03

August 26, 2019

#Control Flow Statements
##If
if some_condition:

```
algorithm
```

```
In [1]: x = 12
        if x >10:
            print("Hello")
```

```
Hello
```

##If-else
if some_condition:

```
algorithm
```

else:

```
algorithm
```

```
In [2]: x = 12
        if x > 10:
            print("hello")
        else:
            print("world")
```

```
hello
```

##if-elif
if some_condition:

```
algorithm
```

elif some_condition:

```
algorithm
```

else:

```
algorithm
```

```
In [3]: x = 10
        y = 12
        if x > y:
            print("x>y")
        elif x < y:
            print("x<y")
        else:
            print("x=y")
```

```
x<y
```

if statement inside a if statement or if-elif or if-else are called as nested if statements.

```
In [4]: x = 10
        y = 12
        if x > y:
            print("x>y")
        elif x < y:
            print("x<y")
            if x==10:
                print("x=10")
            else:
                print("invalid")
        else:
            print("x=y")
```

```
x<y
x=10
```

##Loops
###For
for variable in something:

```
algorithm
```

```
In [5]: for i in range(5):
            print(i)
```

```
0
1
2
3
4
```

In the above example, i iterates over the 0,1,2,3,4. Every time it takes each value and executes the algorithm inside the loop. It is also possible to iterate over a nested list illustrated below.

```
In [6]: list_of_lists = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
        for list1 in list_of_lists:
                print(list1)
```

```
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
```

A use case of a nested for loop in this case would be,

```
In [7]: list_of_lists = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
        for list1 in list_of_lists:
            for x in list1:
                print(x)
```

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

###While
while some_condition:

```
algorithm
```

```
In [8]: i = 1
        while i < 3:
            print(i ** 2)
            i = i+1
        print('Bye')
```

```
1
4
Bye
```

##Break
As the name says. It is used to break out of a loop when a condition becomes true when executing the loop.

```
In [9]: for i in range(100):
            print(i)
            if i>=7:
                break
```

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

##Continue

This continues the rest of the loop. Sometimes when a condition is satisfied there are chances of the loop getting terminated. This can be avoided using continue statement.

```
In [10]: for i in range(10):
             if i>4:
                 print("The end.")
                 continue
             elif i<7:
                 print(i)
```

```
0
1
2
3
4
The end.
The end.
The end.
The end.
The end.
```

##List Comprehensions

Python makes it simple to generate a required list with a single line of code using list comprehensions. For example If i need to generate multiples of say 27 I write the code using for loop as,

```
In [11]: res = []
         for i in range(1,11):
             x = 27*i
             res.append(x)
         print(res)
```

```
[27, 54, 81, 108, 135, 162, 189, 216, 243, 270]
```

Since you are generating another list altogether and that is what required, List comprehensions is a more efficient way to solve this problem.

```
In [12]: [27*x for x in range(1,11)]

Out[12]: [27, 54, 81, 108, 135, 162, 189, 216, 243, 270]
```

That's it!. Only remember to enclose it in square brackets

Understanding the code, The first bit of the code is always the algorithm and then leave a space and then write the necessary loop. But you might be wondering can nested loops be extended to list comprehensions? Yes you can.

```
In [13]: [27*x for x in range(1,20) if x<=10]

Out[13]: [27, 54, 81, 108, 135, 162, 189, 216, 243, 270]
```

Let me add one more loop to make you understand better,

```
In [15]: [2*z for i in range(50) if i==27 for z in range(1,11)]

Out[15]: [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

Pyramid program

```
In [ ]: n=5
        for i in range(0, n):
                # inner loop to handle number of columns
                # values changing acc. to outer loop
                for j in range(0, i+1):

                    # printing stars
                    print("* ",end="")

                # ending line after each row
                print("\r")
```

Another way for the above pattern

```
In [ ]: n = 5
        myList = []
        for i in range(1,n+1):
            myList.append("* "*i)
        print("\n".join(myList))
```

```
In [ ]: # number of spaces
        k = 2*n - 2
        # outer loop to handle number of rows
        for i in range(0, n):
                # inner loop to handle number spaces
                # values changing acc. to requirement
                for j in range(0, k):
                    print(end=" ")
                # decrementing k after each loop
                k = k - 2
                # inner loop to handle number of columns
                # values changing acc. to outer loop
                for j in range(0, i+1):
                    # printing stars
                    print("* ", end="")
                # ending line after each row
                print("\r")
```

Pascal Triangle

```
In [16]: n=int(input("Enter number of rows: "))
         a=[]
         for i in range(n):
             a.append([])
             a[i].append(1)
             for j in range(1,i):
                 a[i].append(a[i-1][j-1]+a[i-1][j])
             if(n!=0):
                 a[i].append(1)
         for i in range(n):
             print("   "*(n-i),end=" ",sep=" ")
             for j in range(0,i+1):
                 print('{0:6}'.format(a[i][j]),end=" ",sep=" ")
             print()
```

```
Enter number of rows: 3
               1
            1      1
         1      2      1
```

Reverse the number

```
In [17]: Rev=0
         N = int(input("Enter the Value of N : "))
         K=N
         while (K > 0) :
             Rem = K % 10
             K = K // 10
```

```
            Rev = Rev * 10 + Rem
        print ("The Reverse of a ",N," is : ", Rev)
```

```
Enter the Value of N : 123
The Reverse of a  123  is :  321
```

Reverse the string

```
In [18]: RStr=""
         Str=input("Enter the String : ")
         i = len(Str)
         while(i != 0):
             RStr = RStr + Str[i-1]
             i=i-1
         print("The Reversed String is ",RStr)
```

```
Enter the String : Sai
The Reversed String is  iaS
```

Fibonacci Series

```
In [22]: A=0
         B=1
         C=B
         N=int(input("Enter the Value of N : "))
         print(A,end=' ')
         print(B,end=' ')
         while (C <= N-2):
             C=A+B
             A=B
             B=C
             print(C,end=' ')
```

```
Enter the Value of N : 7
0 1 1 2 3 5 8
```

Factorial of a Number

```
In [23]: Fact=1
         N=int(input("Enter the Value of N : "))
         for i in range( 1, N + 1 ) :
             Fact = Fact * i
         print("The Factorial of ",N," is : ", Fact)
```

```
Enter the Value of N : 4
The Factorial of  4  is :  24
```

Even or Odd check

```
In [24]: num = int(input("Enter a number: "))
         if (num % 2) == 0:
             print(" {0} is Even".format(num))
         else:
             print(" {0} is Odd".format(num))
```

```
Enter a number: 12
 12 is Even
```

CGPA Calculation

```
In [27]: CGPA = float(input("Enter the student CGPA : "))
         if((CGPA >= 9) and (CGPA <= 10)) :
             print("Outstanding")
         elif((CGPA >= 8) and (CGPA < 9)) :
             print("Excellent")
         elif((CGPA >= 7) and (CGPA < 8)) :
             print("Good")
         elif((CGPA >= 6) and (CGPA < 7)) :
             print("Average")
         elif((CGPA >= 5) and (CGPA < 6)) :
             print("Need to Improve")
         elif((CGPA >=0) and (CGPA < 5)):
             print("Very Poor")
         else:
             print("Invalid")
```

```
Enter the student CGPA : 9.8
Outstanding
```

```
In [31]: import cmath
         print("Program to find Roots of Quadratic Equation")
         A=float(input("Enter the Co-efficient of A : "))
         B=float(input("Enter the Co-efficient of B : "))
         C=float(input("Enter the Co-efficient of C : "))
         D = ((B**2)-(4*A*C))
         Root1 = (-B + cmath.sqrt(D))/(2 * A)
         Root2 = (-B - cmath.sqrt(D))/(2 * A)
         if(D==0):
             print("roots are real and eqal,",end='')
             print(' The roots are {0} and {1}'.format(Root1.real,Root2.real))
         elif(D>0):
             print("roots are real and uneqal,",end='')
             print(' The roots are {0} and {1}'.format(Root1.real,Root2.real))
         else:
```

```
            print("roots are imaginary,",end='')
            print(' The roots are {0} and {1}'.format(Root1,Root2))
```

```
Program to find Roots of Quadratic Equation
Enter the Co-efficient of A : 2
Enter the Co-efficient of B : 4
Enter the Co-efficient of C : 9
roots are imaginary, The roots are (-1+1.8708286933869707j) and (-1-1.8708286933869707j)
```

Sum of 1st natural numbers

```
In [32]: sum = 0
         n = int(input("Enter n value "))
         for i in range(1 , n+1):
             sum += i
         print("summation of first", n,end='')
         print(" natural number is ",sum)
```

```
Enter n value 5
summation of first 5 natural number is  15
```

Write a program that read a group 'g' of five numbers and another number 'n' and print a number in 'g' if it is a factor for a given number n?

```
In [33]: my_list = []
         for i in range(1,6):
             my_list.append(int(input("Enter the value for element-{0}: ".format(i))))
         #print(my_list[:])
         n = int(input("enter the number to check its factors: "))
         #print(n)
         for element in my_list:
                if(n%element == 0):
                     print(element)
```

```
Enter the value for element-1: 4
Enter the value for element-2: 7
Enter the value for element-3: 6
Enter the value for element-4: 9
Enter the value for element-5: 8
[4, 7, 6, 9, 8]
enter the number to check its factors: 36
4
6
9
```

Write a menu driven program which get user choice to perform add/sub/mul/div with the obtained two input?

```
In [34]: num1 = int(input("enter the 1st num: "))
         num2 = int(input("enter the 2nd num: "))
         choice = input("enter your choice:add/sub/mul/div: ")
         if(choice=='add'):
             result = (num1+num2)
             print(result)
         elif(choice=='sub'):
             result = (num1-num2)
             print(result)
         elif(choice == 'mul'):
             result = (num1*num2)
             print(result)
         elif(choice == 'div'):
             if(num2!=0):
                 result = (num1/num2)
                 print(result)
             else:
                 print("Divide by zero not possible")
         else:
             print("invaild choice")
```

```
enter the 1st num: 7
enter the 2nd num: 0
enter your choice:add/sub/mul/div: add
7
```

Write a program to display few odd multiples of a odd number n?

```
In [35]: num = int(input("enter an odd number: "))
         if(num%2!=0):
             for i in range(2,num):
                 if(num%i==0 and i%2!=0):
                     if(i<=int(num/i)):
                         print(i,",",int(num/i))
         else:
             print("entered number is not an odd number")
```

```
enter an odd number: 45
3 , 15
5 , 9
```