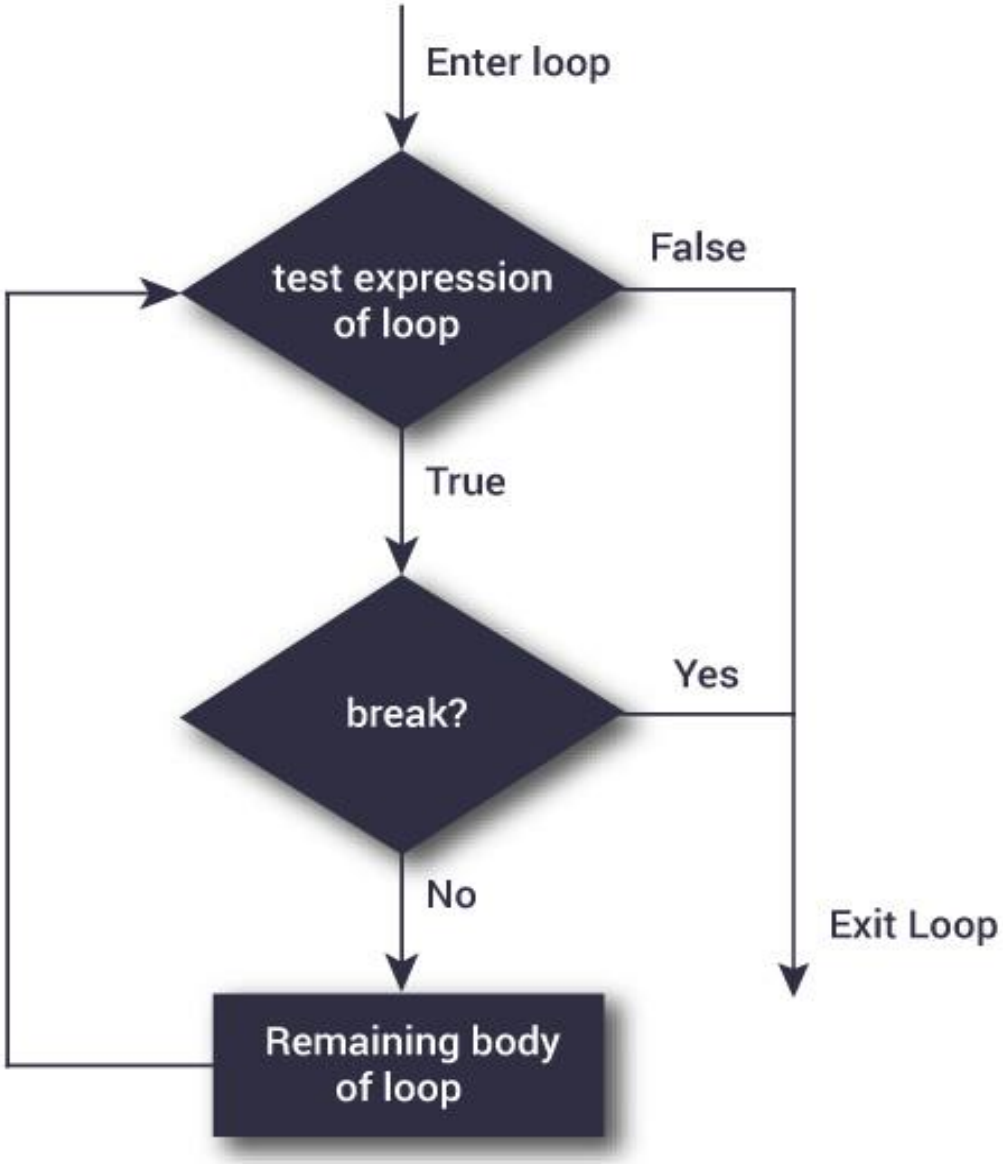
Control of the program flows to the statement immediately after the body of the loop.

If break statement is inside a nested loop (loop inside another loop), break will terminate the innermost loop.

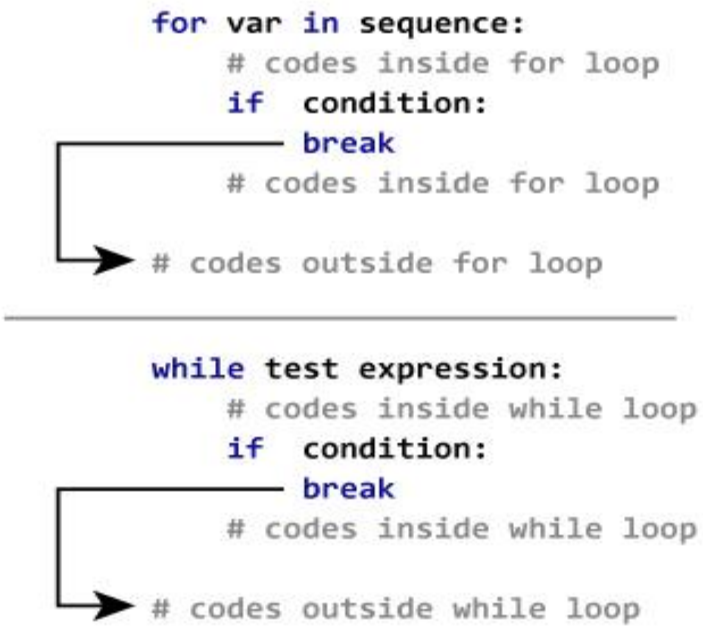
Syntax of break

```
break
```

Flowchart of break



The working of break statement in for loop and while loop is shown below.



Example: Python break

script.py IPython Shell

```
1  # Use of break statement inside loop
2
3  for val in "string":
4      if val == "i":
5          break
6      print(val)
7
8  print("The end")
```

Output

```
s

t

r

The end
```

In this program, we iterate through the `"string"` sequence. We check if the letter is `"i"`, upon which we break from the loop. Hence, we see in our output that all the letters up till `"i"` gets printed. After that, the loop terminates.

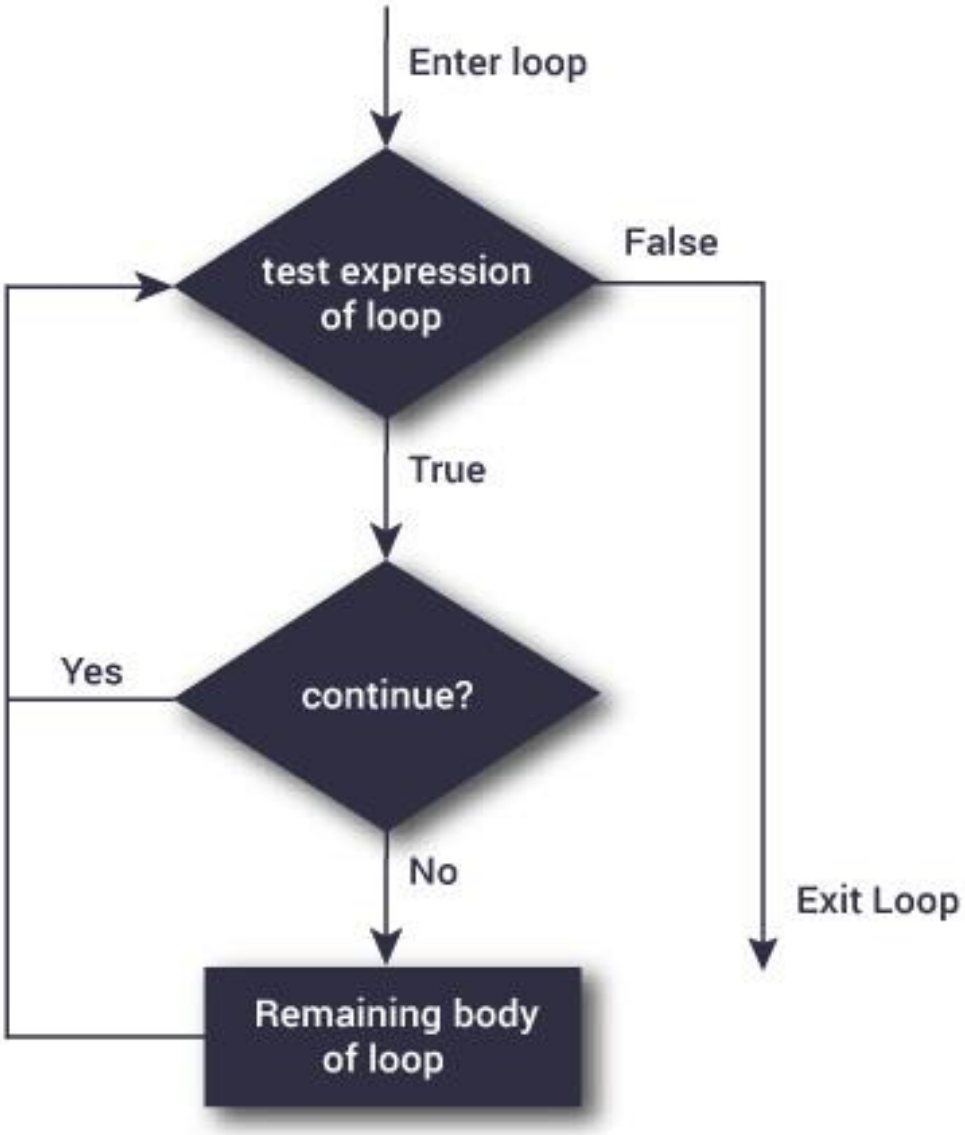
Python continue statement

The continue statement is used to skip the rest of the code inside a loop for the current iteration only. Loop does not terminate but continues on with the next iteration.

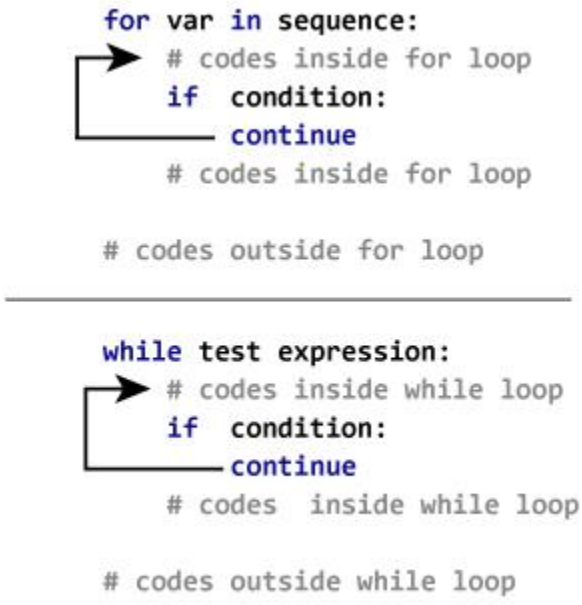
Syntax of Continue

```
continue
```

Flowchart of continue



The working of continue statement in for and while loop is shown below.



Example: Python continue

script.py IPython Shell

```
1  # Program to show the use of continue statement inside loops
2
3  for val in "string":
4      if val == "i":
5          continue
6      print(val)
7
8  print("The end")
```

Output

```
s

t

r

n

g

The end
```

This program is same as the above example except the break statement has been replaced with continue.

We continue with the loop, if the string is `"i"`, not executing the rest of the block. Hence, we see in our output that all the letters except `"i"` gets printed.

Python Program to Check Prime Number

Example to check whether an integer is a prime number or not using for loop and if...else statement. If the number is not prime, it's explained in output why it is not a prime number.

To understand this example, you should have the knowledge of following Python programming topics:

- **Python if...else Statement**
- **Python for Loop**
- **Python break and continue**

A positive integer greater than 1 which has no other factors except 1 and the number itself is called a prime number. 2, 3, 5, 7 etc. are prime numbers as they do not have any other factors. But 6 is not prime (it is composite) since, 2 x 3 = 6.

Source Code

script.py IPython Shell

```
# Python program to check if the input number is prime or not

num = 407

# take input from the user
# num = int(input("Enter a number: "))

# prime numbers are greater than 1
if num > 1:
   # check for factors
   for i in range(2,num):
       if (num % i) == 0:
           print(num,"is not a prime number")
           print(i,"times",num//i,"is",num)
           break
   else:
       print(num,"is a prime number")

# if input number is less than
# or equal to 1, it is not prime
else:
   print(num,"is not a prime number")
```

The *pass* Statement:

The **pass** statement in Python is used when a statement is required syntactically but you do not want any command or code to execute.
The **pass** statement is a *null* operation; nothing happens when it executes. The **pass** is also useful in places where your code will eventually go, but has not been written yet (e.g., in stubs for example):

Example:

```
#!/usr/bin/python

for letter in 'Python':
   if letter == 'h':
      pass
      print 'This is pass block'
   print 'Current Letter :', letter

print "Good bye!"
```

This will produce following result:

```
Current Letter : P
Current Letter : y
Current Letter : t
This is pass block
Current Letter : h
Current Letter : o
Current Letter : n
Good bye!
```

The preceding code does not execute any statement or code if the value of *letter* is 'h'. The *pass*statement is helpful when you have created a code block but it is no longer required.

You can then remove the statements inside the block but let the block remain with a pass statement so that it doesn't interfere with other parts of the code.