Python Iterative Statements

Iteration statements or loop statements allow us to execute a block of statements repeatedly as long as the condition is true.

(Loops statements are used when we need to run same code again and again)

Type of Iteration Statements In Python 3
In Python Iteration (Loops) statements are of three types :-

1. While Loop

2. For Loop

3. Nested Loops

---

1. While Loop In Python

While Loop In Python is used to execute a block of statement till the given condition is true. And when the condition is false, the control will come out of the loop.

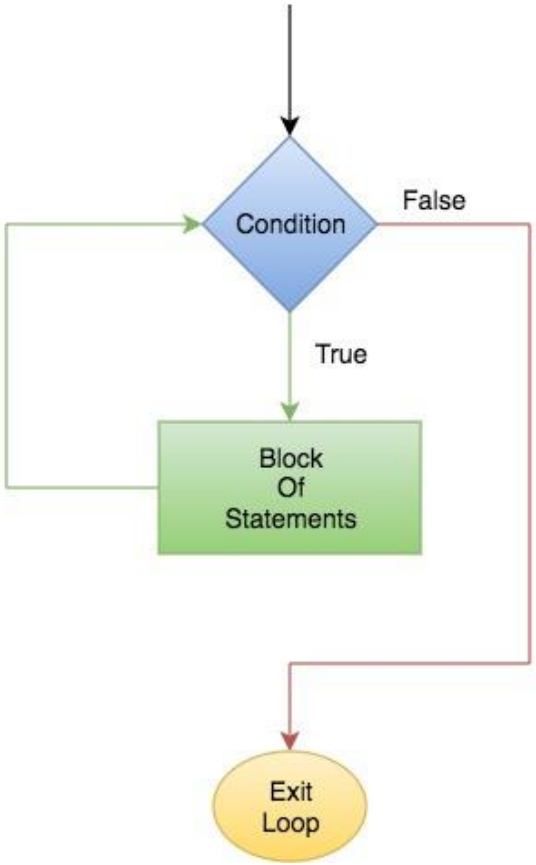
The condition is checked every time at the beginning of the loop.

While Loop Syntax

while (<condition>):

    statements

Flowchart of While Loop



Python Flowchart of While Loop

Examples Of While Loop

x = 0
while (x < 5):
  print(x)
  x = x + 1
 	
Output :-
0
1
2
3
4
	x = 1
	while (x <= 5):
	  print(“Welcome “)
	  x = x + 1
	
	Output :-
	Welcome
	Welcome
	Welcome
	Welcome
	Welcome

While Loop With Else In Python

The else part is executed if the condition in the while loop becomes False.

Syntax of While Loop With Else

```
while (condition):
        loop statements
else:
        else statements
```

Example of While Loop With Else

```
x = 1
while (x < 5):
  print('inside while loop value of x is ',x)
  x = x + 1
else:
  print('inside else value of x is ', x)
```

Output :-

inside while loop value of x is 1
inside while loop value of x is 2
inside while loop value of x is 3
inside while loop value of x is 4
inside else value of x is 5

****Infinite While Loop In Python** A Infinite loop is a loop in which condition always remain True.

Example of Infinite While Loop

```
x = 1
while (x == 1):
  print('hello')
```

Output :-

hello
hello
hello
___
___

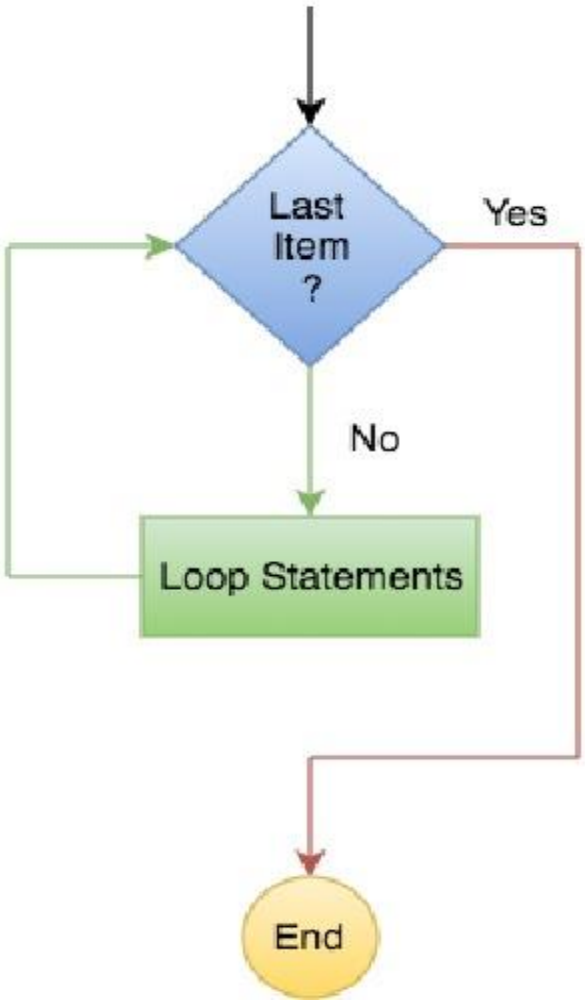
2. For Loop In Python

For loop in Python is used to iterate over items of any sequence, such as a list or a string.

For Loop Syntax

```
for val in sequence:
        statements
```

Flowchart of For Loop



Python Flowchart of For Loop

ExampleS of For Loop

for i in range(1,5):
  print(i)
 	
Output :-
1
2
3
4
	for i in [1,2,3,4] :
	  print("WELCOME ")
	
	Output :-
	WELCOME
	WELCOME
	WELCOME
	WELCOME

****The range() Function In Python

The range() function is a built-in that is used to iterate over a sequence of numbers.

Syntax Of range() Function range(start, stop[, step])

The range() Function Parameters

start: Starting number of the sequence.

stop: Generate numbers up to, but not including this number.

step(Optional): Determines the increment between each numbers in the sequence

Example 1 of range() function
for i in range(5):
  print(i)
Output :-
0
1
2
3
4
	Example 2 of range() function
	for i in range(2,9):
	  print(i)
	Output :-
	2
	3
	4
	5
	6
	7
	8

Example 3 of range() function using step parameter
for i in range(2,9,2):
  print(i)
Output :-
2
4
6
8
	Example 4 of range() function
	for i in range(0,-10,-2):
	  print(i)
	Run Code
	Output :-
	0
	-2
	-4
	-6
	-8

For Loop With Else In Python

The else is an optional block that can be used with for loop.Theelse block with for loop executed only if for loops terminatesnormally.This means that the loop did not encounter any break.

Example 1 of For Loop With Else

```
list=[2,3,4,6,7]
for i in range(0,len(list)):
  if(list[i]==4):
    print('list has 4')
else:
  print('list does not have 4')
```

Run Code

Output :-
List has 4
list does not have 5

Example 2 of For Loop With Else

```
for i in range(0,len(list)):
  if(list[i]==5):
    print('5 is there in the list')
    break
else:
  print('list does not have 5')
```

Run Code

Output :-
list does not have 5

NESTED loops in Python : The placing of one **loop** inside the body of another **loop** is called **nesting**. When you "nest" two **loops**, the outer **loop** takes control of the number of complete repetitions of the inner **loop**

1. Nested while loop

Syntax :

```
initialization
while(condition):
        initialization of inner loop
        while(conition):
                ------
                ------
                Update expression of inner loop
 Update expression of outer loop
```

Examples

```
# To Print Pyramid
'''
1
1 2
1 2 3
1 2 3 4'''
i=1
while(i<=4):
    j=1
    while(j<=i):
        print(j,end=' ')
        j+=1
    print("")
    i+=1

# To Print Pyramid
'''
5
5 4
5 4 3
5 4 3 2
5 4 3 2 1'''
i=5
while(i>=1):
    j=5
    while(j>=i):
        print(j,end=' ')
        j-=1
    print("")
    i-=1
```

1. Nested for loop

Syntax

```
for iterating_var in sequence:
   for iterating_var in sequence:
      statements(s)
   statements(s)
```

Example

```
# To Print Pyramid
'''
1
1 2
1 2 3
1 2 3 4'''

for i in range(1,5):
    for j in range(1,i+1):
        print(j,end=' ')
    print("")


# To Print Pyramid
'''
5
5 4
5 4 3
5 4 3 2
5 4 3 2 1'''

for i in range(5,0,-1):
    for j in range(5,i-1,-1):
        print(j,end=' ')
    print("")
```