# **Python break statement**

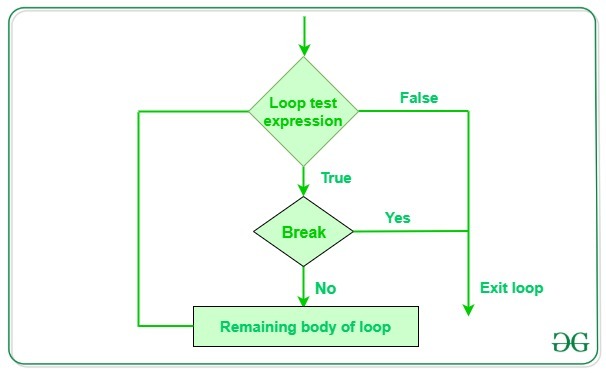
Using loops in Python automates and repeats the tasks in an efficient manner. But sometimes, there may arise a condition where you want to exit the loop completely, skip an iteration or ignore that condition. These can be done by **loop control statements**. Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed. Python supports the following control statements.

* Continue statement
* Break statement
* Pass statement

In this article, the main focus will be on break statement.

#### **Break statement**

Break statement in Python is used to bring the control out of the loop when some external condition is triggered. Break statement is put inside the loop body (generally after if condition).



**Syntax:**

break

**Example:**

|  |
| --- |
| # Python program to  # demonstrate break statement    s ='geeksforgeeks'  # Using for loop  forletter ins:        print(letter)      # break the loop as soon it sees 'e'      # or 's'      ifletter =='e'orletter =='s':          break    print("Out of for loop")  print()    i =0    # Using while loop  whileTrue:      print(s[i])        # break the loop as soon it sees 'e'      # or 's'      ifs[i] =='e'ors[i] =='s':          break      i +=1    print("Out of while loop") |

**Output:**

g

e

Out of for loop

g

e

Out of while loop

In the above example, both the loops are iterating the string ‘geeksforgeeks’ and as soon as they encounter the character ‘e’ or ‘s’, the if condition becomes true and the flow of execution is brought out of the loop.