Experiment No.: 2

Aim

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

CO1

Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
x= int(input("Enter the starting Year:"))
y= int(input("Enter the end year:"))
print("The Following are the Leap Years")
for i in range(x,y):
 if((i%400==0)or((i%100!=0)and(i%4==0))):
    print(i)
```

Output Screenshot

```
C:\Users\deepe.LAPTOP-HQF6HJQJ\PycharmProjects\Deependra\venv\Scripts\python.exe "C:/Users/deepe.LAPTOP-HQF6HJQJ/PycharmProjects/Deependra/Program
Enter the starting Year:2010
Enter the end year:2024
The Following are the Leap Years
2012
2016
2020
```

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 3

Aim

List comprehensions:

(a) Generate positive list of numbers from a given list of integers

(b) Square of N numbers

(c) Form a list of vowels selected from a given word

CO1

Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

(A)

Procedure

```
list1=[12,34,-16,75,-44]
print("List items are:",list1)
print("Positive numbers are:")
for num in list1:
    if (num>0):
        print(num)
```

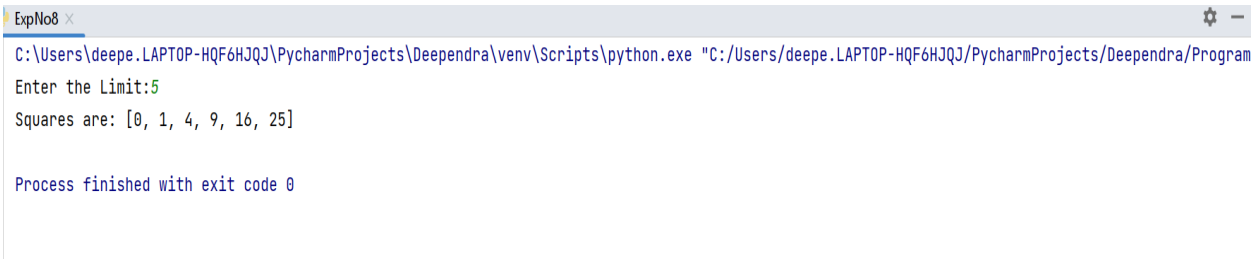
Output Screenshot

(B)

Procedure

```
n=int(input("Enter the Limit:"))
squares=[i*i for i in range(0,n+1)]
print("Squares are:",squares)
```

Output Screenshot

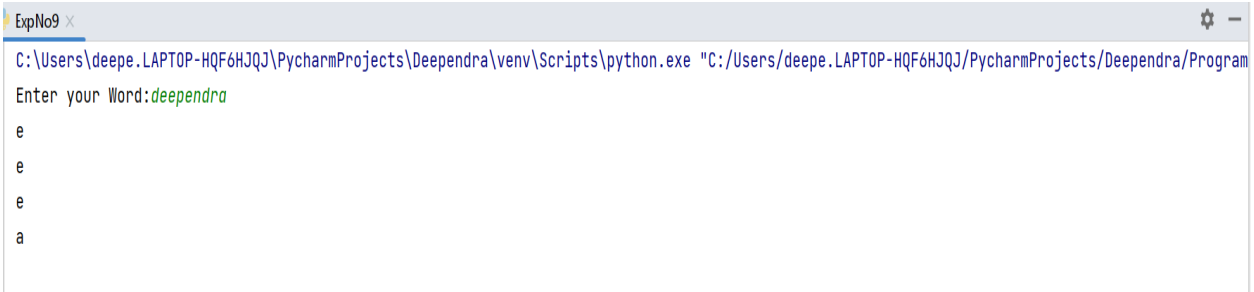


(C)

Procedure

```
word=input("Enter your Word:")
for letter in word:
    if letter in "aeiou":
        print(letter)
```

Output Screenshot



(D)

Procedure

```
list1=[]
list2=["Windows","Linux"]
for item in list2:
    for ele in item:
        result=ord(ele)
        list1.append(result)
print(list1)
```

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 4

Aim

Count the occurrences of each word in a line of text.

CO1

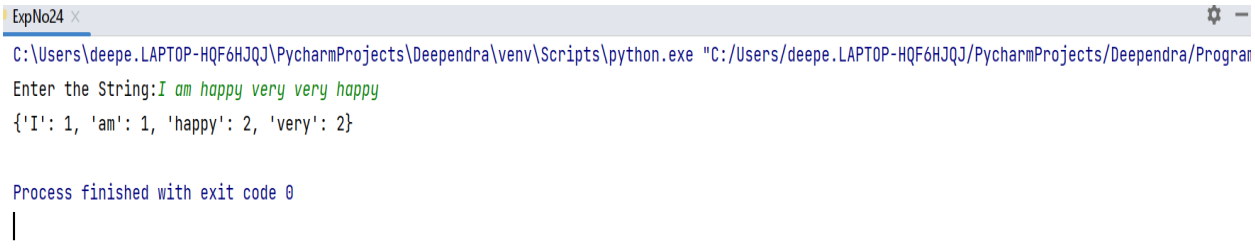
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
str=input("Enter the String:")
counts = dict()
str = str.split()

for i in str:
    if i in counts:
        counts[i] += 1
    else:
        counts[i] = 1

print(counts)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 5

Aim

Prompt the user for a list of integers. For all values greater than 100, store ‘over’ instead.

CO1

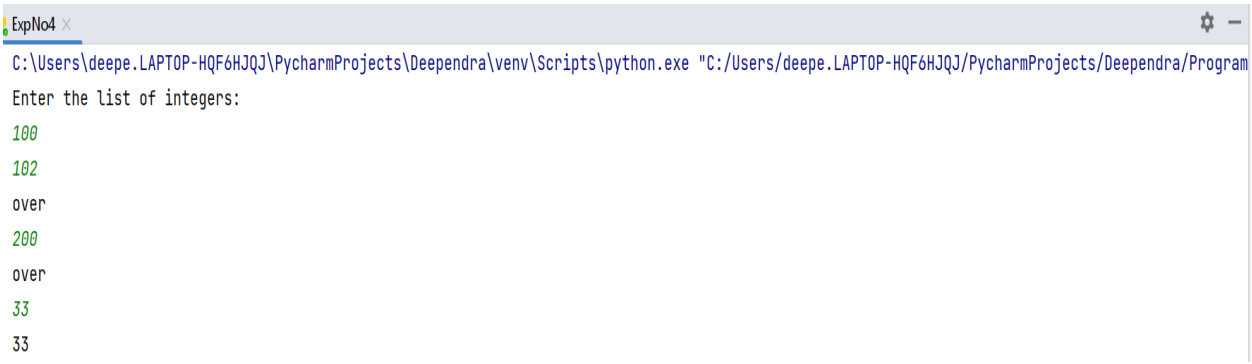
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
x=int(input("Enter the list of integers:\n"))
for i in range(0,x):
    a=(int(input()))
    if(a>100):
        a="over"
        print(a)
    else:
        print(a)

print(counts)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 6

Aim

Store a list of first names. Count the occurrences of ‘a’ within the list

CO1

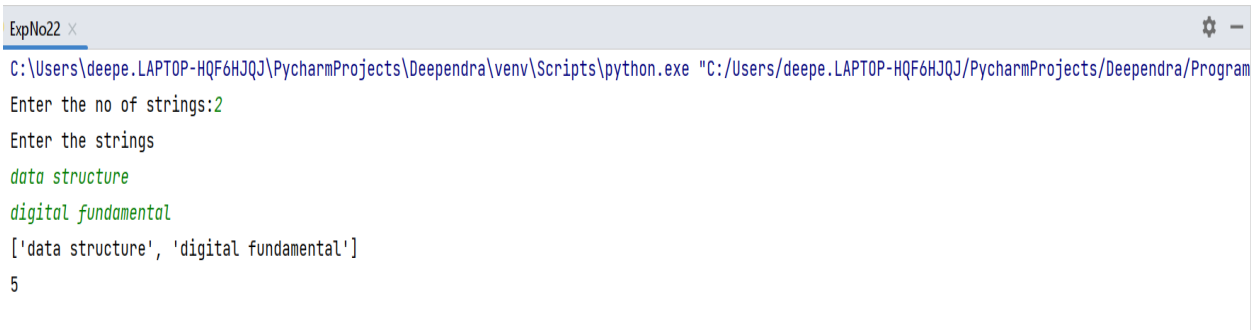
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
a=int(input("Enter the no of strings:"))
print("Enter the strings")
list=[]
count=0
for i in range(0,a):
    ele=input()
    list.append(ele)
print(list)
for i in list:
    for j in i:
        if j == 'a':
            count=count+1
print(count)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 7

Aim

Enter 2 lists of integers. Check (a) Whether list are of same length (b) whether list sums to same value (c) whether any value occur in both

CO1

Understands basics of Python Programming language including input/output functions, operators, basic and collection data types

Procedure

```
list1=[]
n = int(input("Enter number of elements : "))
print("Enter the elements:")
for i in range(0, n):
    i = int(input())
    list1.append(i)
print(list1)
list2=[]
k = int(input("Enter number of elements : "))
print("Enter the elements:")
for i in range(0, k):
    i = int(input())
    list2.append(i)
print(list2)
total=0
final=0
a=len(list1)
b=len(list2)
if(a==b):
    print("list1 and list2 are of same length")
```

```
for i in range(0, len(list1)):
    total = total + list1[i]
print("sum of list1 is:",total)

for j in range(0,len(list2)):
    final=final+list2[j]
print("sum of list2 is:",final)
z=total+final

print("sum of 2 lists:",z)
if(total==final):
    print("list sums to the same value")
else:
    print("sum of both list aren't the same")
c=list(set(list1).intersection(list2))
print("common elements are:",c)
```

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 8

Aim

Get a string from an input string where all occurrences of first character replaced with '$', except first character.

CO1

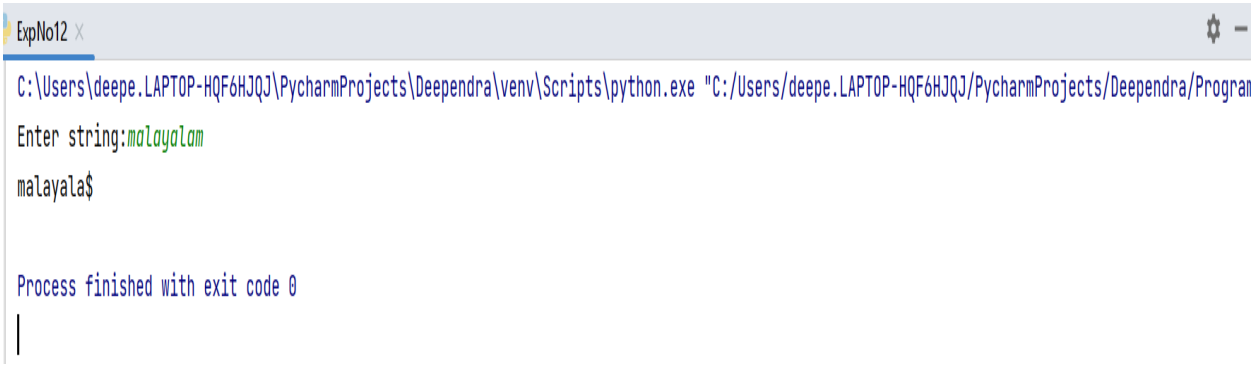
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
n=input("Enter string:")
a=n[0]
for i in n:
    if i==a:
        n=n.replace(i,"$")
        n=a+n[1:]
print(n)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 9

Aim

Create a string from given string where first and last characters exchanged.

[eg: python -> nythop]

CO1

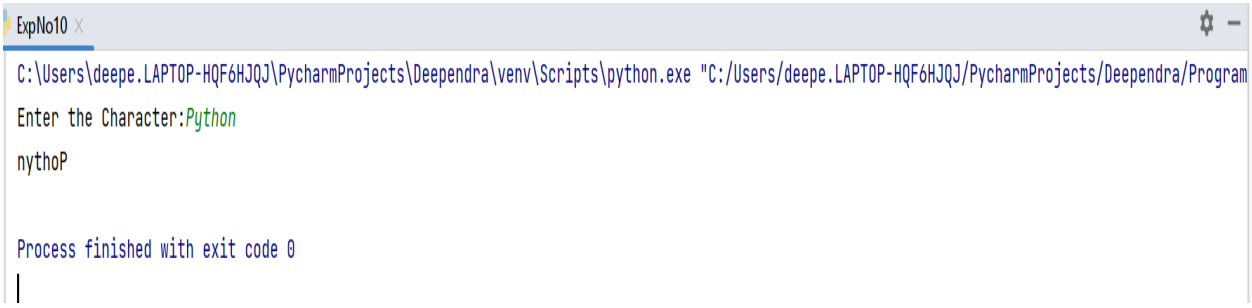
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
str=input("Enter the Character:")
a=str[-1]
b=str[0]
c=str[1:-1]
print(a+c+b)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 10

Aim

Accept the radius from user and find area of circle.

CO1

Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

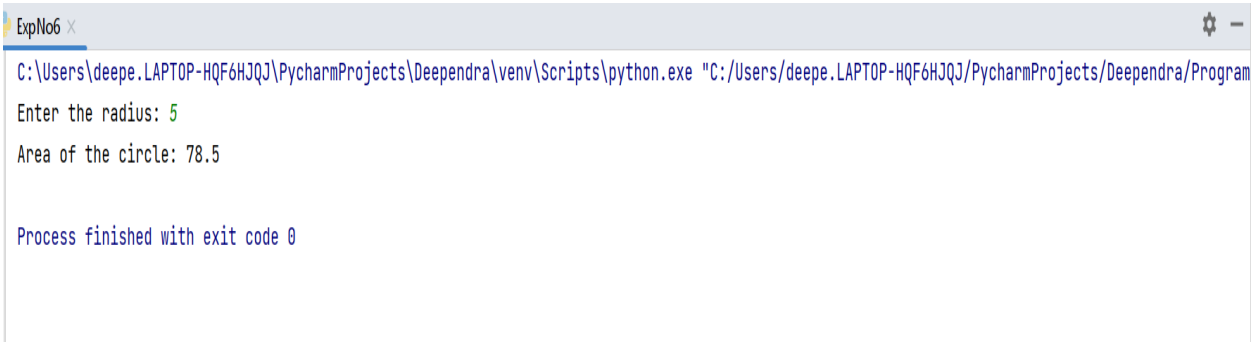
π = 3.14

Radius = float (input ("Enter the radius: "))

a = π * Radius * Radius

print ("Area of the circle:" , a)

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 11

Aim

Find biggest of 3 numbers entered.

CO1

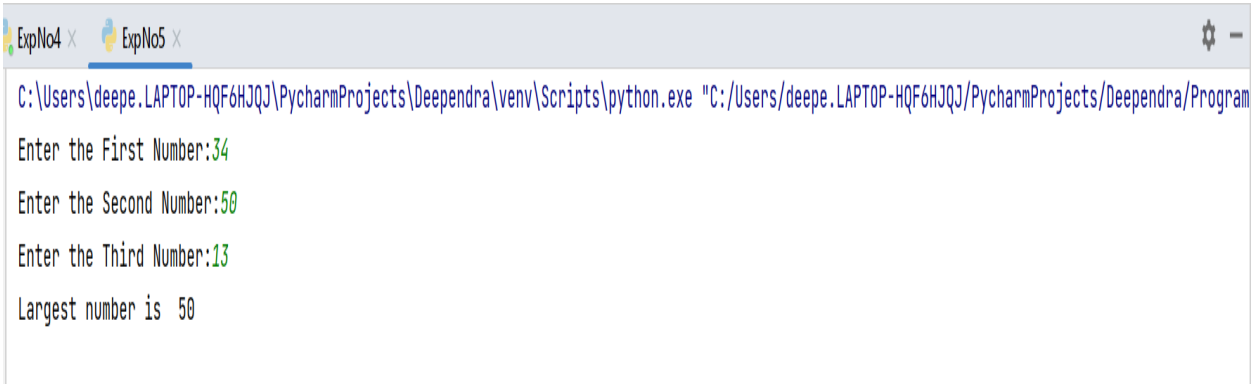
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
x=int(input("Enter the First Number:"))
y=int(input("Enter the Second Number:"))
z=int(input("Enter the Third Number:"))
if(x>=y)and(x>=z):
   largest=x
elif (y>=x)and(y>=z):
   largest=y
else:
   largest=z

print("Largest number is ",largest)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 12

Aim

Accept a file name from user and print extension of that.

CO1

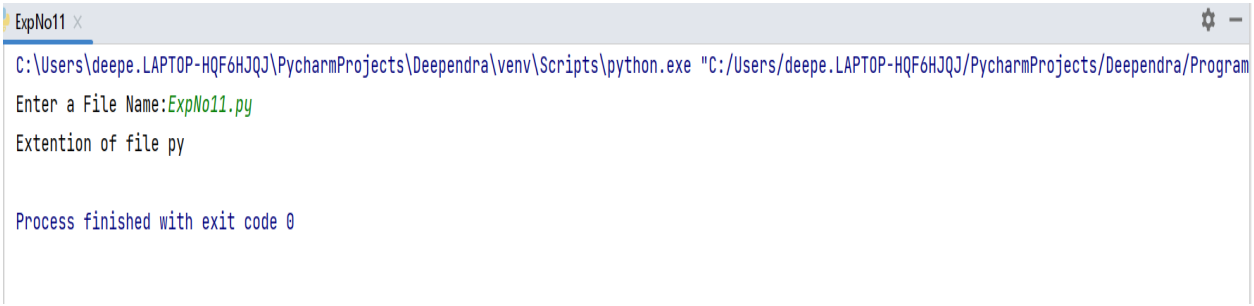
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
file1=input("Enter a File Name:")
text=file1.split('.')
print("Extention of file",text[1])
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 13

Aim

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

CO1

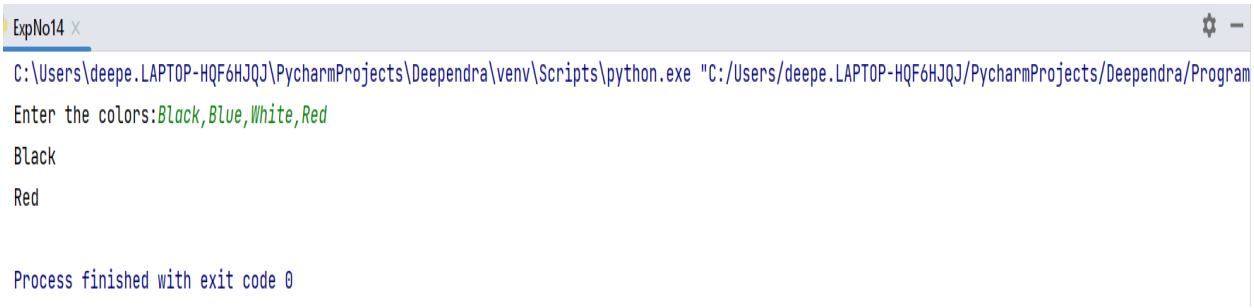
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
color=input("Enter the colors:")
colorList=color.split(',')
print(colorList[0])
print(colorList[-1])
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 14

Aim

Accept an integer n and compute n+nn+nnn.

CO1

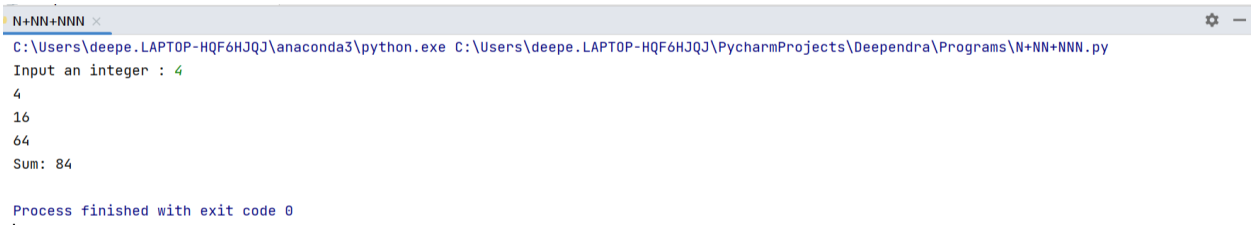
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
n = int(input("Input an integer : "))
sum = 0
for i in range(3):
  h = int(pow(n, i+1))
  print(h)
  sum = sum + h
print("Sum :",sum)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 15

Aim

Print out all colors from color-list1 not contained in color-list2.

CO1

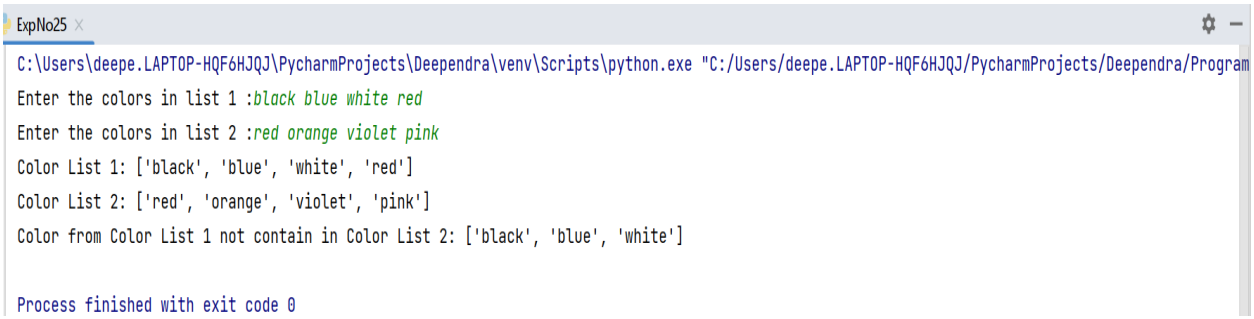
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
color_list1=input("Enter the colors in list 1 :")
color_list2=input("Enter the colors in list 2 :")
list1=color_list1.split()
list2=color_list2.split()
list3=[]

print("Color List 1:",list1)
print("Color List 2:",list2)

for i in list1:
    if i not in list2:
        list3.append(i)
print("Color from Color List 1 not contain in Color List 2:",list3)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 16

Aim

Create a single string separated with space from two strings by swapping the character at position 1.

CO1

Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
str1=input("Enter first string:")
str2=input("Enter second string:")
a=str1[0]
b=str2[0]
newstr1=b+str1[1:]
newstr2=a+str2[1:]
print(newstr1+" "+newstr2)
```

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 17

Aim

Sort dictionary in ascending and descending order

CO1

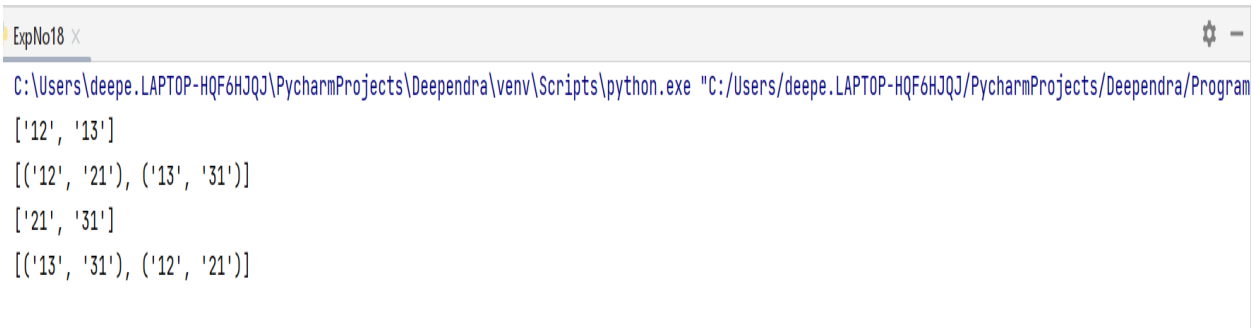
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
dict1={'13':'31','12':'21'}
sorted_a=sorted(dict1.keys())
print(sorted_a)
sorted_a=sorted(dict1.items())
print(sorted_a)
sorted_a=sorted(dict1.values())
print(sorted_a)
sorted_a=sorted(dict1.items(),reverse=True)
print(sorted_a)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 18

Aim

Merge two dictionaries

CO1

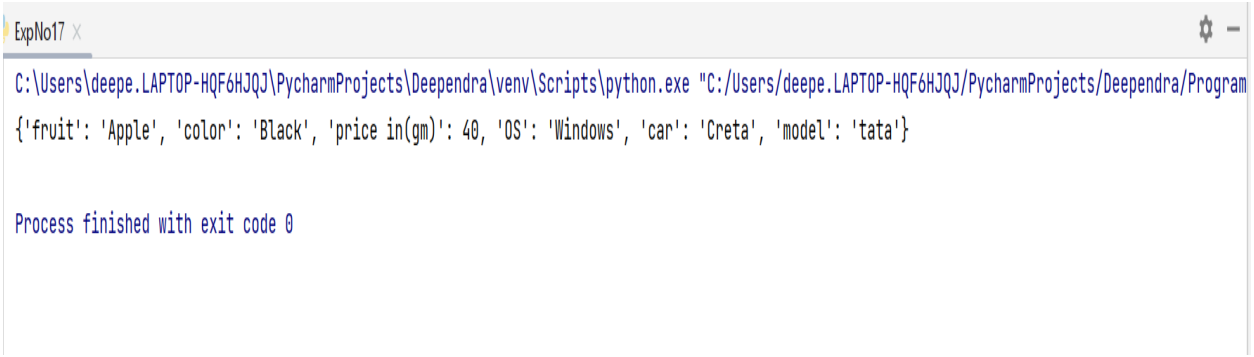
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
d1={'fruit':'Apple','color':'Black','price in(gm)':40}
d2={'OS':'Windows','car':'Creta','model':'tata'}
d1.update(d2)
print(d1)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 19

Aim

Find gcd of 2 numbers

CO1

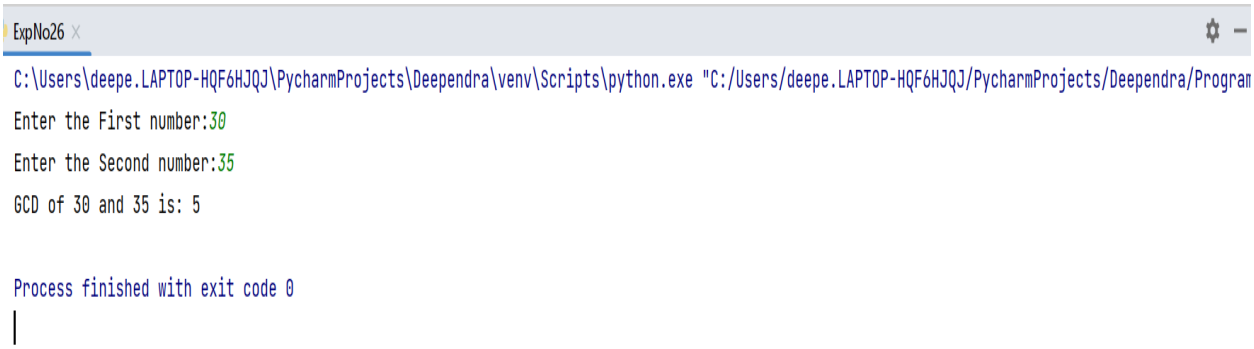
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
import math
num1=int(input("Enter the First number:"))
num2=int(input("Enter the Second number:"))

print("GCD of " + str(num1) + " and " + str(num2) + " is:",math.gcd(num1,num2))
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 20

Aim

From a list of integers, create a list removing even numbers

CO1

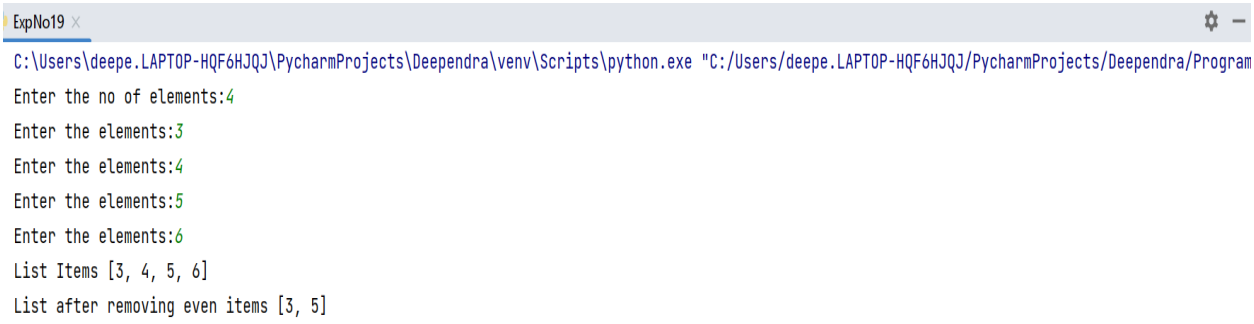
Understands basics of Python Programming language including input/output

functions, operators, basic and collection data types

Procedure

```
a=int(input("Enter the no of elements:"))
list=[]

for i in range(a) :
    n = int(input("Enter the elements:"))
    list.append(n)
print("List Items",list)

for i in list:
    if(i%2==0):
        list.remove(i)
print("List after removing even items",list)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 21

Aim

Program to find the factorial of a number

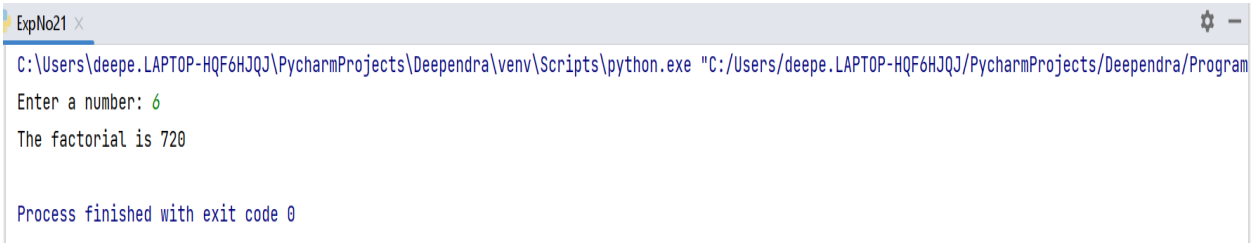
CO2

Implement decision making, looping constructs and functions

Procedure

```
n=int(input("Enter a number: "))
fact=1
if n<0:
   print("Factorial for negative numbers does not exist")
elif n==0:
   print("The factorial of 0 is 1")
else:
   for i in range(1,n+1):
      fact=fact*i
   print("The factorial is",fact)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 22

Aim

Generate Fibonacci series of N terms

CO2

Implement decision making, looping constructs and functions

Procedure

```
num=int(input("Enter the no of terms:"))
n1=0
n2=1
print("Fibonacci Series:", n1, n2, end=" ")
for i in range(2, num):
    n3 = n1 + n2
    n1 = n2
    n2 = n3
    print(n3, end=" ")
```

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 23

Aim

Find the sum of all items in a list

CO2

Implement decision making, looping constructs and functions

Procedure

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 24

Aim

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

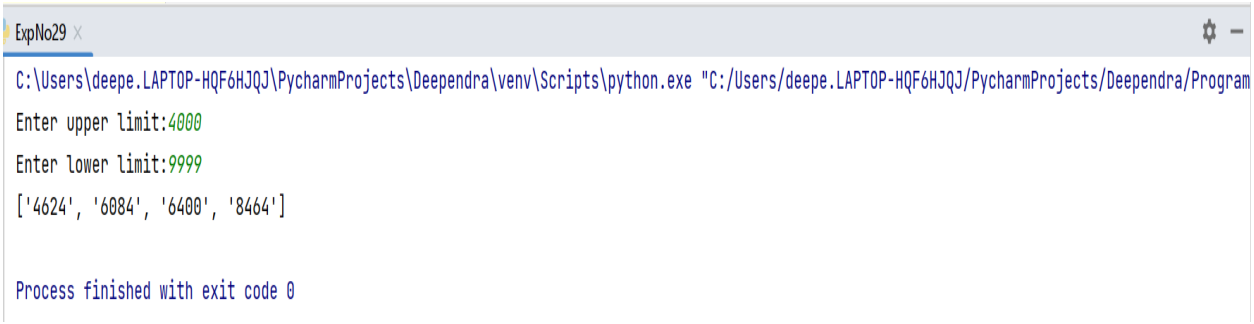
CO2

Implement decision making, looping constructs and functions

Procedure

```
a=int(input("Enter upper limit:"))
b=int(input("Enter lower limit:"))
lists=[]

for i in range (a,b+1):

  for j in range(1,i):
    if j * j == i:
      string=str(i)
      if int(string[0]) % 2 == 0 and int(string[1]) % 2 == 0 and int(string[2]) % 2 == 0 and
int(string[3]) % 2 == 0:
          lists.append(string)
print(lists)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 25

Aim

Display the given pyramid with step number accepted from user.

Eg: N=4

1

2 4

3 6 9

4 8 12 16

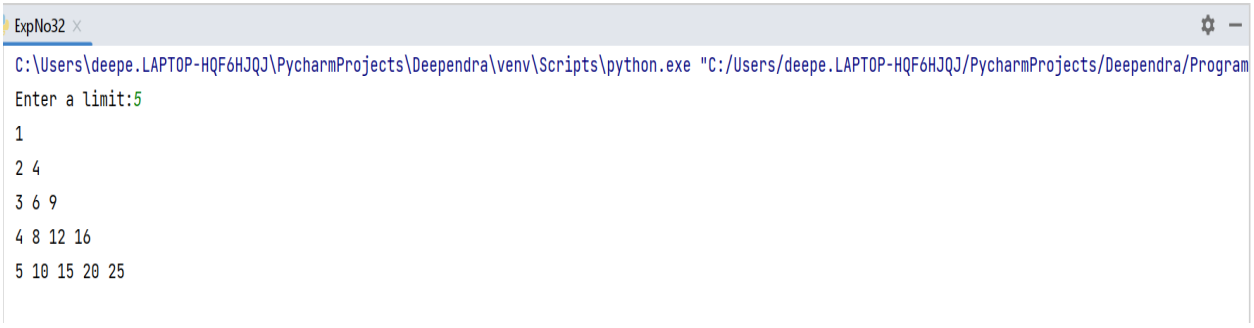
CO2

Implement decision making, looping constructs and functions

Procedure

```
n=int(input("Enter a limit:"))
for i in range(1,n+1):
    for j in range(1,i+1):

        print(i*j,end=" ")
    print()
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 26

Aim

Count the number of characters (character frequency) in a string

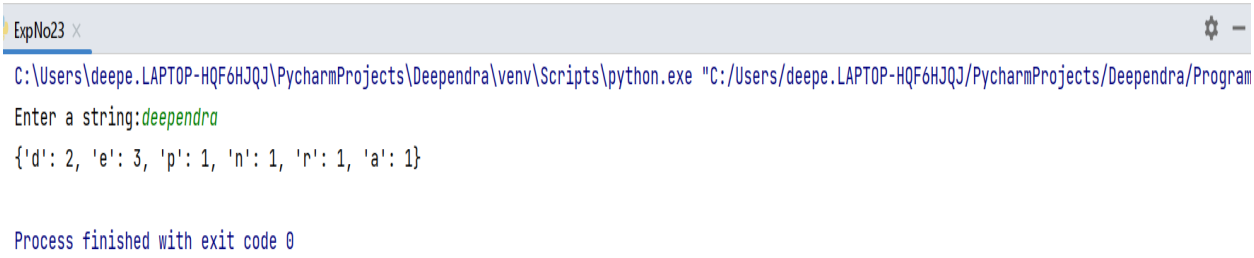
CO2

Implement decision making, looping constructs and functions

Procedure

```
str=input("Enter a string:")
dict={}
for i in str:
    if i in dict:
        dict[i]+=1
    else:
        dict[i]=1
print(dict)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 27

Aim

Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’

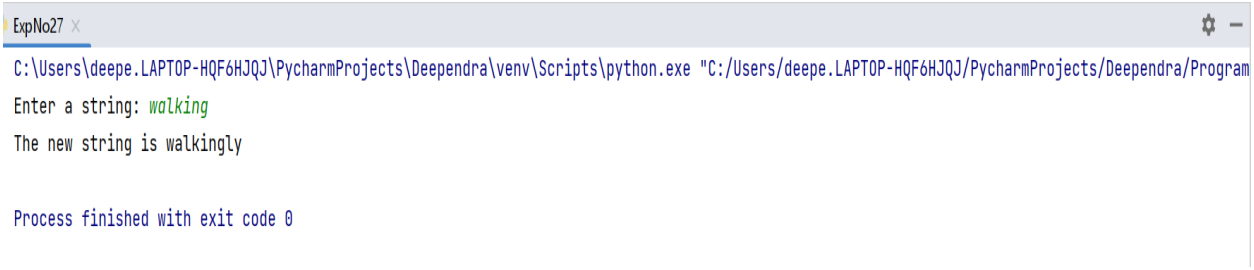
CO2

Implement decision making, looping constructs and functions

Procedure

```
str=input("Enter a string: ")
if str[-3:] == 'ing':
    print("The new string is",str + "ly")
else:
    print("The new string is",str + "ing")
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 28

Aim

Accept a list of words and return length of longest word

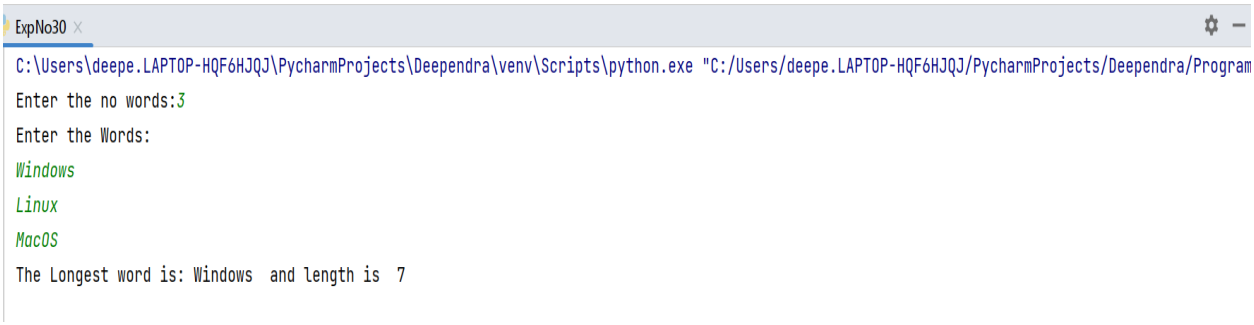
CO2

Implement decision making, looping constructs and functions

Procedure

```
def longestLength(a):
    max1 = len(a[0])
    temp = a[0]

    for i in a:
        if (len(i) > max1):
            max1 = len(i)
            temp = i

    print("The Longest word is:", temp,
          " and length is ", max1)

a=[]
n = int(input("Enter the no words:"))
print("Enter the Words:")
for j in range(0,n):
    ele=input()
    a.append(ele)
longestLength(a)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 29

Aim

Construct following pattern using nested loop

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

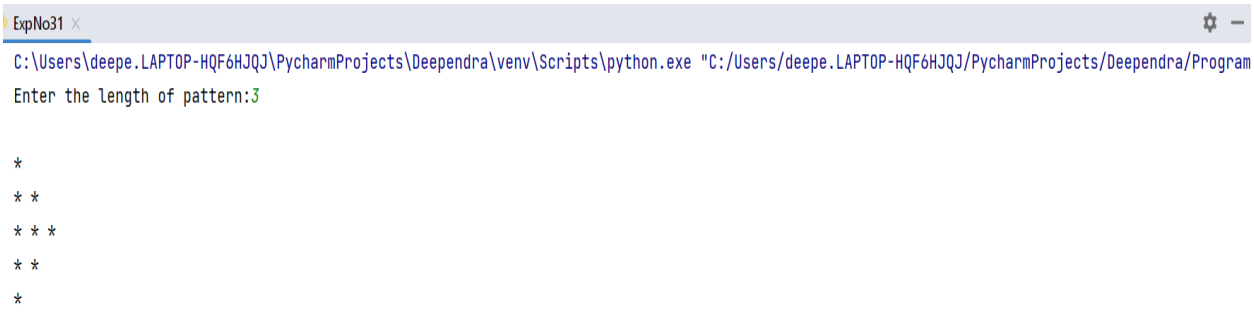
CO2

Implement decision making, looping constructs and functions

Procedure

```
n= int(input("Enter the length of pattern:"))
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 Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 30

Aim

Generate all factors of a number.

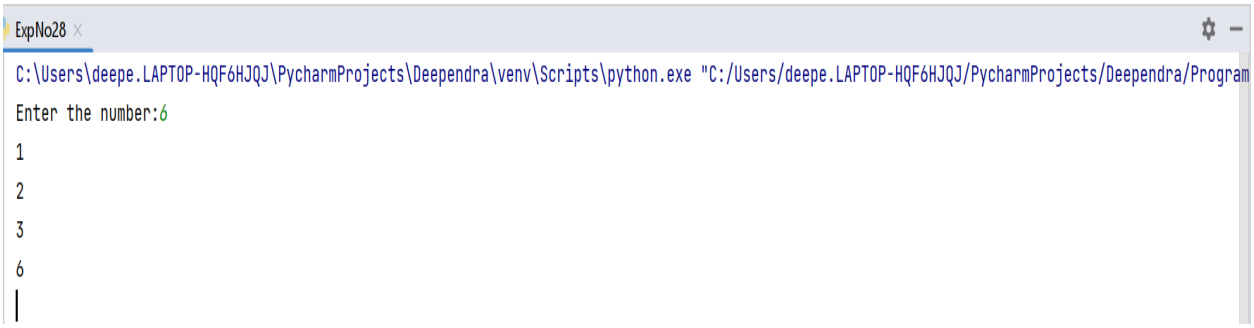
CO2

Implement decision making, looping constructs and functions

Procedure

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

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 31

Aim

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

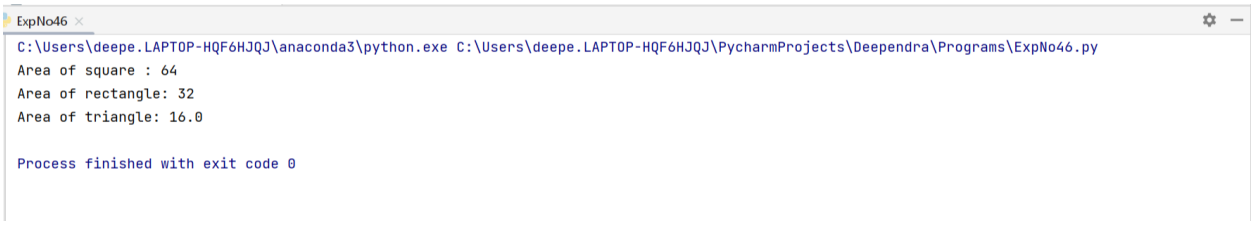
CO2

Implement decision making, looping constructs and functions

Procedure

```
area_square = lambda x:x*x
area_rectangle = lambda x,y: x*y
area_triangle = lambda x,y: 1/2*x*y
a=8
b=4
print("Area of square :",area_square(a))
print("Area of rectangle:",area_rectangle(a,b))
print("Area of triangle:",area_triangle(a,b))
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 32

Aim

Work with built-in packages

CO3

Design modules and packages - built in and user defined packages

Procedure

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 33

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. (Include selective import of modules and import * statements)

CO3

Design modules and packages - built in and user defined packages

Procedure

Circle.py

```
def circleArea(r):
    z=3.14*r*r
    print("Area of circle:",z)

def circlePerimeter(r):
    y=2*3.14*r
    print("Perimeter of circle:",y)
```

Rectangle.py

```
def rectangleArea(l,b):
    x=l*b
    print("Area of rectangle:",x)

def rectanglePerimeter(l,b):
    y=2*(l+b)
    print("Perimeter of rectangle:",y)
```

Cuboid.py

```
def cuboidArea(l,w,h):
    q=2*l*w+2*l*h+2*h*w
    print("Area of cuboid:",q)

def cuboidPerimeter(l,w,h):
    w=4*(l+w+h)
    print("Perimeter of cuboid:",w)
```

Sphere.py

```
def sphereArea(r):
    t=4*3.14*r*r
    print("Area of sphere:",t)

def spherePerimeter(r):
    u=4*3.14*r
    print("Perimeter of sphere:",u)
```

ExpNo33

```
import Graphics.circle
import Graphics.rectangle
import Graphics.ThreeDGraphics.cuboid
import Graphics.ThreeDGraphics.sphere

Graphics.circle.circleArea(6)
Graphics.circle.circlePerimeter(6)

Graphics.rectangle.rectangleArea(10,5)
Graphics.rectangle.rectanglePerimeter(10,5)
```

Graphics.ThreeDGraphics.cuboid.cuboidArea(3,6,6)

Graphics.ThreeDGraphics.cuboid.cuboidPerimeter(3,6,6)

Graphics.ThreeDGraphics.sphere.sphereArea(2)

Graphics.ThreeDGraphics.sphere.spherePerimeter(2)

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 34

Aim

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

CO4

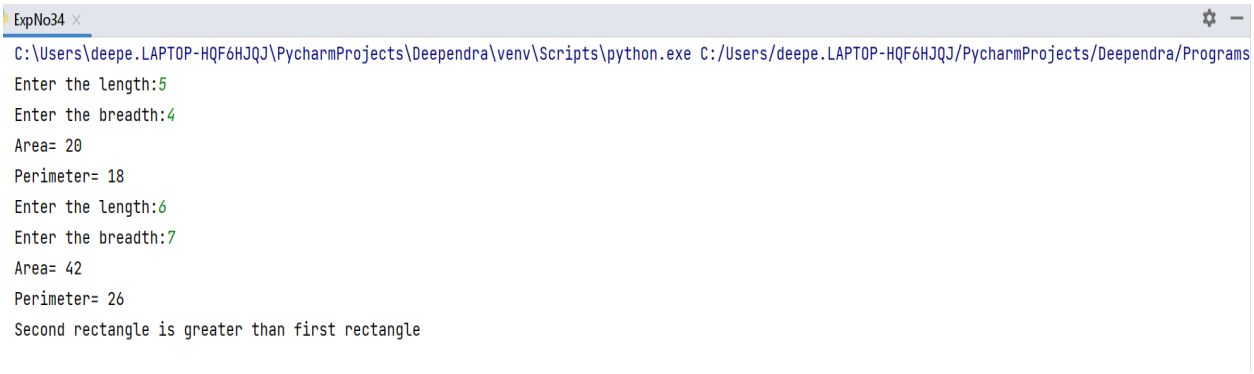
Implement object-oriented programming and exception handling.

Procedure

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

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

l=int(input("Enter the length:"))
b=int(input("Enter the breadth:"))
o=Rectangle(l,b)
x=o.area()
y=o.perimeter()
print("Area=",x)
print("Perimeter=",y)

l1=int(input("Enter the length:"))
b1=int(input("Enter the breadth:"))
p=Rectangle(l1,b1)
x1=p.area()
```

```
y1=p.perimeter()
print("Area=",x1)
print("Perimeter=",y1)

if(x>x1):
    print("First rectangle is greater than second rectangle")
else:
    print("Second rectangle is greater than first rectangle")
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 35

Aim

Create a Bank account with members account number, name, type of account and balance. Write constructor and methods to deposit at the bank and withdraw an amount from the bank.

CO4

Implement object-oriented programming and exception handling.

Procedure

```
class Bank:
    def __init__(self,accno,name,accty):
        self.accno=accno
        self.name=name
        self.accty=accty
        self.bal=0
    def showaccount(self):
        print("Account holder name",self.name)
        print("The Account no is", self.accno)
        print("Account type",self.accty)
        print("Account Balence",self.bal)

    def deposit(self,d1):
        self.bal=self.bal+d1
        return  self.bal
    def withdraw(self,w1):
        self.bal=self.bal-w1
        return self.bal
```

```
b=int(input("Enter Your Account no:"))
a=input("Enter Account name:")
c=input("Enter Account type:")
d=Bank(b,a,c)
d.showaccount()
while(True):
    print("MENU")
    print("\n 1.Deposit")
    print("\n 2.Withdraw")
    c=int(input("Enter choice:"))
    f=0
    if(c==1):
        f=int(input("Enter the amount to deposit:"))
        print("Your total bank deposit is\n ",d.deposit(f))

    elif(c==2):
        g=int(input("Enter the amount to withdraw:"))
        if(g<f):
            print("Insufficient balance")
        else:
            print("Total balance amount is:\n",d.withdraw(g))
    else:
        print("Enter valid choice")
```

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 36

Aim

Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of 2 rectangles.

CO4

Implement object-oriented programming and exception handling.

Procedure

```
class Rectangle:
    def __init__(self, length, width):
        self.__length = length
        self.__width = width

    def area(self):
        return self.__length * self.__width

    def __lt__(self, other):
        return self.area() < other.area()

rect1 = Rectangle(4, 7)
rect2 = Rectangle(4, 5)

print("Area of rectangle 1 :",rect1.area())
print("Area of rectangle 2 :",rect2.area())

if rect1<rect2:
    print("Area of rectangle 1 less than Rectangle 2")
else:
    print("Area of rectangle 2 is less than rectangle 1")
```

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 37

Aim

Create a class Time with private attributes hour, minute and second. Overload '+' operator to find sum of 2 time.

CO4

Implement object-oriented programming and exception handling.

Procedure

```
class Time:
    def __init__(self, hour, minute, second):
        self.hour = hour
        self.minute = minute
        self.second = second

    def __str__(self):
        return f"{self.hour}:{self.minute}:{self.second}"

    def __add__(self, other):
        hour = self.hour + other.hour
        minute = self.minute + other.minute
        second = self.second + other.second

        if second >= 60:
            second -= 60
            minute += 1

        if minute >= 60:
            minute -= 60
            hour += 1
```

```
    return Time(hour, minute, second)

time1 = Time(1, 20, 30)
time2 = Time(2, 45, 15)

print(time1)
print(time2)
print(time1 + time2)
```

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 38

Aim

Create a class Publisher (name). Derive class Book from Publisher with attributes title and author. Derive class Python from Book with attributes price and no_of_pages. Write a program that displays information about a Python book. Use base class constructor invocation and method overriding.

CO4

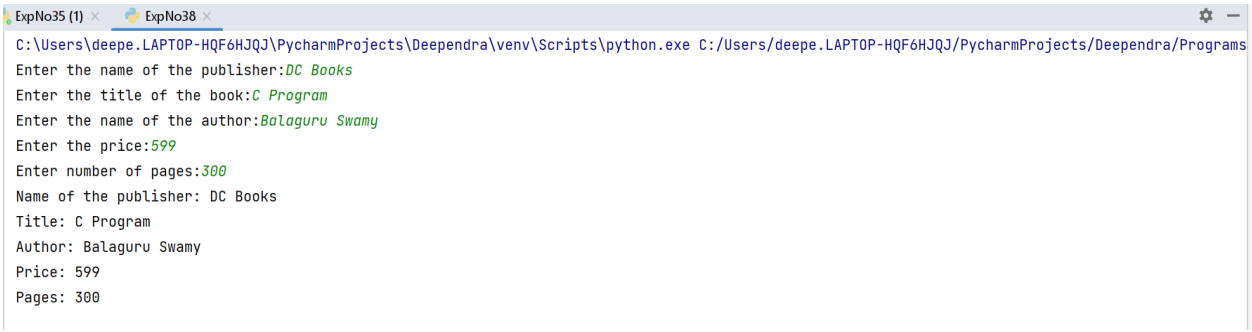
Implement object-oriented programming and exception handling.

Procedure

```
class Publisher:
    def getpublisher(self):
        self.name=input("Enter the name of the publisher:")

    def display(self):
        print("Name of the publisher:",self.name)

class Book(Publisher):
    def getbook(self):
        self.title=input("Enter the title of the book:")
        self.author=input("Enter the name of the author:")

    def display1(self):
        print("Title:",self.title)
        print("Author:",self.author)

class Python(Book):
    def bookdetails(self):
```

```
    self.price=int(input("Enter the price:"))
    self.pages=int(input("Enter number of pages:"))

  def display2(self):

      print("Price:",self.price)
      print("Pages:",self.pages)


obj1=Python()
obj1.getpublisher()
obj1.getbook()
obj1.bookdetails()
obj1.display()
obj1.display1()
obj1.display2()
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 39

Aim

Write a Python program to read a file line by line and store it into a list.

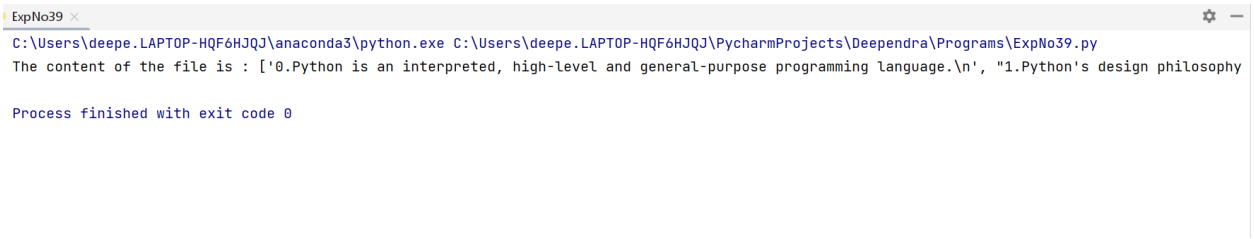
CO5

Create files and form regular expressions for effective search operations on strings and files.

Procedure

```
f=open("demo.txt","r")
list=f.readlines()
print("The content of the file is :",list)
f.close()
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 40

Aim

Python program to copy odd lines of one file to other

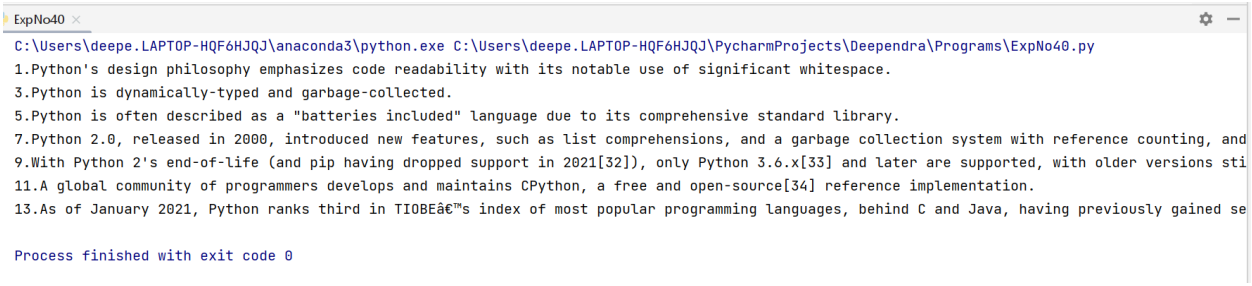
CO5

Create files and form regular expressions for effective search operations on strings and files.

Procedure

```
f=open("demo.txt","r")
f1=open("odd.txt","w")
content=f.readlines()
for i in range(0,len(content)):
    if(i%2!=0):
        f1.write(content[i])
    else:
        pass
f.close()
f1.close()

f=open("odd.txt","r")
c=f.read()
print(c)
f.close()
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 41

Aim

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

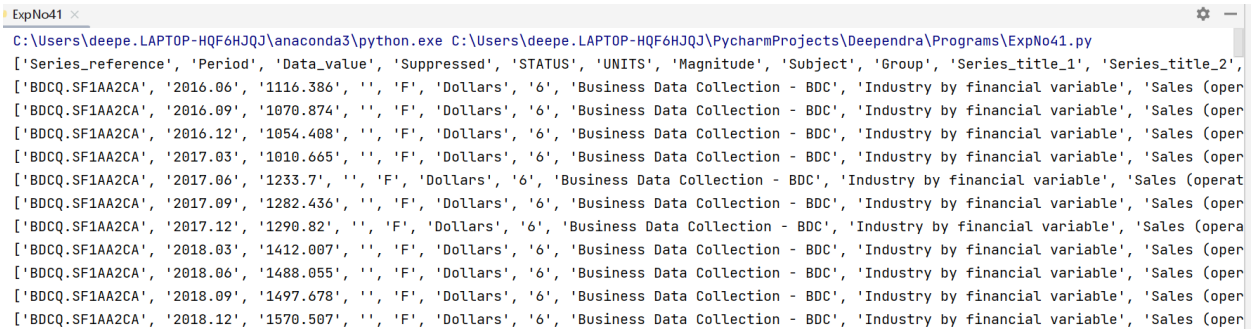
CO5

Create files and form regular expressions for effective search operations on strings and files.

Procedure

```
import csv
with open("dep.csv","r")as f:
    r=csv.reader(f)
    for i in r:
        print(i)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 42

Aim

Write a Python program to read specific columns of a given CSV file and print the content of the columns.

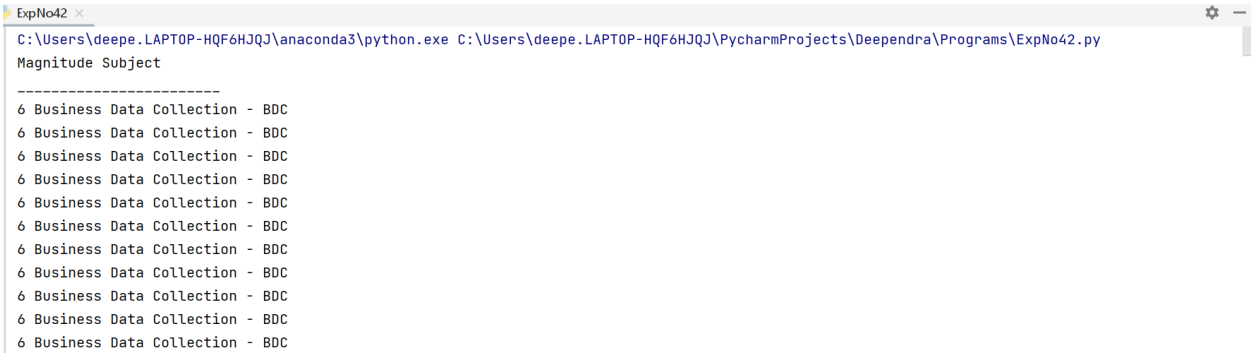
CO5

Create files and form regular expressions for effective search operations on strings and files.

Procedure

```
import csv
with open("dep.csv",newline='')as f:
    d=csv.DictReader(f)
    print("Magnitude Subject")
    print("_____")
    for i in d:
        print(i['Magnitude'],i['Subject'])
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 43

Aim

Write a Python program to write a Python dictionary to a csv file. After writing the CSV file read the CSV file and display the content.

CO5

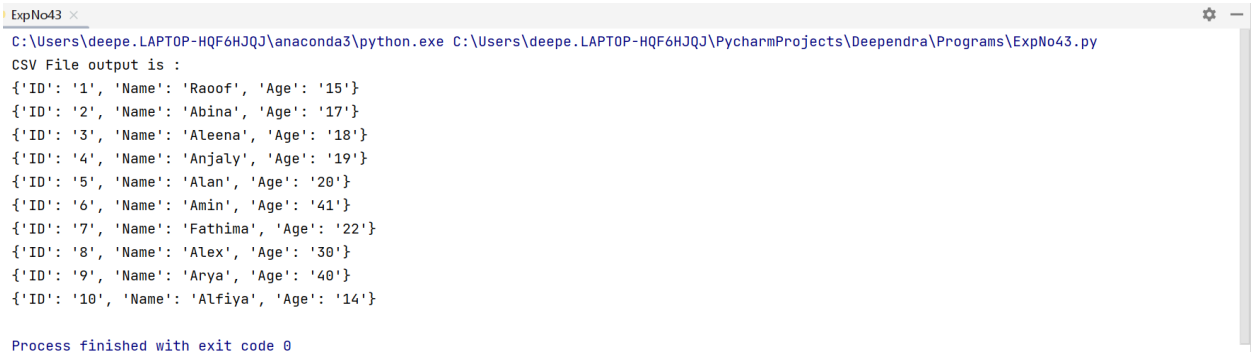
Create files and form regular expressions for effective search operations on strings and files.

Procedure

```
import csv
c_col=['ID','Name','Age']
dict_data=[{'ID':1,'Name':'Raoof','Age':15},
        {'ID':2,'Name':'Abina','Age':17},
        {'ID':3,'Name':'Aleena','Age':18},
        {'ID':4,'Name':'Anjaly','Age':19},
        {'ID':5,'Name':'Alan','Age':20},
        {'ID':6,'Name':'Amin','Age':41},
        {'ID':7,'Name':'Fathima','Age':22},
        {'ID':8,'Name':'Alex','Age':30},
        {'ID':9,'Name':'Arya','Age':40},
        {'ID':10,'Name':'Alfiya','Age':14}]
try:
   with open("name.csv","w")as f:
      write=csv.DictWriter(f,fieldnames=c_col)
      write.writeheader()
      for i in dict_data:
         write.writerow(i)
except IOError:
   print("Input/Output Error")
```

```
d=csv.DictReader(open("name.csv"))
print('CSV File output is : ')
for i in d:
    print(i)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 44

Aim

Write a python program to read a CSV file and write the first and display the result in

List format

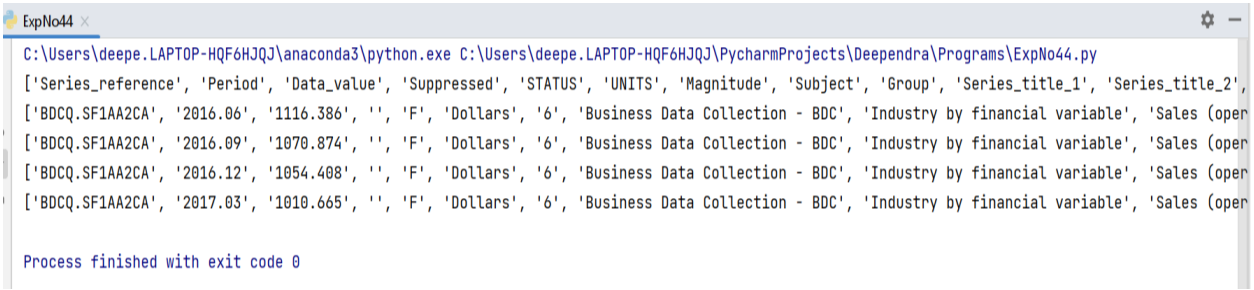
CO5

Create files and form regular expressions for effective search operations on strings and files.

Procedure

```
import csv
with open('dep.csv', 'r') as original_file:
    reader = csv.reader(original_file)

    with open('new.csv', 'w', newline='') as new_file:
        writer = csv.writer(new_file)

        for i in range(5):
            writer.writerow(next(reader))

    with open('new.csv', 'r') as new_file:
        reader = csv.reader(new_file)
        for row in reader:
            print(row)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 45

Aim

Write a python program to read a CSV file and write the odd rows to a new file and even rows another new file and finally display both.

CO5

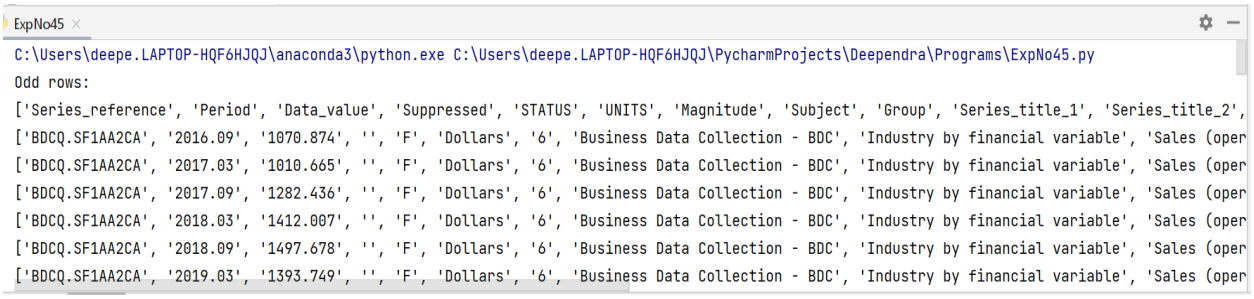
Create files and form regular expressions for effective search operations on strings and files.

Procedure

```
import csv

with open('dep.csv', 'r') as original_file:
   reader = csv.reader(original_file)


   with open('odd.csv', 'w', newline='') as odd_file:
      odd_writer = csv.writer(odd_file)


      with open('even.csv', 'w', newline='') as even_file:
         even_writer = csv.writer(even_file)


         i = 1
         for row in reader:
            if i % 2 == 1:
               odd_writer.writerow(row)
            else:
               even_writer.writerow(row)
            i += 1

   with open('odd.csv', 'r') as odd_file:
      reader = csv.reader(odd_file)

      print("Odd rows:")
      for row in reader:
         print(row)
```

```
with open('even.csv', 'r') as even_file:
    reader = csv.reader(even_file)

    print("Even rows:")
    for row in reader:
        print(row)
```

Output Screenshot



Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.

Experiment No.: 0

Aim

CO

Procedure

Output Screenshot

Result

The program was executed and the result was successfully obtained. Thus CO1 was obtained.