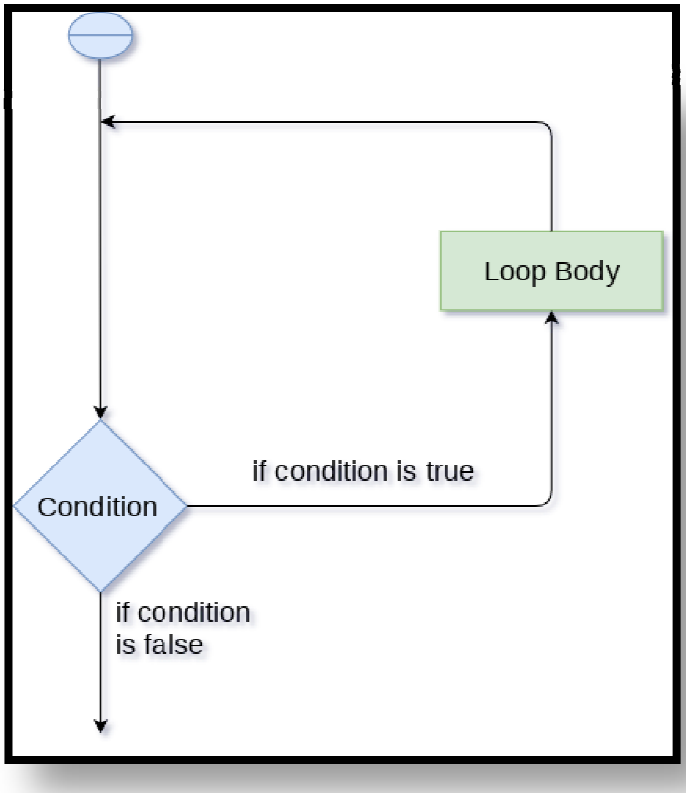
Loops -The flow of the programs written in any programming language is sequential by default. Sometimes we may need to alter the flow of the program. The execution of a specific code may need to be repeated several numbers of times.

Advantages of loops -There are the following advantages of loops in Python -

- It provides code re-usability.
- Using loops, we do not need to write the same code again and again.
- Using loops, we can traverse over the elements of data structures (array or linked lists)

Loop Statement	Description
for loop	• The for loop is used in the case where we need to execute some part of the code until the given condition is satisfied. • The for loop is also called as a per-tested loop. It is better to use for loop if the number of iteration is known in advance.
while loop	• The while loop is to be used in the scenario where we don't know the number of iterations in advance. • The block of statements is executed in the while loop until the condition specified in the while loop is satisfied. It is also called a **pre-tested loop**.
do-while loop	• The do-while loop continues until a given condition satisfies. It is also called post tested loop. • Python does not have any do-while loop



For Loops - The for loop in Python is used to iterate the statements or a part of the program several times. It is frequently used to traverse the data structures like list, tuple, or dictionary.

Syntax -
```
for iterating_var in sequence:
statement(s)
```

For Loop using range() function –

The range() function is used to generate the sequence of the numbers. If we pass the range(10), it will generate the numbers from 0 to 9. The syntax of the range() function is given below.

Syntax - `range(start,stop,step size)`

- The start represents the beginning of the iteration.
- The stop represents that the loop will iterate till stop-1. The **range(1,5)** will generate numbers 1 to 4 iterations. It is optional.
- The step size is used to skip the specific numbers from the iteration. It is optional to use. By default, the step size is 1. It is optional.

Nested for loop in python -Python allows us to nest any number of for loops inside a **for** loop. The inner loop is executed n number of times for every iteration of the outer loop.

Syntax -
```
for iterating_var1 in sequence:          #outer loop
    for iterating_var2 in sequence:      #inner loop
    #block of statements
#other statements
```

While Loop -The Python while loop allows a part of the code to be executed until the given condition returns false. It is also known as a pre-tested loop.

Syntax -
```
while expression:
Statements
```

Infinite Loop -If the condition is given in loop that never becomes false, then the loop will never terminate, and it turns into the infinite loop.

Syntax -

```
while (1):
    print("Hi! we are inside the infinite while loop")
```

Using else with For Loop & While Loop -Python allows us to use the else statement with the for loop and while loop which can be executed only when all the iterations are exhausted.

- For Loop –

```
for i in range(0,5):
    print(i)
else:
print("for loop completely exhausted...")
```

- While Loop -

```
i=1
while(i<=5):
    print(i)
    i=i+1
else:
    print("The while loop exhausted")
```

Break Statement -The break is a keyword in python which is used to bring the program control out of the loop. The break statement breaks the loops one by one, i.e., in the case of nested loops, it breaks the inner loop first and then proceeds to outer loops.

Syntax -

```
#loop statements
break;
```

Example -

```
list =[1,2,3,4]
count = 1;
for i in list:
    if i == 4:
        print("item matched")
        count = count + 1;
        break;
print("found at",count,"location");
```

Example -*break statement with while loop*

```
i = 0;
while 1:
    print(i," ",end=""),
    i=i+1;
    if i == 10:
        break;
print("came out of while loop");
```

Continue -The continue statement in Python is used to bring the program control to the beginning of the loop. The continue statement skips the remaining lines of code inside the loop and start with the next iteration. It is mainly used for a particular condition inside the loop so that we can skip some specific code for a particular condition.The continue statement in Python is used to bring the program control to the beginning of the loop.

Syntax –

```
#loop statements
continue
#the code to be skipped
```

Example –

```
i = 0
while(i < 10):
   i = i+1
   if(i == 5):
      continue
   print(i)
```

Pass Statement – The pass statement is a null operation since nothing happens when it is executed. It is used in the cases where a statement is syntactically needed but we don't want to use any executable statement at its place.Pass is also used where the code will be written somewhere but not yet written in the program file.

```
list = [1,2,3,4,5]
```

```
flag = 0
for i in list:
    print("Current element:",i,end=" ");
    if i==3:
        pass
        print("\nWe are inside pass block\n");
        flag = 1
    if flag==1:
        print("\nCame out of pass\n");
        flag=0
```