Looping Statements

Looping statement is a type of statement in programming as looping statement avoid the duplication of code because it provide repetation of execution code until condition is met. There are mainly two keywords used in the looping statement are **for** keyword and **while** keyword. **for** keyword is used in for-each loop and for loop statement and **while** keyword is used in while loop statement.

While Loop Statement

While loop statement number of iteration is not known. While loop excute until the condition return true as a result, when condition false as a result then loop will not execute and loop stops. Here, we have to use iterative variable which we use in while loop to iterate over each element once that iterative element is commonly know as **ith** variable in while loop. Also remember to add increament otherwise while continue repetation of code without any stopage because everytime condition is true and this is called **infinite loop**.

To write while loop first of all, we have make an iterative variable **i** then in next line we have to write **while** keyword and then we write the condition which we want to check and then use colon after that in next line write the program and after program write increament statement **i=i+1** like this atlast.

In [20]:
```python
#while loop
i = 0
while (i < 11):
    print(i)
    i = i + 1
```

```
0
1
2
3
4
5
6
7
8
9
10
```

While Loop Using Break Keyword

We use **break** keyword in looping statement to stop the repetition of loop program after meeting certain condition and hit **break** keyword. When the program hit **break** keyword, the command or control goes outside the loop and loop stop working. Let us see the example code given below:

```
In [22]: #while using break
i = 0
while(i < 11):
    if(i==8):
        break;
    print(i)
    i = i + 1
```

```
0
1
2
3
4
5
6
7
```

While Loop Using Continue Keyword

We use **continue** keyword in looping statement to stop the repetation for particular condition it hits only not for all after hitting the condition like **break** keyword. **continue** keyword just skip the iterative program for particular condition it hits only in the looping program.

```
In [ ]: #while using continue
i = 0
while(i < 11):
    if(i == 8):
        continue
    print(i)
    i = i + 1
```

```
0
1
2
3
4
5
6
7
```

While Loop Using Else Statement

We use **else** statement in while loop to execute the code when condition does not meet our requirement. Suppose we execute a while loop when its condition returns true then only it executes the program but when condition returns false then loop stops instead of stoping loop we just want to send message that the condition is failed so the loop end. For this we use **else** statement.

In [1]:
```
#while using else
i = 0
while(i < 11):
    print(i)
    i = i + 1
else:
    print("Loop End")
```

```
0
1
2
3
4
5
6
7
8
9
10
Loop End
```

For Loop Statement

For loop statement uses **for** keyword. Here, in for loop the number of iteration of loop is known. Here, we do not need iterative variable. This loop iterate over each element once. We use either membership operator **(in)** or **range()** function in **for loop**. When we use membership operator **(in)** then this is called for-each loop but when we use **range()** function with membership operator (**in**) then it is called for loop. First of all, we write **for** keyword then we write name of the iterative variable after that we use membership operator **(in)** and after that we use **range()** function then in first argument we pass the starting value of iterative variable and in second argument we pass the end value of the iterative variable and atlast we pass the number of steps it jump means step of the loop.

In [1]:
```
#for loop
for i in range(0,11,1):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
10
```

You can also use single argument in the **range()** fnction and the single argument value in range function is the end value of the loop. One more thing in **range()** function the end value is excluded but before that is included in the loop.

By using **break** keyword in the for loop, **break** keyword is mainly used to remove control out of the for loop when it hits the condition where break statement present. It is mainly used to stop the loop.

```
In [3]: #for loop  using break
        for i in range(0,11,1):
            if(i == 4):
                break
            print(i)
```

```
0
1
2
3
```

By using **continue** keyword in the for loop, **continue** keyword only skip the loop when condition met where continue statement present. Only for that contion the loop is skipped but after that loop will continue running till for loop condition.

```
In [7]: #for loop using continue
        for i in range(0,7,1):
            if(i == 4):
                continue
            print(i)
```

```
0
1
2
3
5
6
```

Else statement is mainly used in for loop to excute program or print message when condition does not meet our requirements or it return false. This will print the message that the loop is now going to end.

```
In [9]: #for loop using else
        for i in range(0,5,1):
            print(i)
        else:
            print("End")
```

```
0
1
2
3
4
End
```

Here, we use **for-each loop** as a looping statement because just like list and tuple not all the sequential datatypes are in ordered form or having index so it is too difficult to print their values as it is unordered and unindexed. For that for-each loop is used that visit each value once and throw them to **i** variable then that **i** variable we print. This is how for-each loop works.

In [17]:
```
setAmrit = {1,"Amrit",3,True,5,6,"Keshari",9}
for i in setAmrit:
    print(i)
```

```
1
3
Amrit
5
6
Keshari
9
```

pass is also a keyword that is mainly used in conditional statement, loop statement and during defination of functions. It is used when we do not write program for particular statement just like we write if statement and write some program but for else statement we do not write anything so instead of that we use **pass** keyword.

In [22]:
```
for i in setAmrit:
    if(i==5):
        print("FIVE")
    else:
        pass
```

```
FIVE
```