Program no: 16.

AIM: Program to find factorial of a number.

```
i= int (input ("Enter number :"))
fac=1
while (i>0):
    fac = fac * i
    i = i-1
print ("factorial=", fac)
```

Output:-

Enter number: 5

factorial = 120

Program no: 17

AIM: Generate Fibonacci series of N terms.

```
n= int (input (" Enter number:"))
a=0
b=1
c=0
while (c<=n):
    print (c)
    a=b
    b=c
    c=a+b
```

Output:

Enter number : 8

0

1

1

2

3

5

8

Program 18

AIM: Find sum of all items in a list.

```
mylist = [6, 12, 18, 24]
total = 0
for i in range (0, len (mylist)):
    total = total + mylist [i]
print (" Sum of all elements in given list:", total)
```

Output:

Sum of all elements in given list: 60

Program 19:

AIM: Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.

```
def call ():
    n=0
    for x in range (1000, 10000):
    num = str (x)
    number = int (x)
     first = int (num [0])
     second = int (num [1])
     third = int (num [2])
     fourth = int (num [3])
      if first %2 ==0:
         if second %2 ==0:
            if third %2 ==0:
               if fourth %2 ==0
```

```
for i in range (2, number):
    if i*i = =number:
        print (number)
```

Output:-

4624

6084

6400

8464

Program 20.

AIM:- Display the given pyramid with step number accepted from user.

```
lines = int (input ("Enter a number :"))
i=1
j=1
while i<= lines:
    j=1
    while j<=i:
        temp=i+j
        print (temp, end=' ', flush = True)
        print (" ", end=' ', flush=True)
        j=j+1;
    print ("\n");
    i= i+1;
```

Output:-

Enter a number = 4

1

2 4

3 6 9

4 8 12 16

Program 21:

AIM:- Count the number of characters in a string

```
string = input ("Enter any string:")
c= input ("Enter characters to check frequency:")
count=0
for i in string:
    if i==c:
        count+ =1
print (c," occurs", count, " time(s).")
```

Output:

Enter any string : good morning

Enter character to check frequency : o

o occurs 3 time(s).

Program 22:

AIM: Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'.

```
def add_string (str1):
    length = len (str1)
    if length > 2:
        if str1[-3:] == 'ing':
            str1 += 'ly'
        else:
            str1 += 'ing'
        return str1
print (add_string ('jump'))
print (add_string ('jumping'))
```

Output:-

jamping

jumping

jumpingly.

Program 23:

AIM: Accept a list of words and return length of longest word.

```
string = input ("Enter any string:")
count = 0
word = 0
for i in string:
  if (i==' '):
    count=0
  else
    count = count+1
  if (word < count):
    word = count
print ("The length of longest word", word)
```

Output:-

Enter any string: have a nice day!

The length of longest word = 4

Program: 24

AIM: Construct the following pattern using nested loop

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

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

Output:-

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

Program 25

AIM: Generate all factors of a number.

```
x= int (input (" Enter the number:"))
for i in range (1, x+1):
    if x%i==0:
        print (i)
```

Output:-

Enter the number: 12

1
2
3
4
6
12

Program 26:

AIM: Write lambda functions to find area of square, rectangle and triangle.

```
s= int (input ("Enter the sides:"))
Enter the sides: 4
x= lamba a:a*a
print (x(s))
16
l= int (input ("Enter length:"))
Enter length: 3
b= int (input ("Enter breadth:"))
Enter breadth: 4
y= lambda l, b:l*b
print (y(l,b))
12
b= int (input ("Enter height of a triangle:"))
Enter height of a triangle: 6
b1 = int (input ("Enter base of triangle :"))
```

Enter ~~height~~ base of triangle: 6

```
z= lambda h, b1: (1*b1)/2
print (z(h, b1))
```

12.0

Program 27

AIM: Square of N

```
def sq(n):
S = [i*i for i in range (1,n+1)]
return s
print (sq(9))
```

Output:-

1, 4, 9, 16, 25, 36, 49, 64, 81

Program 28

AIM: Form a list of vowels selected from a given word.

```
str1 = input ("Enter a string")
str1_lower = str1.lower()
vowels = "aeiou"
count = 0
for i in str1_lower:
    if i in vowels:
    count = count + 1
print (i)
print ("count of vowels in the given string:"
                                    count)
```

Output:-

Enter a string: language

a

Count of vowels in the given string: 1

u

Count of vowels in the given string: 2

a

Count of vowels in the given string: 3

e

Count of vowels in the given string: 4

Program 29:

AIM: Store a list of names. Count the occurence of 'a' within the list

```
test_str = " Simplicity is the ultimate sophistication"
count = 0
for i in test_str:
    if i = 'a':
        count = count + 1
print (" count of a in simplicity is the
    ultimate sophistication is :" + str (count))
```

Output:-

Count of a in simplicity is the ultimate sophistication is : 2

Program 30:

AIM: Sort dictionary in ascending and descending order.

```
import operator
d= {1: 2, 3:4 , 4:3, 2:1 , 0:0}
print ('dictionary :', d)
s= sorted (d. items(), key= operator. itemgetter(1))
print ('ascending order : ',s)
s1 = dict (sorted (d.items(), key= operator .itemgetter(1),
        reverse =True))
print ('descending order:', s1)
```

Output:-

dictionary : (1: 2, 3: 4, 4: 3 , 2: 1, 0: 0)

ascending order: [(0,0) , (2,1) (1,2), (4,3), (3, 4)]

descending order: {3: 4, 4: 3, 1: 2, 2: 1, 0: 0}

Program 31:

AIM: Merge two dictionaries

```
def Merge (dict1, dict2):
    return (dict2.update (dict1))
dict1 = {'a':4, 'b':8}
dict2 = {'c':2, 'd':6}
print (Merge (dict1, dict2))
None
print (dict2)
{'c':2, 'd':6, 'a':4, 'b':8}
```

Output:

None

{'c': 2, 'd': 6, 'a': 4, 'b': 8}

Program 32:

AIM: Display future leap years from current year to a final year entered by user.

```
import datetime
a = datetime.datetime.now()
a = int (a.year)
b = int (input (" Enter final year :"))
print (" \n Leap year :")
for i in range (a, b+1):
    if (i % 4 == 0):
        print (i)
```

Output:-
Enter final year : 2029
Leap Years:

2024

2028

Program 33:

AIM: Generate positive list of numbers from a given list of integers.

```
list1= [-3, -5, 7, 10, -5, -9, 11]
pos= list ()
for i in list1:
    if i>0:
        pos. append (i)
print ('Original list:', list1)
print ('Positive integer list:', pos)
```

Output

Original list: [-3, -5, 7, 10, -5, -9, 11]

Positive integer list: [7, 10, 11]

Program 34.

AIM: Find biggest of 3 numbers entered.

```
a= int (input ('Enter 1st number:'))
b= int (input ('Enter 2nd number:'))
c= int (input ('Enter 3rd number:'))
if a>b and b>c:
    print (a,' is the biggest number')
elif b>a and b>c:
    print (b, ' is the biggest number')
else:
    print (c,' is the biggest number')
```

Output:-

Enter 1st number : 29

Enter 2nd number : 25

Enter 3rd number : 14

29 is the biggest number.

Program 35:

AIM: Create a list of colors from comma-separated color names entered by user. Display first and last colors.

```
Colors = (input ('Enter colors separated by commas:')).
                                    split (',')
print ('First color:', colors[0])
print ('Last color:', colors [len (colors) - 1])
```

Output:

Enter colors seperated by commas: Yellow, Green, Red

First color: Yellow

Last color: Red.

Program 36:

AIM: Print out all colors from color-list 1 not contained in color-list 2.

```
colors1= set (raw_input ('Enter colors seperated by commas:')).
                                    split (',')
colors 2 = set (raw_input ('Enter colors seperated by commas:')).
                                    split (',')
print ('Colors in color-list1 not contained in
                        color-list2 are: ',
  list (colors1. difference (colors2)))
```

Output:

Enter colors seperated by commas: Black, Yellow, Lavender, Red, Orange

Enter colors seperated by commas: Green, white, Grey, Blue, Pink.

Colors in color-list1 not contained in color-list2 are:

['Orange', 'Red', 'Black', 'Yellow', 'Lavender']

Program 37:

AIM: Create a single string seperated with space from two strings by swapping the character at position.

```
str1 = input ('Enter a string:')
str2 = input ('Enter another string:')
str3 = str2[0] + str1[1:] + ' ' + str1[0] + str2[1:]
print (str3)
```

Output:-

Enter a string: happy

Enter another string: smile

sappy hmile.

Program: 38

AIM: Create a package graphics with modules rectangle, circle, and sub-package 3D-graphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module. Write programs that finds area and perimeter of figures by different importing statements.

```
def area (r):
    print ('Area of circle with radius', r, 'is:
        ', '%.2f' % (3.14*r*r), 'sq. units')
def circumference (r):
    print ('Circumference of circle with radius',
        r, 'is:', '%.2f' % (3.14*2*r), 'units')

def area (a,b):
    print ('Area of rectangle with sides', a, 'and',
```

```
b, ' is: ', '%.2f' % (a*b), 'sq.units')

def perimeter (a,b):
    print ('Perimeter of rectangle with sides', a, 'and',
        b, 'is:', '%.2f' % (2* (a+b)), 'units')

def area (l,b,h):
    print ('Total surface area of cuboid with
dimensions', l, ',', b, ',', h, 'is:', '%.2f' % (2*
        ((l*b)+ (b*h) + (l*h))), 'sq.units')
def perimeter (l,b,h):
    print ('Perimeter of cuboid with dimensions', l,
', ', b, ', ', h, 'is:', '%.2f' % (4* (l+b+h)), '
                                units')

def area (r):
    print ('Area of sphere with radius', r, 'is:', '
        %.2f' % (4* (3.14*r*r)), 'sq. units')
```

```
def perimeter (r):
    print ('perimeter of (great circle of) sphere with radius', r, 'is:', '%.2f' % (2*3.14*r), 'units')

import circle
from rectangle import *
from Graphics.-3D-graphics import cuboid, sphere
a=float (input ('Enter length of the rectangle:'))
b=float (input ('Enter breadth of the rectangle:'))
area(a,b)
r= float (input ('Enter the radius of circle:'))
circle. area (r)
l= float (input ('Enter length of cuboid:'))
b= float (input ('Enter breadth of cuboid:'))
h= float (input ('Enter height of cuboid:'))
cuboid. area (l, b, h)
r= float (input ('Enter the radius of sphere:'))
sphere. area (r)
```

```
import circle
from rectangle import *
from Graphics.-3D-graphics import cuboid, sphere
a= float (input ('Enter length of the rectangle:'))
b= float (input ('Enter breadth of the rectangle:'))
perimeter (a,b)
r= float (input ('Enter the radius of the circle:'))
circle.circumference (r)
l= float (input ('Enter length of the cuboid:'))
b= float (input ('Enter breadth of the cuboid:'))
h= float (input ('Enter height of the cuboid:'))
cuboid.perimeter (l,b,h)
r= float (input ('Enter the radius of sphere:'))
sphere.perimeter (r)
```

Output:-

Perimeter of a circle with radius 10 is 62.83185

Area of a circle with radius 10 is 314.1592

Area of a Rectangle with length and width 10 is: 100.

Perimeter of Rectangle with length and width 10 is: 40

Area of cuboid with length, width, height 10 is: 600

Perimeter of a cuboid with length, width, height 10 is: 120

Area of sphere with radius 10 is: 1256.637061

Perimeter of a sphere with radius 10 is: 62.8318530

Program 39:

AIM: Create Rectangle class with attributes length and breadth & methods to find area and perimeter. Compare two Rectangle objects by their area.

```
class Rectangle:
def __init__ (self, l,b):
    self. length =l
    self. breadth = b

def area (self):
        return self. length * self. breadth

def perimeter (self):
        return 2* (self. length + self. breadth)

def cmp (self , obj):
        if self . area () > obj . area ():
            print ('Rectangle with length= ', self. length,
' and breadth = ', self. breadth, ' has the greater area')
```

```
elif self.area() < obj.area():
    print('Rectangle with length=', obj.length, 'and
breadth=', obj.breadth, 'has the greater area')
else:
    print('They have equal area!')

r1 = Rectangle(11, 3)
r2 = Rectangle(56, 23)
r1.cmp(r2)
```

Output:

Rectangle with length = 56 and breadth = 23 has greater area.

Program 40

AIM: Create a class Time with private attributes hour, minute, and second. Overload '+' operator to find sum of a time.

```
class Time:
    def __init__(self, hh=0, mm=0, ss=0):
        self.__hour=hh
        self.__minute=mm
        self.__second=ss
    def __add__(self, other):
        second = int((self.__second + other.__second)%
                                                60)
        minute = int((self.__minute + other.__minute)%60+
                                    (self.__second +
                                    other.__second/60)).
        hour = int((self.__hour + other.__hour)%24 +
            (self.__minute + other.__minute)/60)
```

```
print ('Time [hh:mm:ss] ', hour, ':', minute, ':' second)

T1 = Time (7, 35, 60)

T2 = Time (9, 45, 20)

T1 + T2
```

Output:-

Time [hh:mm:ss] 17:21:20.

Program 41

AIM: Write a python program to read a file line by line and store it into a list.

```
def file_read (fname):
    with open (fname) as f:
        # x Content_List is the list that contain the
read lines
            c= f. readlines ()
            print (c)
        # print (len (c))

file read ("file1.txt")
```

Output:

['Python is an interpretted programming language.\n',
'Used as a supporting language.']

Program 42:

AIM: Python program to copy odd lines of one file to another.

```
a= open ('file1.txt', 'r')
b= open ('hello1.txt', 'w')
c= a.readlines ()
for i in range (0, len(c)):
  if (i%2 ! = 0):
     b.write (c[i])

  else:
     pass
  b.close ()
b= open ('file1.txt', 'r')
d= b.read ()
print (d)
a.close ()
b.close ()
```

Output:-

Python is a programming language

It is easy to use

Program 43

AIM: Write a Python program to read each row from a given csv file and print a list of strings.

```
import csv
with open ('download.csv', newline='') as csvfile:
    d= csv.reader (csvfile, delimiter ='',
                    quotechar ='|')

    for r in d:
        print (',' join (r))
```

Output:

1. "Eldon, Base, for, stackable, storage, shelf, platinum"
2. "1.7, cubic, foot, compact, "cube", office, Refrigerator",
3. "Cardinal, slant-D®, Ring, Binder, Heavy, Gauge, Vinyl",
4. "R300, clay, Roxendel, 483, 1198-97, 195-99, 3-99,"