Chapter 6
Flow of Control (1 mcq+2*1+5*1=8 Marks)

INTRODUCTION

- Selection
- Indentation
- Repetition
- Break and Continue Statements
- Nested Loops

Flow of Control

The order of execution of the statements in a program is known as flow of control. The events can flow in a sequence, or on branch based on a decision or even repeat some part for a finite number of times. The flow of control can be implemented using control structures. Python supports two types of control structures—selection and repetition.

There are three basic programming constructs.

- Sequence
- Selection
- Repetition

Sequence: The statements are executed one after another, i.e., in a sequence.

#Program to print the difference of two input numbers

```
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))
diff = num1 - num2
print("The difference of",num1,"and",num2,"is",diff)
```

SELECTION: Selection of the statements is done for execution based on the condition.

- It is also known as conditional construct.
- This structure helps the programmer to take appropriate decision.
- There are four kinds of selection constructs.

a. Simple – if
b. if –else
c. if – elif
d. Nested –if

a. <u>Simple - if:</u>

- This structure helps to decide the execution of a particular statement based on condition.
- This statement is also called as **one-way branch**.

Syntax:

if condition:
    Statement 1

#Example for simple if

```
age = int(input("Enter your age:  "))
```

```
 if age >= 18:
    print("Eligible to vote")
```

#to calculate the discount using simple if:

```
amt=int(input("Enter the amount: "))
disc=0
if amt>1000:
        disc=0.1
disamt=amt*disc
netamt=amt-disc
print("Amount=",amt)
print("Discount Amount=",disamt)
print("Net Amount=",netamt)
```

b. **if – else statement:**

This structure helps to decide whether a set of statements should be executed or another set of statements should be executed. This statement is also called as two-way branch.

Syntax:

```
 if testcondition:
        Statement 1
 else:
        Statement 2
```

#if-else example

```
age = int(input("Enter your age "))
if age >= 18:
        print("Eligible to vote")
 else:
        print("Not eligible to vote")
```

#calculate discount

```
amt=int(input("Enter the amount: "))
disc=0
if amt>0:
        disc=0.1
else:
      disc=0
disamt=amt*disc
netamt=amt-disc
print("Amount=",amt)
print("Discount Amount=",disamt)
print("Net Amount=",netamt)
```

#Subtract the smaller number from the bigger number so that we always get a positive difference.

```
num1 = int(input("Enter first number: "))
```

```
num2 = int(input("Enter second number: "))
if num1 > num2:
        diff = num1 - num2
else:
        diff = num2 - num1
print("The difference of", num1, " and ", num2, " is", diff)
```

#print the student is pass or fail

```
marks=int(input("Enter marks of a student: "))
if marks>=35:
        print("Passed")
else:
        print("Failed")
```

#Design an algorithm to find the number is positive or negative

```
n=int(input("enter the number:" ))
if n>0:
        print(n," is positive number")
else:
        print(n, "is negative number")
```

#Design an algorithm to find the larger of two numbers

```
m=int(input("enter the first number:" ))
n=int(input("enter the second number:" ))
if m>n:
        print(m," is larger")
else:
        print(n, "is larger")
```

#Design an algorithm to find the number is odd or even

```
n=int(input("enter the number:" ))
if n%2==0:
        print(n," is even number")
else:
        print(n, "is odd number")
```

c. if - elif statement:

This structure helps the programmer to decide the execution of a statement from multiple statements based on a condition. There will be more than one condition to test. This statement is also called as multiple-way branch structure.

Syntax:

```
if condition 1:
        statement1
elif condition 2:
        statement2
elif condition 3:
```

```
        statement3
        …….. ……
elif condition n:
        statement n
else:
        default statement
```

#Check whether a number is positive, negative, or zero.

```
number = int(input("Enter a number: "))
if number > 0:
    print("Number is positive")
elif number < 0:
    print("Number is negative")
else:
    print("Number is zero")
```

#input the marks and print the result

```
marks=int(input("Enter marks="))
if marks>=85:
   print("Distinction")
elif marks>=60:
   print("First Class")
elif marks>=50:
   print("Second Class")
elif marks>=35:
   print("Pass Class")
else:
   print("Failed")
```

#Find the greatest of three numbers (A, B, C):

```
a, b, c = map(int, input("Enter three numbers: "). split()) #input 3 integers in a single line
if a>b and a>c:
   large=a
elif b>c:
   large=b
else:
   large=c
print("Large=",large)
```

 # categorise a person as either child (<13), teenager (>=13 but <20) or adult (>=20), based on age specified

```
age=int(input("Enter your age: "))
if age<13:
   print("Child")
elif age>=13 and age<20:
   print("Teenager")
```

```
else:
   print("Adults")
```

 # Program to classify numbers as “Single Digit”, “Double Digit” or “Big”.

```
num=int(input("Enter Number="))
if num<=9:
   print("Single digit number")
elif num<=99:
   print("Double digit number")
else:
   print("Big")
```

#Display the appropriate message as per the color of signal at the road crossing.

```
signal = input("Enter the color: ")
if signal == "red" or signal == "RED":
        print("STOP")
elif signal == "orange" or signal == "ORANGE":
        print("Be Slow")
elif signal == "green" or signal == "GREEN":
        print("Go!")
```

#Program to create a four function calculator

```
result = 0
val1 = float(input("Enter value 1: "))
val2 = float(input("Enter value 2: "))
op = input("Enter any one of the operator (+,-,*,/): ")
if op == "+":
   result = val1 + val2
elif op == "-":
   if val1 > val2:
      result = val1 - val2
   else:
      result = val2 - val1
elif op == "*":
      result = val1 * val2
elif op == "/":
   if val2 == 0:
      print("Error! Division by zero is not allowed. Program terminated")
   else:
      result = val1/val2
else:
   print("Wrong input, program terminated")
print("The result is ",result)
```

d. Nested if statement:

The if statement within another if statement is called Nested – if statement.

Syntax:

```
if condition1:
    if condition2:
        statement1
    else:
        statement2
else:
    if condition3:
        statement3
    else:
        statement4
```

#example for nested if

```
n=int(input("Enter the Number="))
if n>=0:
        if n%2==0:
                print ("even number")
        else:
                print ("odd number")
else:
        print("negative number")
```

Indentation:

- Python uses indentation for block as well as for nested block structures.
- Leading whitespace (spaces and tabs) at the beginning of a statement is called indentation.
- In Python, the same level of indentation associates statements into a single block of code.
- The interpreter checks indentation levels very strictly and throws up syntax errors if indentation is not correct.

#Program to find larger of the two numbers

```
 num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))
 if num1 > num2:
         print("first number is larger")
        print("Bye")
 else:
         print("second number is larger")
        print("Bye Bye")
```

Repetition / Iteration / Looping:

Looping means execution of some program statements repeatedly till some specified condition is satisfied.

Example:

- Payment of electricity bill, which is done every month.

- The life cycle of butterfly.

There are 2 looping constructs in Python. **for** and **while**.

The 'For' Loop:

- The for statement is used to iterate over a range of values or a sequence.
- The for loop is executed for each of the items in the range.
- These values can be numeric, string, list, or tuple.
- With every iteration of the loop, the control variable checks whether each of the values in the range have been traversed or not.
- When all the items in the range are exhausted, the statements within loop are not executed; the control is then transferred to the statement immediately following the for loop.
- While using for loop, it is known in advance the number of times the loop will execute.

Syntax:

```
for <control-variable> in <sequence/items in range>:
        <statements inside body of the loop>
```

#Print the characters in word PYTHON using for loop

```
for letter in 'PYTHON':
        print(letter)
```

#Print the given sequence of numbers using for loop

```
count = [10,20,30,40,50]
for num in count:
   print(num)
```

#Print even numbers in the given sequence

```
numbers = [1,2,3,4,5,6,7,8,9,10]
for num in numbers:
   if (num % 2) == 0:
      print(num, 'is an even Number')
```

The range() Function: The range() is a built-in function in Python.

Syntax: range ([start], stop [, step])

- It is used to create a list containing a sequence of integers from the given start value upto stop value (excluding stop value), with a difference of the given step value.
- In function range(), start, stop and step are parameters. The start and step parameters are optional.
- If start value is not specified, by default the list starts from 0.
- If step is also not specified, by default the value increases by 1 in each iteration.
- All parameters of range ( ) function must be integers.
- The step parameter can be a positive or a negative integer excluding zero.
- The function range( ) is often used in for loops for generating a sequence of numbers

#start and step not specified

```
list(range(10))
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

#default step value is 1
list(range(2, 10))
[2, 3, 4, 5, 6, 7, 8, 9]

#step value is 5
list(range(0, 30, 5))
[0, 5, 10, 15, 20, 25]

#step value is -1. Hence, decreasing sequence is generated
list(range(0, -9, -1))
[0, -1, -2, -3, -4, -5, -6, -7, -8]

#print natural numbers from 1 to 10
for i in range(11):
   print (i)

#print natural numbers from 1 to 10
for i in range(1,11,1):
   print (i)

#print natural numbers from 11 to 20
for i in range(11,21,1):
   print (i)

#print natural numbers from -1 to -10
for i in range(-1,-11,-1):
   print (i)

#print even series from 1 to 20
for i in range(2,21,2):
   print (i)

#print odd series from 1 to 20
for i in range(1,20,2):
   print (i)

#print the series 100, 95, 90,…5
for i in range(100,0,-5):
   print (i)

#Program to print the multiples of 10 for numbers in a given range.
m=int(input("Minimum Range="))
n=int(input("Maximum Range="))
for i in range(m,n,10):
   print(i)
```

The 'While' Loop:

- The while statement executes a block of code repeatedly as long as the control condition of the loop is true.
- The control condition of the while loop is executed before any statement inside the loop is executed.
- After each iteration, the control condition is tested again and the loop continues as long as the condition remains true.
- When this condition becomes false, the statements in the body of loop are not executed and the control is transferred to the statement immediately following the body of while loop.
- If the condition of the while loop is initially false, the body is not executed even once.
- The statements within the body of the while loop must ensure that the condition eventually becomes false; otherwise the loop will become an infinite loop, leading to a logical error in the program.

Syntax:

```
while test_condition:
        body of while
```

#Print first 5 natural numbers using while loop

```
count = 1
while count <= 5:
    print(count,end= ' ')
    count += 1
```

#Find the factors of a number using while loop

```
num = int(input("Enter a number to find its factor: "))
i = 1
while i <= num/2 :
    if num % i == 0:
        print(i, end=' ')
    i += 1
```

Break and continue Statement:

- Looping constructs allow programmers to repeat tasks efficiently.
- In certain situations, when some particular condition occurs, we may want to exit from a loop (come out of the loop forever) or skip some statements of the loop before continuing further in the loop.
- These requirements can be achieved by using break and continue statements, respectively.
- Python provides these statements as a tool to give more flexibility to the programmer to control the f low of execution of a program.

Break Statement:

The break statement alters the normal flow of execution as it terminates the current loop and resumes execution of the statement following that loop.

#Program to demonstrate the use of break statement in loop

```
for num in range(10):
    num = num + 1
    if num == 8:
        break
```

```
    print(num,end=' ')
```

#Find the sum of all the positive numbers entered by the user till the user enters a negative number.

```
sum = 0
print("Enter numbers to find their sum, negative number ends the loop:")
while True:
    num = int(input("Enter the number: "))
    if (num < 0):
        break
    sum += num
print("Sum =", sum)
```

#Write a Python program to check if a given number is prime or not.

```
n=int(input("Enter the number="))
f=1
for i in range(2,n):
   if n%i==0:
      f=0
      break;
else:
   print(n,"is prime")
if f==0:
   print(n,"is not prime")
```

or

```
n=int(input("Enter the number="))
for i in range(2,n):
   if n%i==0:
     print(n, " is not prime")
      break;
else:
    print(n,"is prime")
```

Continue Statement

- When a continue statement is encountered, the control skips the execution of remaining statements inside the body of the loop for the current iteration and jumps to the beginning of the loop for the next iteration.
- If the loop's condition is still true, the loop is entered again, else the control is transferred to the statement immediately following the loop.

#Prints values from 0 to 6 except 3

```
for num in range(6):
   num = num + 1
   if num == 3:
      continue
   print(num, end= ' ')
```

Nested Loops: A loop may contain another loop inside it. A loop inside another loop is called a nested loop.

#Program to demonstrate working of nested for loops

```
for i in range(4):
   for j in range(i):
      print(i)
   print()
```

Output:

```
1

2
2

3
3
3
```

#Program to print the pattern for a number input by the user.

```
n = int(input("Enter the number of rows to generate a pattern = "))
for i in range(1,n + 1):
    for j in range(1, i + 1):
        print(j, end = " ")
    print()
```

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

#Program to print the pattern for a number input by the user.

```
r=int(input("Rows="))
for i in range(1,r+1):
   for j in range(1,i+1):
      print(i,end=" ")
   print()
```

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

#Program to find prime numbers between 2 to 50 using nested for loops.

```
for n in range(2,51):
   for j in range(2,n):
      if n%j == 0:
         break
   else:
      print(n, end=" ")
```

5. Find the output of the following program segments:

a = 110 while a > 100: print(a) a -= 2 **110 108 106 104 102**	for i in range(20,30,2): print(i) **20 22 24 26 28**

country = 'INDIA'
for i in country:
print (i)
I
N
D
I
A
	i = 0; sum = 0
	while i < 9:
	if i % 4 == 0:
	sum = sum + i
	i = i + 2
	print (sum)
	12

for x in range(1,4): for y in range(2,5):	var = 7 while var > 0:

```
        if x * y > 10:
            break
        print (x * y)
```

2
3
4
4
6
8
6
9

```
print ('Current variable value: ', var)
    var = var -1
    if var == 3:
        break
    else:
        if var == 6:
            var = var -1
            continue
    print ("Good bye!")
```

Current variable value: 7
Current variable value: 5
Good bye!
Current variable value: 4

#Write a function to print the table of a given number. The number has to be entered by the user.

```
num=int(input("Enter the number: "))
for i in range(1,11):
   print(num,"*",i,"=",num*i)
```

#Write a program that prints minimum and maximum of five numbers entered by the user.

```
smallest = 0
largest = 0
for a in range(0,5):
   x = int(input("Enter the number: "))
   if a == 0:
      smallest = largest = x
   if(x < smallest):
      smallest = x
   if(x > largest):
      largest = x
print("The smallest number is", smallest)
print("The largest number is ",largest)
```

#Write a program to check if the year entered by the user is a leap year or not.

1. The year is multiple of 400.
2. The year is a multiple of 4 and not a multiple of 100.

1900, 2100, 2100, 2200, 2300 are not not leap years.

```
year=int(input("Year="))
if year%4==0 and year%100!=0 or year%400==0:
   print(year,"is a leap year")
else:
   print(year,"is not a leap year")
```

#Write a program to generate the sequence: –5, 10, –15, 20, –25….. n, where n is an integer input by the user

```
num = int(input("Enter the number: "))
```

```
for a in range(1,num+1):
   if(a%2 == 0):
      print(a * 5, end=",")
   else:
      print(a * -5,end=",")
```

#Write a program to find the sum of 1+ 1/8 + 1/27.....1/n3, where n is the number input by the user.

```
n=int(input("Enter the n Value: "))
sum=0
for i in range(1,n+1):
   sum=sum+1/(i**3)
   print("Sum of the series=",sum)
```

#Write a program to find the sum of digits of an integer number, input by the user.

```
sum = 0
n = int(input("Enter the number: "))
while n > 0:
   r = n % 10
   sum = sum + r
   n = n//10
print("The sum of digits of the number is", sum)
```

#Write a function that checks whether an input number is a palindrome or not.

```
n=int(input("Enter number:"))
temp=n
rev=0
while(n>0):
   r=n%10
   rev=rev*10+r
   n=n//10
if(temp==rev):
   print("The number is a palindrome!")
else:
   print("The number isn't a palindrome!")
```

[Note: A number or a string is called palindrome if it appears same when written in reverse order also. For example, 12321 is a palindrome while 123421 is not a palindrome]

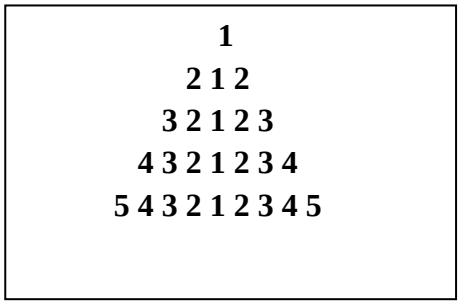
#Write a program to print the following patterns:

```
n=3
for i in range (1,n+1):
   space=(n-i)*" "
   star=(2*i-1)* "*"
   print(space,star)
for j in range(n - 1, 0, -1):
   space = (n - j)*" "
   star = (2 * j - 1)*"*"
```

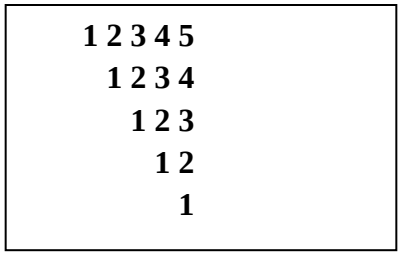
```
   *
  * * *
 * * * * *
  * * *
   *
```

```
print(space, star)
```

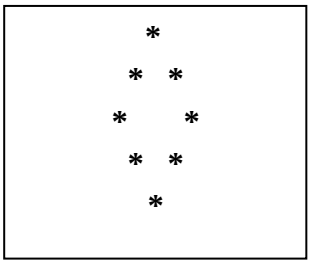
```
n=5
for i in range(1,n+1):
    space=(n-i)*" "
    print(space,end=" ")
    for k in range(i,1,-1):
        print(k,end=" ")
    for j in range(1,i+1):
        print(j,end=" ")
    print()
```



```
n=5
for i in range(n,0,-1):
    space=(n-i)*" "
    print(space,end="")
    for k in range(1,i+1):
        print(k,end="")
    print()
```



```
n=3
k=0
for i in range(1, n+1):
    space=(n-i)* " "
    print(space,end=' ')
    while(k!=(2*i-1)):
        if(k==0 or k==2*i-2):
            print('*',end="")
        else:
            print(' ',end="")
        k=k+1
    k=0
    print()
for j in range(n-1,0,-1):
    space=(n-j)*" "
    print(space,end=" ")
    k=(2*j-1)
    while k>0:
        if k==1 or k==2*j-1:
            print("*",end="")
        else:
            print(" ",end="")
```



```
    k=k-1
  print()
```

Write a program to find the grade of a student when grades are allocated as given in the table below.

Percentage of Marks	Grade
Above 90%	A
80% to 90%	B
70% to 80%	C
60% to 70%	D
Below 60%	E

Percentage of the marks obtained by the student is input to the program.

```
n = float(input('Enter the percentage of the student: '))
if(n > 90):
   print("Grade A")
elif(n > 80):
   print("Grade B")
elif(n > 70):
   print("Grade C")
elif(n >= 60):
   print("Grade D")
else:
   print("Grade E")
```