**FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)TM**

**HORMIS NAGAR, MOOKKANNOOR, ANGAMALY-683577**



**FOCUS ON EXCELLENCE**

**20MCA131 PROGRAMMING LAB LABORATORY RECORD**

**Name: BIVINA M V**

**Branch: MASTER OF COMPUTER APPLICATIONS**

**Semester: 1 Batch: A Roll No: 45**

**MARCH 2022**

**FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)TM**

**HORMIS NAGAR, MOOKKANNOOR, ANGAMALY-683577**



**FOCUS ON EXCELLENCE**

**CERTIFICATE**

*This is to certify that this is a Bonafede record of the Practical work done by* ***BIVINA M V*** *in the* ***20MCA131 PROGRAMMING LAB*** *Laboratory**towards the partial fulfilment for the award of the Master Of Computer Applications during the academic year 2021-2022.*

Signature of Staff in Charge Signature of H O D

Name: Name:

**Date of University practical examination ………………………**

Signature of Signature of

Internal Examiner External Examiner

**CONTENT**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl No** | **Date of Experiment** | **Title of the Experiment** | **Page No:** | **Signature of Staff –In – Charge** |
| 1 | 23/10/21 | Display future leap years from current year to a final year entered by user. | 7 |  |
| 2 | 23/10/21 | List comprehensions:   1. Generate positive list of numbers from a given list of integers 2. Square of N numbers 3. Form a list of vowels selected from a given word 4. List ordinal value of each element of a word (Hint: use ord() to get ordinal values) | 8-11 |  |
| 3 | 23/10/21 | Count the occurrences of each word in a line of text. | 12 |  |
| 4 | 23/10/21 | Prompt the user for a list of integers. For all values greater than 100, store ‘over’ instead. | 13 |  |
| 5 | 10/11/21 | Store a list of first names. Count the occurrences of ‘a’ within the list | 14 |  |
| 6 | 10/11/21 | 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 | 15-18 |  |
| 7 | 10/11/21 | Get a string from an input string where all occurrences of first character replaced with ‘$’, except first character.  [eg: onion ->oni$n] | 19 |  |
| 8 | 10/11/21 | Create a string from given string where first and last characters exchanged. [eg: python - >nythop] | 20 |  |
| 9 | 10/11/21 | Accept the radius from user and find area of circle. | 21 |  |
| 10 | 11/11/21 | Find biggest of 3 numbers entered. | 22 |  |
| 11 | 11/11/21 | Accept a file name from user and print extension of that. | 23 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl No** | **Date of Experiment** | **Title of the Experiment** | **Page No:** | **Signature of Staff –In – Charge** |
| 12 | 11/11/21 | Create a list of colors from comma-separated color names entered by user. Display first and last colors. |  |  |
| 13 | 11/11/21 | Accept an integer n and compute n+nn+nnn. |  |  |
| 14 | 11/11/21 | Print out all colors from color-list1 not contained in color-list2. |  |  |
| 15 | 17/11/21 | Create a single string separated with space from two strings by swapping the character at position 1. |  |  |
| 16 | 17/11/21 | Sort dictionary in ascending and descending order. |  |  |
| 17 | 17/11/21 | Merge two dictionaries. |  |  |
| 18 | 17/11/21 | Find gcd of 2 numbers. |  |  |
| 19 | 17/11/21 | From a list of integers, create a list removing even numbers. |  |  |
| 20 | 25/11/21 | Program to find the factorial of a number. |  |  |
| 21 | 25/11/21 | Generate Fibonacci series of N terms. |  |  |
| 22 | 25/11/21 | Find the sum of all items in a list |  |  |
| 23 | 25/11/21 | Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square. |  |  |
| 24 | 2/12/21 | Display the given pyramid with step number accepted from user.  Eg: N=4  1   1. 4 2. 6 9   8 12 16 |  |  |
| 25 | 2/12/21 | Count the number of characters (character frequency) in a string. |  |  |
| 26 | 2/12/21 | Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’ |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl No** | **Date of Experiment** | **Title of the Experiment** | **Page No:** | **Signature of Staff –In – Charge** |
| 27 | 9/12/21 | Accept a list of words and return length of longest word. |  |  |
| 28 | 9/12/21 | Construct following pattern using nested loop  \*   * \* * \* \* * \* \* \* * \* \* \* \* * \* \* \* * \* \* * \* |  |  |
| 29 | 9/12/21 | Generate all factors of a number. |  |  |
| 30 | 29/1/22 | 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) |  |  |
| 31 | 13/1/22 | Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area. |  |  |
| 32 | 13/1/22 | 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. |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sl No | Date of Experiment | Title of the Experiment | Page No: | Signature of Staff –In – Charge |
| 33 | 13/1/22 | Create a class Rectangle with private attributes length and width. Overload ‘<’ operator to compare the area of 2 rectangles. |  |  |
| 34 | 20/1/22 | Create a class Time with private attributes hour, minute and second. Overload ‘+’ operator to find sum of 2 time. |  |  |
| 35 | 20/1/22 | 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. |  |  |
| 36 | 3/2/22 | Write a Python program to read a file line by line and store it into a list. |  |  |
| 37 | 3/2/22 | Write a Python program to read each row from a given csv file and print a list of string. |  |  |

# Course Outcome 1 Programs

**PROGRAM 1**

**AIM**

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

## PROGRAM

print("leap year from current year")

startyear=int(input("Enter the startyear:"))

endyear=int(input("Enter the endyear:"))

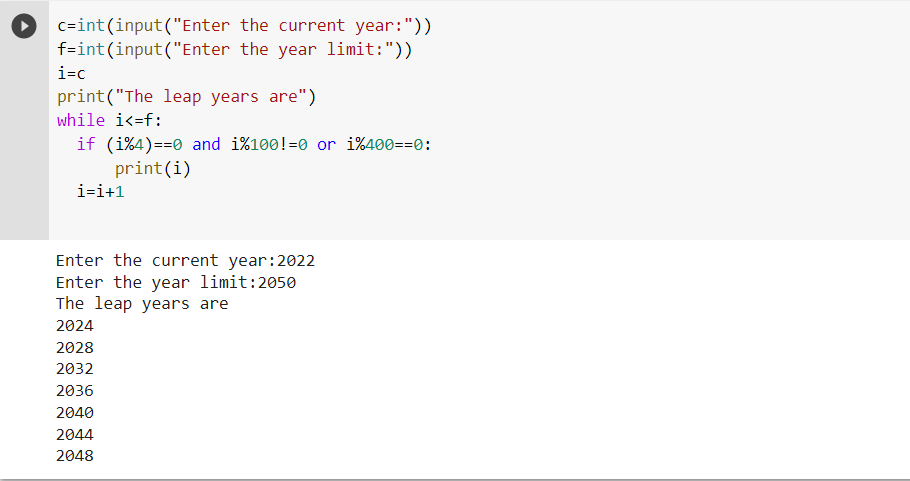
print("list of the leap years:")

for year in range (startyear,endyear):

if (0 == year % 4) and (0 != year % 100) or (0 == year % 400):

print(year)

## OUTPUT



## PROGRAM 2(a)

## AIM

List comprehensions:

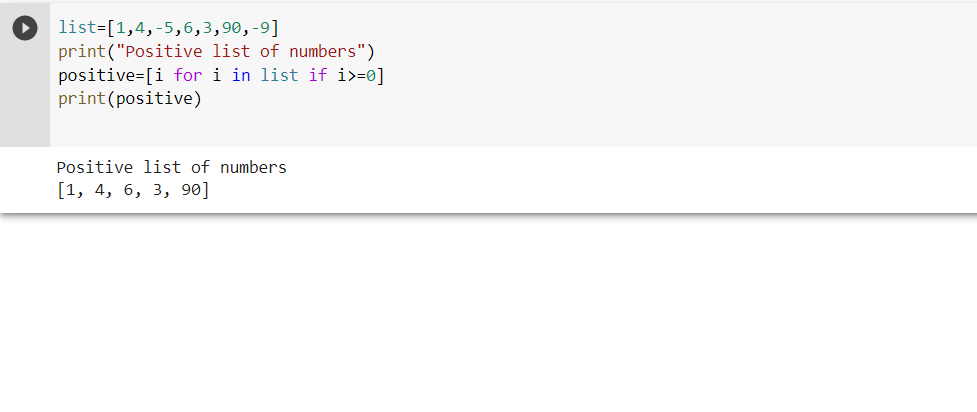
Generate positive list of numbers from a given list of integers

## PROGRAM

list=[1,4,-5,6,3,90,-9]

print("Positive list of numbers") positive=[i for i in list if i>=0] print(positive)

## OUTPUT



**PROGRAM 2(b)**

**AIM**

Square of N numbers

## PROGRAM

n=int(input("Enter the limit:")) i=1

while i<=n:

squ=(i\*i) print(i,"\*",i,"=",squ) i=i+1

## OUTPUT



**PROGRAM 2(c)**

**AIM**

Form a list of vowels selected from a given word

## PROGRAM

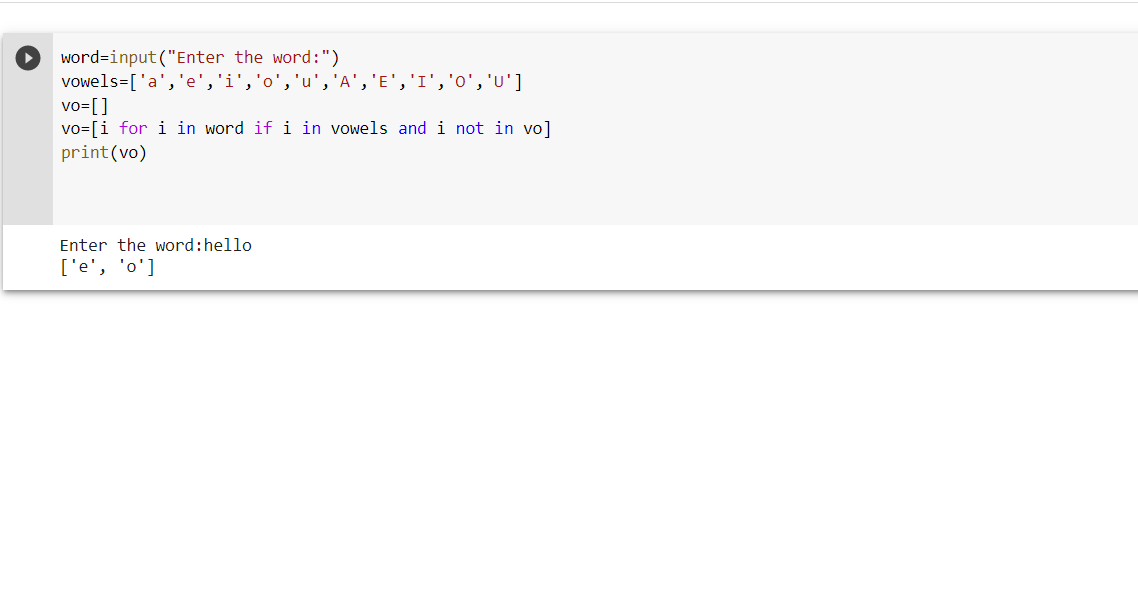
word=input("Enter the word:")

vo=[]

vowels=['a','e','i','o','u','A','E','I','O','U']

vo=[i for i in word if i in vowels and i not in vo] print(vo)

## OUTPUT



## PROGRAM 2(d)

## AIM

List ordinal value of each element of a word (Hint: use ord() to get ordinal values)

## PROGRAM

s=input("Enter the string:") print("Ordinal value")

for i in s:

print(i,"=",ord(i))

## OUTPUT



**PROGRAM 3**

**AIM**

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

## PROGRAM

s=input("Enter a string:")

count=dict()

w=s.split()

for i in w:

if i not in count:

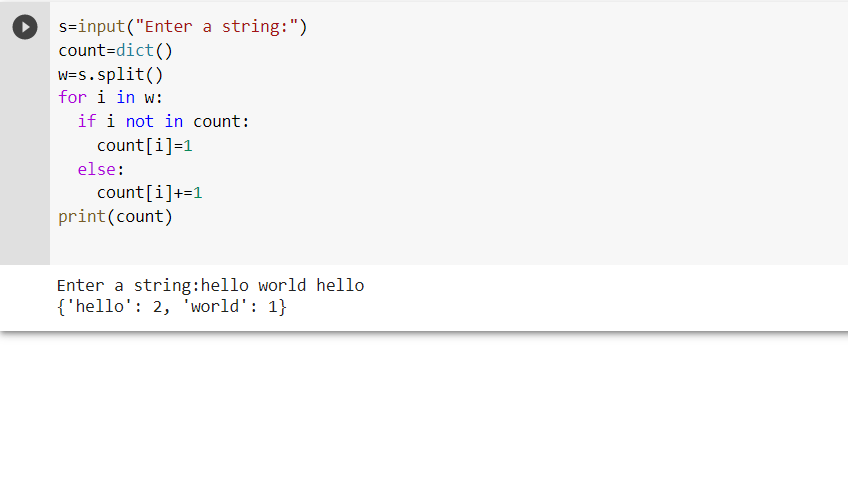
count[i]=1

else:

count[i]+=1

print(count)

## OUTPUT



## PROGRAM 4

## AIM

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

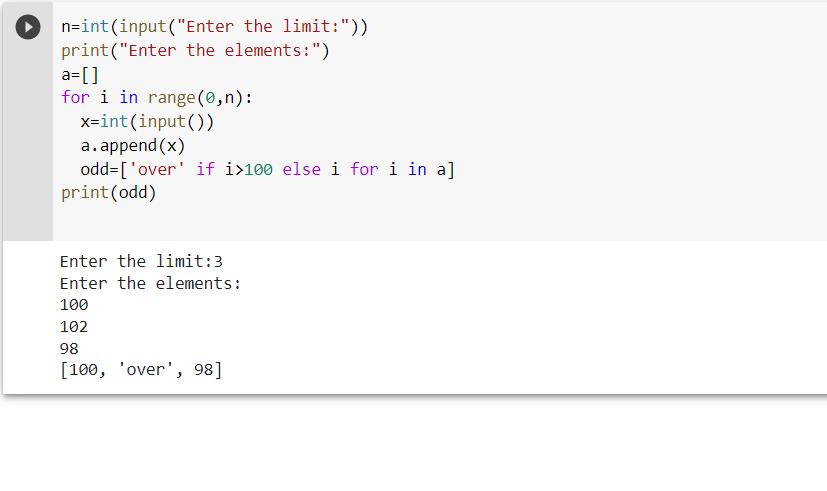
## PROGRAM

n=int(input("Enter the limit:")) print("Enter the elements:") a=[] for i in range(0,n): x=int(input())

a.append(x)

odd= ['over' if i>100 else i for i in a] print(odd)

## OUTPUT



**PROGRAM 5**

**AIM**

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

## PROGRAM

list1=['apple','orange','mango']

s=0

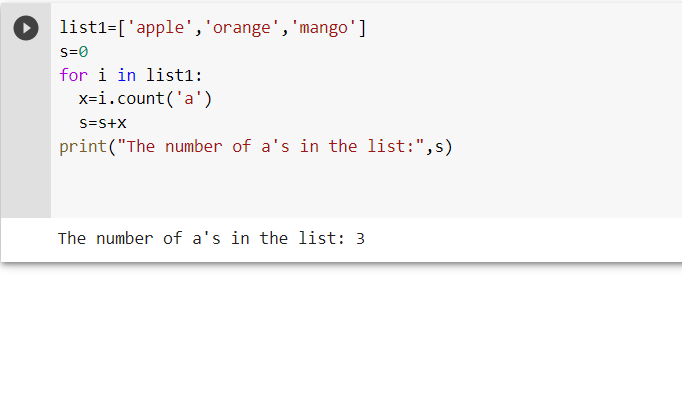
for i in list1:

x=i.count('a')

s=s+x

print("The number of a's in the list:",s)

## OUTPUT



**PROGRAM 6**

**AIM**

Enter 2 lists of integers. Check

(a) Whether list is of same length

(b) Whether list sums to same value

(c) Whether any value occur in both

## PROGRAM (a)

first=[1,3,6,9,10]

second=[2,4,7,8,]

x=len(first)

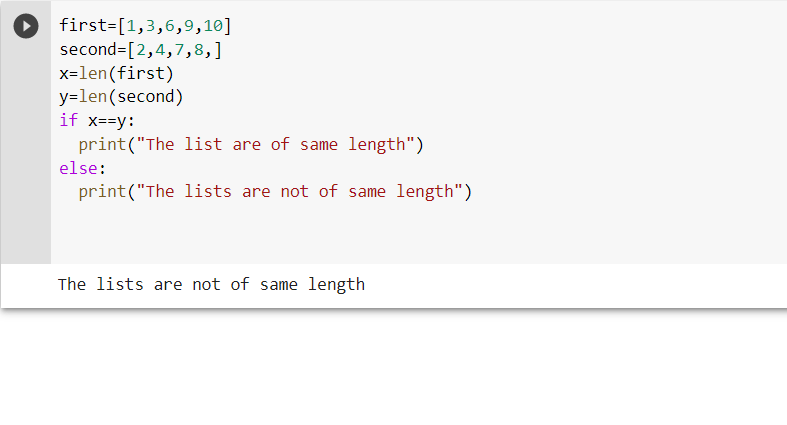
y=len(second)

if x==y:

print("The list are of same length")

else:

print("The lists are not of same length")

**OUTPUT**

## PROGRAM (b)

first=[3,5,7,9]

second=[4,6,6,8]

sum1=0

sum2=0

i=0

j=0

for i in first:

sum1=sum1+i

i=i+1

for j in second:

sum2=sum2+j

j=j+1

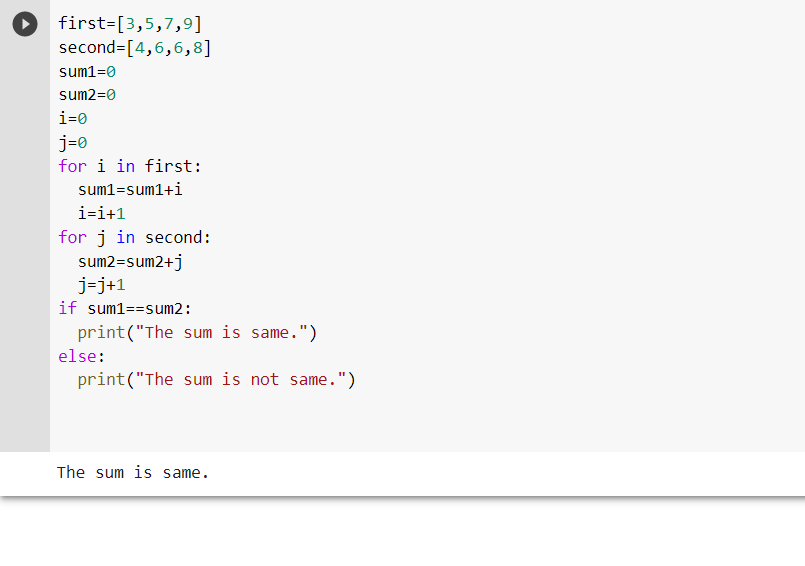
if sum1==sum2:

print("The sum is same.")

else:

print("The sum is not same.")

**OUTPUT**



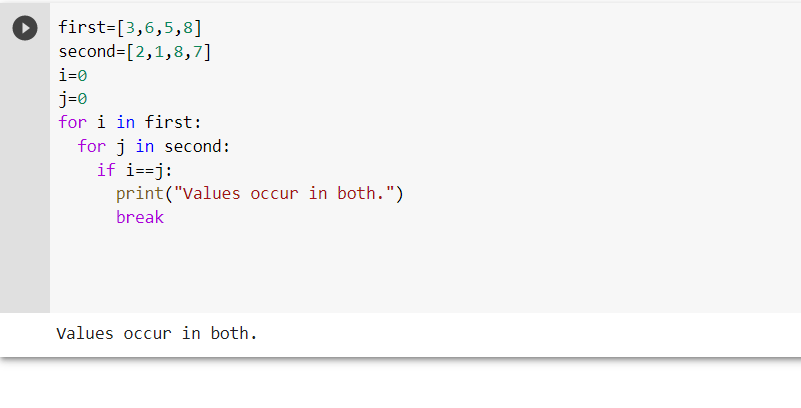
## PROGRAM (c)

first=[1,2,5,7,8] second=[2,1,8,8,7] i=0 j=0

for i in first: for j in second: if i==j:

print("Values occur in both.") break

## OUTPUT

.

**PROGRAM 7**

**AIM**

Get a string from an input string where all occurrences of first character replaced with

‘$’, except first character.

[eg: onion -> oni$n]

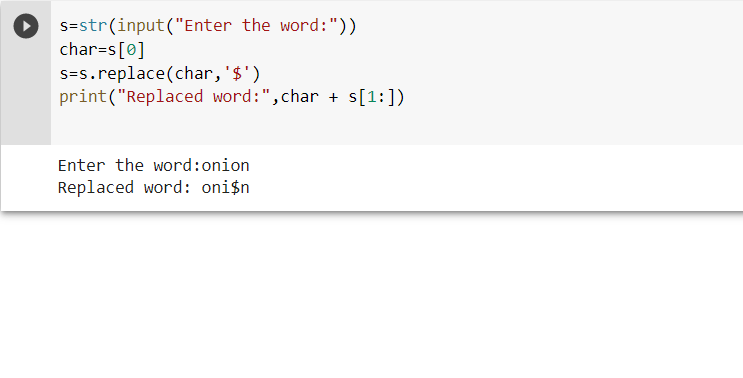
## PROGRAM

s=str(input("Enter the word:")) char=s[0]

s=s.replace(char,'$')

print("Replaced word:",char + s[1:])

## OUTPUT



**PROGRAM 8**

**AIM**

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

[eg: python -> nythop]

## PROGRAM

s=input("Enter the string:")

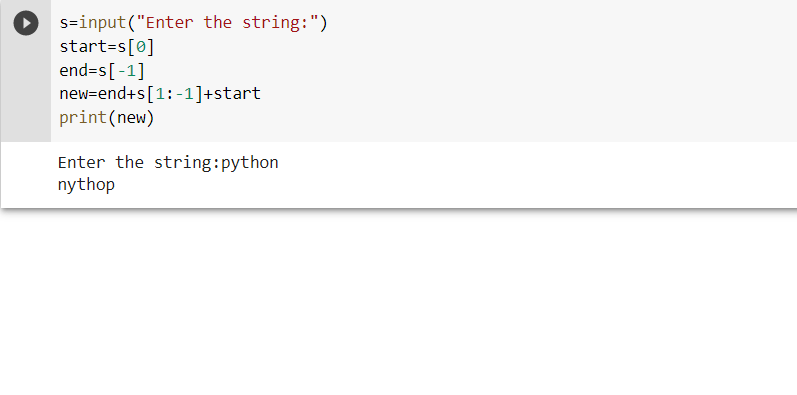
start=s[0]

end=s[-1]

new=end+s[1:-1]+start

print(new)

## OUTPUT



**PROGRAM 9**

**AIM**

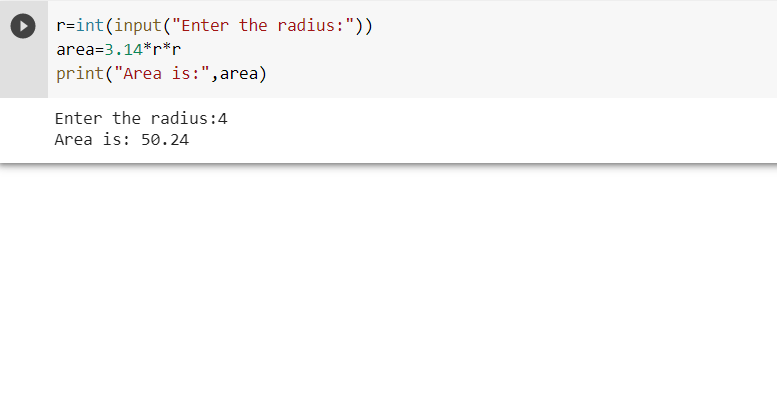
Accept the radius from user and find area of circle.

## PROGRAM

r=int(input("Enter the radius:")) area=3.14\*r\*r

print("Area is:",area)

## OUTPUT



**PROGRAM 10**

**AIM**

Find biggest of 3 numbers entered.

## PROGRAM

a=int(input("Enter 1st number:"))

b=int(input("Enter 2nd number:"))

c=int(input("Enter 3rd number:"))

if a>b and a>c:

print("The large no is",a)

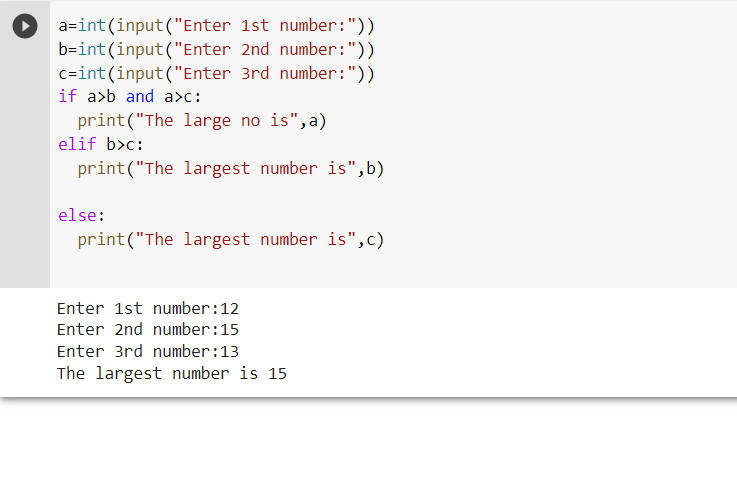
elif b>c:

print("The largest number is",b)

else:

print("The largest number is",c)

## OUTPUT



**PROGRAM 11**

**AIM**

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

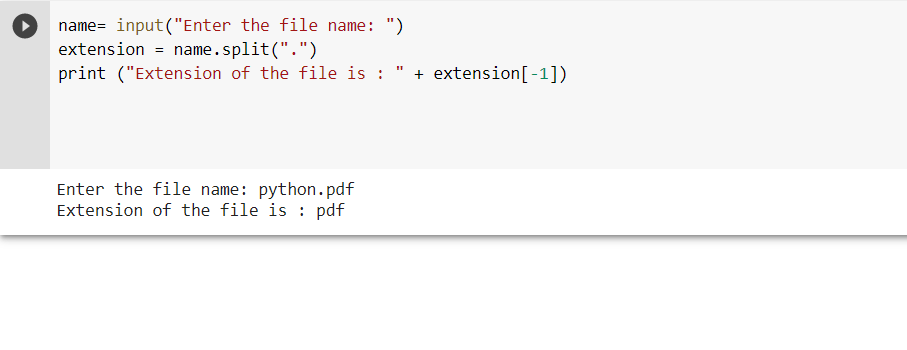
## PROGRAM

name= input("Enter the file name: ")

extension = name.split(".")

print ("Extension of the file is : " + extension[-1])

## OUTPUT



**PROGRAM 12**

**AIM**

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

## PROGRAM

color=input("Enter the colors:") seperate=color.split(",") print("First color:",seperate[0]) print("Last color:",seperate[-1])

## OUTPUT



**PROGRAM 13**

**AIM**

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

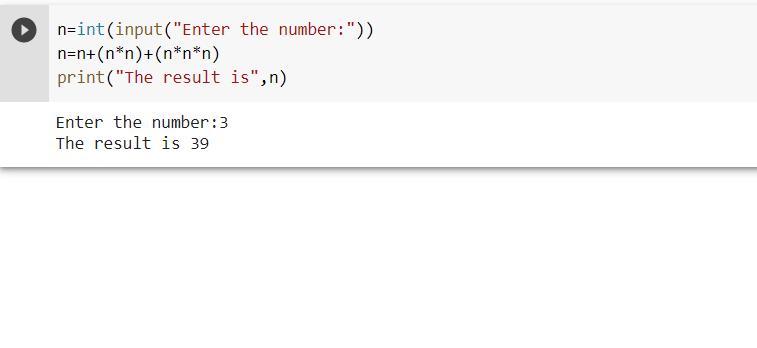
## PROGRAM

n=int(input("Enter the number:"))

n=n+(n\*n)+(n\*n\*n)

print("The result is",n)

## OUTPUT



**PROGRAM 14**

**AIM**

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

## PROGRAM

c1=['rose','pink','brown','black']

c2=['white','yellow','rose']

c3=set(c1).difference(c2)

print(c3)

## OUTPUT



**PROGRAM 15**

**AIM**

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

## PROGRAM

s1=input("Enter the first string:")

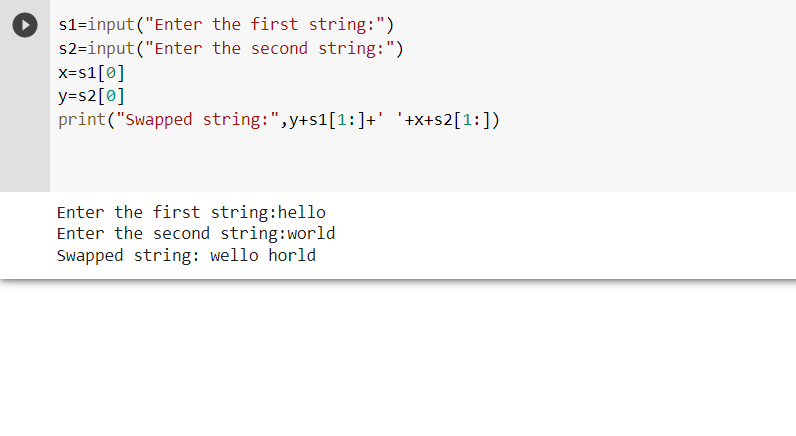
s2=input("Enter the second string:")

x=s1[0]

y=s2[0]

print("Swapped string:",y+s1[1:]+' '+x+s2[1:])

## OUTPUT



**PROGRAM 16**

**AIM**

Sort dictionary in ascending and descending order.

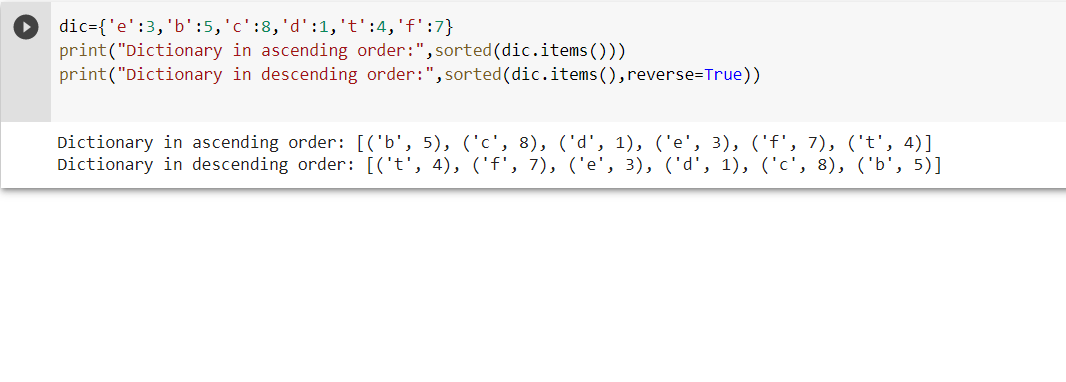
## PROGRAM

dic={'e':3,'b':5,'c':8,'d':1,'t':4,'f':7}

print("Dictionary in ascending order:",sorted(dic.items()))

print("Dictionary in descending order:",sorted(dic.items(),reverse=True))

## OUTPUT



**PROGRAM 17**

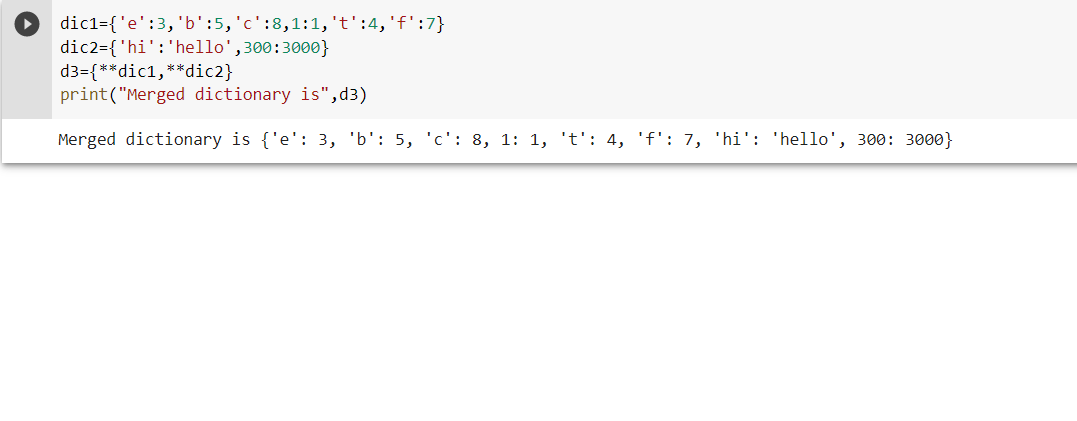
**AIM**

Merge two dictionaries.

## PROGRAM

dic1={'e':3,'b':5,'c':8,1:1,'t':4,'f':7} dic2={'hi':'hello',300:3000} d3={\*\*dic1,\*\*dic2} print("Merged dictionary is",d3)

## OUTPUT



**PROGRAM 18**

**AIM**

Find GCD of 2 numbers.

## PROGRAM

x=int(input("Enter 1st number:"))

y=int(input("Enter 2nd number:"))

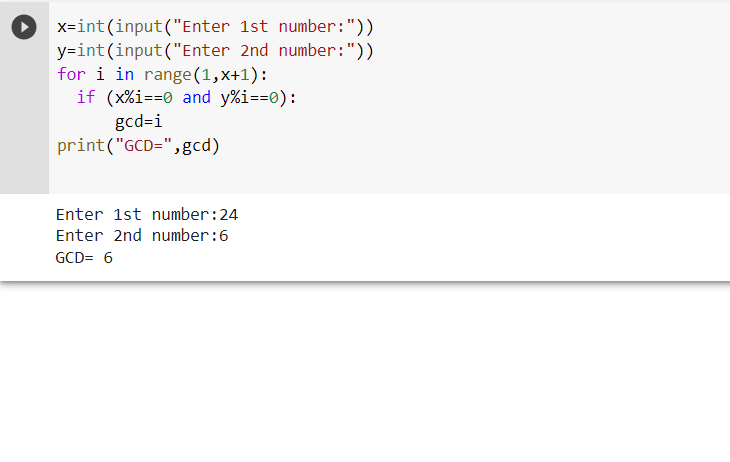
for i in range(1,x+1):

if (x%i==0 and y%i==0):

gcd=i

print("GCD=",gcd)

## OUTPUT



**PROGRAM 19**

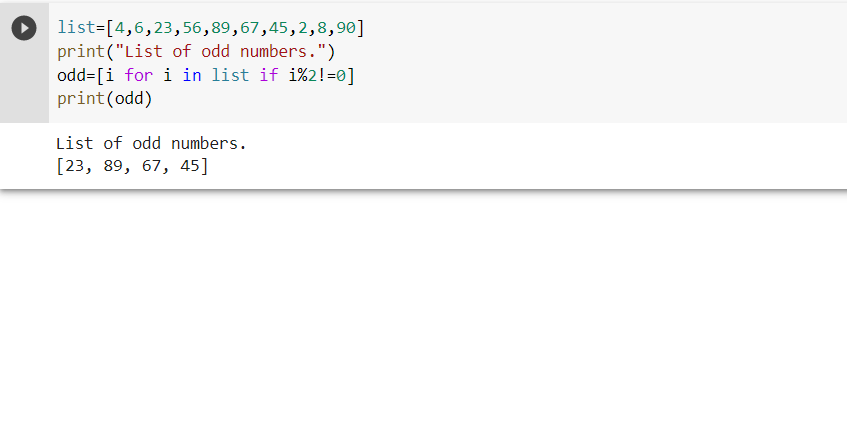
**AIM**

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

## PROGRAM

list=[4,6,23,56,89,67,45,2,8,90] print("List of odd numbers.") odd=[i for i in list if i%2!=0] print(odd)

## OUTPUT



**Course Outcome 2 Programs**

**PROGRAM 20**

**AIM**

Program to find the factorial of a number.

## PROGRAM

n=input("enter number")

num=int(n)

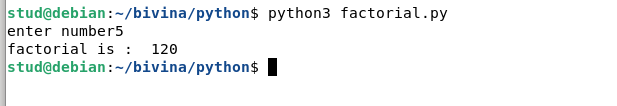
fact=1

for i in range (1, num+1) :

fact=fact\*i

print("factorial is : " , fact)

## OUTPUT



**PROGRAM 21**

**AIM**

Generate Fibonacci series of N terms.

## PROGRAM

n=input("enter range")

a=0

b=1

num=int(n)

print(a,"\n",b)

for i in range (2,num):

c=a+b

print(c)

a=b

b=c

## OUTPUT

**PROGRAM 22**

**AIM**

Find the sum of all items in a list.

## PROGRAM

n=int(input("Enter the limit:"))

print("Enter the elements:")

a=[]

sum=0

for i in range(0,n):

x=int(input())

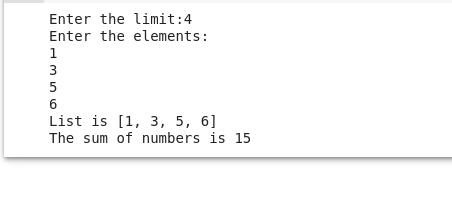
a.append(x)

sum=sum+x

print("List is",a)

print("The sum of numbers is",sum)

## OUTPUT



**PROGRAM 23**

## AIM

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

## PROGRAM

limit1=1000

limit2=9999

list1=[]

for i in range(limit1,limit2):

j=i

digit=[]

while(i!=0):

digit.append(i%10)

i=int(i/10)

count=0

for n in digit:

if n%2==0:

count=count+1

if count==4:

for k in range(31,100):

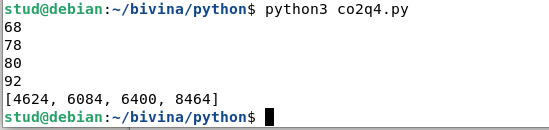
if((k\*\*2)==j):

list1.append(j)

print(k)

print(list1)

## OUTPUT



**PROGRAM 24**

**AIM**

Display the given pyramid with step number accepted from user.

Eg: N=4

1

1. 4
2. 6 9
3. 8 12 16

## PROGRAM

n=int(input('enter the number of steps'))

for i in range(1,n+1):

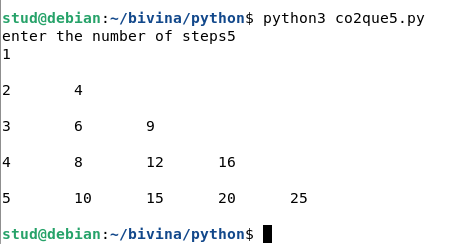
for j in range(1,i+1):

s=i\*j

print(s,'\t',end="")

print("\n")

## OUTPUT



**PROGRAM 25**

**AIM**

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

## PROGRAM

str=input("Enter a string:")  
fnd=input("Enter character:")  
cnt=0  
fnd=fnd.lower()  
str=str.lower()  
for i in str:  
 if i==fnd:  
  cnt=cnt+1  
print("Freq:->",cnt)

## OUTPUT

**PROGRAM 26**

**AIM**

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

‘ly’.

## PROGRAM

a=input("Enter a word\n")

l=len(a)

ll=a[l-3:l]

if(ll=="ing"):

s=a+"ly"

else:

s=a+"ing"

print (s)

## OUTPUT

**PROGRAM 27**

**AIM**

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

## PROGRAM

list=[]

length=[]

print("enter 5 words")

for i in range (5):

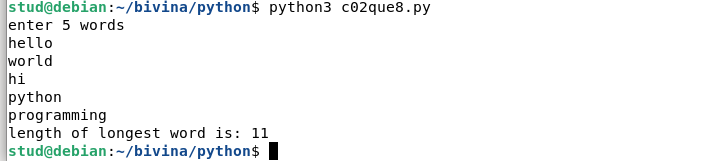
str=input()

list.append(str)

for j in list:

length.append(len(j))

print("length of longest word is:",max(length))

**OUTPUT**

**PROGRAM 28**

**AIM**

Construct following pattern using nested loop

\*

* \*
* \* \*
* \* \* \*
* \* \* \* \*
* \* \* \*
* \* \*
* \*

\*

## PROGRAM

for i in range(1,6):

for j in range(1,i+1):

print("\*",end="")

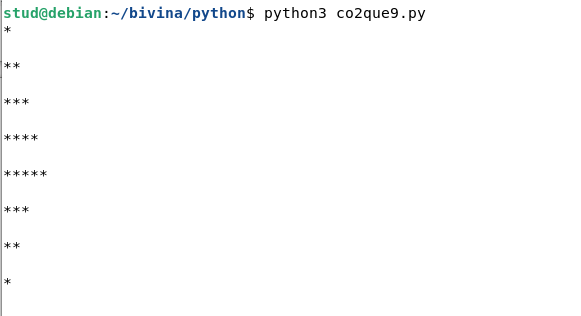
print("\n")

for i in range(5,0,-1):

for j in range(1,i-1):

print("\*",end="")

print("\n")



**PROGRAM 29**

**AIM**

Generate all factors of a number.

## PROGRAM

def print\_factors(x):

print("the factors of",x,"are:\n")

for i in range(2,int(x/2)+1):

if x%i==0:

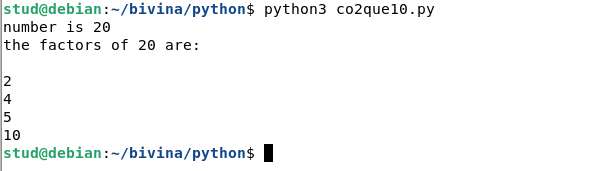
print(i)

num=20

print("number is",num)

print\_factors(num)

## OUTPUT



# Course Outcome 3 Programs

## PROGRAM 30

## 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)

## PROGRAM

# Graphice\circle.py

from math import pi

def area\_circle(radius): return pi\*radius\*radius

def perimeter\_circle(radius): return 2\*pi\*radius

# Graphics\rectangle.py

def area\_rec(length,width): return length\*width

def perimeter\_rec(length,width): return 2\*(length+width)

# Graphics\tdgraphics\cuboid.py

def area\_cuboid(l,b,h): return 2\*(l\*h + b\*h + l\*b)

def volume\_cuboid(l,b,h): return l\*b\*h

# Graphics\tdgraphics\sphere.py

from math import pi

def area\_sphere(radius): return 4\*(pi\*radius\*radius)

def perimeter\_sphere(radius): return 2\*pi\*radius

# Graphics.py (driver code)

import Graphics

from Graphics import circle,rectangle

from Graphics.tdgraphics import cuboid,sphere from Graphics.circle import \*

print("Area of a circle with radius 10 is : ",circle.area\_circle(10)) print("Permeter of a circle with radius 10 is ",circle.perimeter\_circle(10)) print("\n")

print("Area of a Rectangle with length and width 10 is : ",rectangle.area\_rec(10,10)) print("Permeter of a Rectangle with length and width 10 is : ",rectangle.perimeter\_rec(10,10)) print("\n")

print("Area of a cuboid with length,width,height 10 is : ",cuboid.area\_cuboid(10,10,10)) print("Volume of a cuboid with length,width,height 10 is : ",cuboid.volume\_cuboid(10,10,10)) print("\n")

print("Area of a spere with radius 10 is : ",sphere.area\_sphere(10)) print("Permeter of a spere with radius 10 is ",sphere.perimeter\_sphere(10))

## OUTPUT



# Course Outcome 4 Programs

## PROGRAM 31

## AIM

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

## PROGRAM

class Rectangle:

def create (self,l,b):

self.l=l

self.b=b

def area(self):

a=(self.l\*self.b)

print("area=",a)

return a

def perimeter(self):

b= 2\*(self.l+self.b)

print("perimeter=",b)

return b

l=int(input("enter the length"))

b=int(input("enter the breadth"))

r1=Rectangle()

r1.create(l,b)

l=int(input("enter the length"))

b=int(input("enter the breadth"))

r2=Rectangle()

r2.create(l,b)

x=r1.area()

y=r2.area()

z=r1.perimeter()

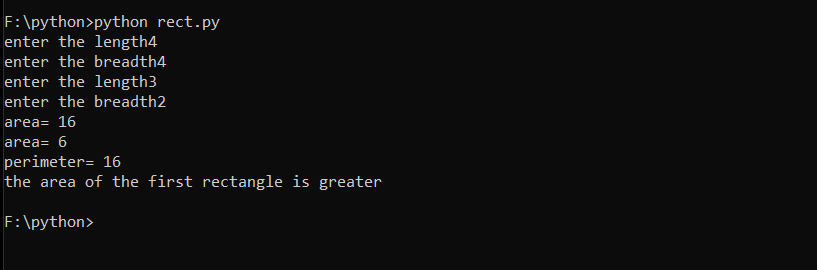
if(x>y):

print("the area of the first rectangle is greater")

else:

print("the area of the second rectangle is greater")

## OUTPUT



## PROGRAM 32

## 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.

## PROGRAM

class Bank\_account:

def \_\_init\_\_(self,ano,name,type,balance):

self.account\_number=ano self.name=name self.type\_of\_account=type self.balance=balance def deposite(self,amount):

self.balance=self.balance+amount def withdraw(self,amount): if(amount>self.balance):

print("Insufficient Balance!!!") else:

self.balance=self.balance-amount account1=Bank\_account(101,"ABC","Savings",50000) account2=Bank\_account(102,"XYZ","Savings",80000)

#Depositing Rs 10000 to account1

print("Before Depositing-Balance of account1=",account1.balance) account1.deposite(10000)

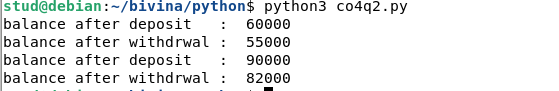
print("After Depositing-Balance of account1=",account1.balance)

#Withdrawing Rs 76000 from account2

print("Before Withdrawal-Balance of account2=",account2.balance) account2.withdraw(76000)

print("After Withdrawal-Balance of account2=",account2.balance)

## OUTPUT



## PROGRAM 33

## AIM

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

## PROGRAM

class Rectangle:

def create (self,l,b):

self.l=l

self.b=b

def area(self):

a=(self.l\*self.b)

print("area=",a)

return a

def perimeter(self):

b= 2\*(self.l+self.b)

print("perimeter=",b)

return b

def \_\_lt\_\_(self,rr):

if(self.b\*self.l>rr.b\*rr.l):

print("the area of the first rectangle is greater")

return True

else:

print("the area of the second rectangle is greater")

return False

l=int(input("enter the length"))

b=int(input("enter the breadth"))

r1=Rectangle()

r1.create(l,b)

l=int(input("enter the length"))

b=int(input("enter the breadth"))

r2=Rectangle()

r2.create(l,b)

x=r1.area()

y=r2.area()

z=r1.perimeter()

r1 > r2

## OUTPUT

## PROGRAM 34

## AIM

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

## PROGRAM

class Time:

def \_\_init\_\_(self,hr,min,sec):

self.\_\_hr=hr

self.\_\_min=min

self.\_\_sec=sec

def \_\_add\_\_(self,t2):

self.\_\_hr=self.\_\_hr+t2.\_\_hr

self.\_\_min=self.\_\_min+t2.\_\_min

self.\_\_sec=self.\_\_sec+t2.\_\_sec

if (self.\_\_sec>60):

self.\_\_sec-=60

self.\_\_min+=1

if(self.\_\_min>60):

self.\_\_min-=60

self.\_\_hr+=1

if(self.\_\_hr>12):

self.\_\_hr-=12

print("Total time is",self.\_\_hr,":",self.\_\_min,":",self.\_\_sec)

t1=Time(3,55,56)

t2=Time(2,24,8)

T1+t2

## OUTPUT



## PROGRAM 35

## 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.

## PROGRAM

class Publisher(object):

def \_\_init\_\_(self,name):

self.name=name

def display1(self):

print(self.title)

print(self.author)

class Book(Publisher):

def \_\_init\_\_(self,name,title,author):

super().\_\_init\_\_(name)

self.title=title

self.author=author

def display2(self):

super().display1()

print(self.title)

print(self.author)

class Python(Book):

def \_\_init\_\_(self,name,title,author,price,no\_of\_pages):

super().\_\_init\_\_(name,title,author)

self.price=price

self.no\_of\_pages=no\_of\_pages

def display3(self):

super().display2()

print(self.price)

print(self.no\_of\_pages)

p=Python("HHH Publications","The Python","Bivina",140,600)

p.display3()

## OUTPUT



# Course Outcome 5 Programs

**PROGRAM 36**

**AIM**

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

## PROGRAM

fp=open("text\_file.txt",'r')

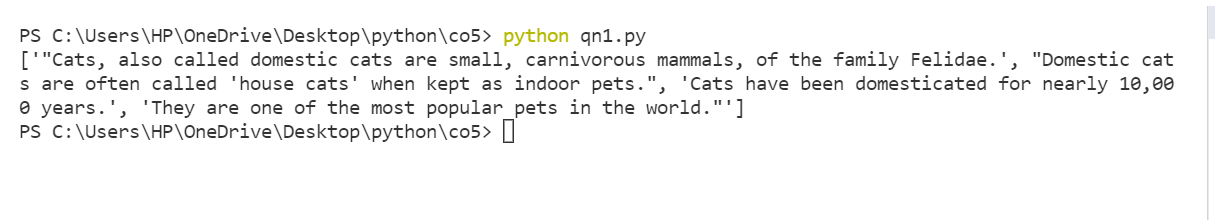
lines=[]

for line in fp:

lines.append(line.strip())

print(lines)

## OUTPUT



## PROGRAM 37

## AIM

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

## PROGRAM

import csv

with open('people.csv', 'r') as file:

reader = csv.reader(file)

for row in reader:

print(row)

## OUTPUT

## 