Control Structures

1. **if** Conditional Statement

```
 def moderate(marks, passMarks):
     if marks==passMarks-1 or marks==passMarks-2:
             marks=passMarks
     return marks

def main():
#this name can be anything but while calling the function it should be as it is
    passMarks=40
    marks=input("Enter marks:")
    marksInInteger=int(marks)
    moderateMarks=moderate(marksInInteger, passMarks)
    print("Your moderated marks is:", moderateMarks)

if __name__=='__main__':

    main()
```

Python functions are not restricted to having a single return statement. If a given function has more than one return statement, then the first one encountered will determine the end of the function's execution and also its return value.

- → **if**
- → **if else**
- → **if elif else**
- → nested **if elif else**

```
if n1>n2:
   if n1>n3:
       maxNum=n1
   else:
       maxNum=n3
elif n2>n3:
    maxNum=n2
else:
    maxNum=n3
```

Nested Function Approach

```
def max3(n1, n2, n3):
    def max2(n1, n2):
        if n1>n2:
          return n1
        else:
          return n2
    return max2(max(n1, n2), n3)
```

2. Iteration (**for** and **while** statements)

   Execution of a sequence of statements in a loop is known as an iteration of the loop.

   i. **for** loop

      the control statement **for** is used when we want to execute a sequence of statements a fixed number of times.

```
n=int(input("Enter number of terms- "))
total=0
for i in range(1,n+1):
    total+=i
print("Total of all positive numbers from 1 to ",n,"   is-->", total)
```

*** range(start, end, increment) [→or decrement]**
*** range function generates a sequence of integers**
*** but the end value is not included**
*** if the increment/ decrement value i.e., isn't included then it's assumed to be 1**
*** if the 1st value i.e., isn't included then it's assumed to be 0**
*** values of start and end should be of integer type**
**** since Python output is always aligned to the left so, to make it aligned to the right we need to use a format string '%5d' to indicate that atleast 5 spaces are to be reserved before the value**

```
num=2

for multiple in range(1, 11):
    product=num*multiple
    print(num, '*', '%2d'%multiple, '=', '%5d'%product)
```

vii. **while** loop

the **while** loop is used for executing a sequence of statements again and again on the basis of some test condition.
as long as the test condition is **True**, the body of the loop is executed otherwise the control immediately moves to the statements after **while** loop

```
num=input(‘Enter a number-> ’)
total=0
while num != ‘’:
    total+=int(num)
    num=input(‘Enter a number-> ’)
print(“Sum of all input numbers is ”, total)
```

➔ Infinite loops

```
import time
while True:
    try:
        print("Loop processing...")
        print("Press ctrl+c to break")
        time.sleep(1)
    except KeyboardInterrupt:
        print('User interrupted the loop...exiting...')
        break
```

break, **continue** and **pass** Statements

To alter the normal flow of the loop in response to the occurrence of an event.

- **i.** The **break** statement enables us to exit the loop and transfer the control to the statement following the body of the loop.
- **ii.** The **continue** statement is used to transfer the iteration to the next iteration of the loop. When the **continue** statement is executed, the code that occurs in the body of the loop next to the **continue** statement is skipped.
- **iii.** The **pass** statement lets the program go through this piece of code without executing any code. Often **pass** is used as reminder for some code to be filled in later.