# DEPARTMENT OF COMPUTER APPLICATION TKM COLLEGE OF ENGINEERING

**KOLLAM – 691005**



**20MCA131 – PROGRAMMING LAB**

PRACTICAL RECORD BOOK

First Semester MCA 2021-2022

**Submitted by:**

NAME : KANNAN R R

REG NO : TKM21MCA-2026

**DEPARTMENT OF COMPUTER APPLICATION TKM COLLEGE OF ENGINEERING**

**KOLLAM – 691005**



**Certificate**

This is a bonafide record of the work done by KANNAN R R(TKM21MCA-2026) in the First Semester in Programming Lab Course(20MCA131) towards the partial fulfillment of the degree of Master of Computer Applications during the academic year 2021-2022

Staff Member in-charge Examiner

……………………….. ……………………….

**LABCYCLE 1 QUESTION 1**

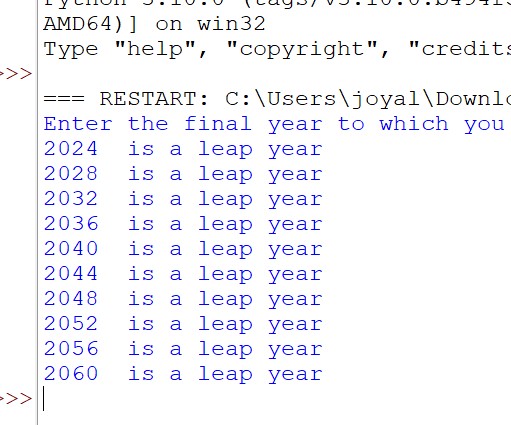
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p1.py** | import datetime  year=int(input("Enter the final year to which you want to display leap years : "))  tyear=datetime.datetime.now().year  for years in range(tyear,year+1):  if years % 4 == 0 or years % 400 == 0:  print(years," is a leap year") |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 2**

**AIM:**

List comprehensions:

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

(b) Square of N numbers

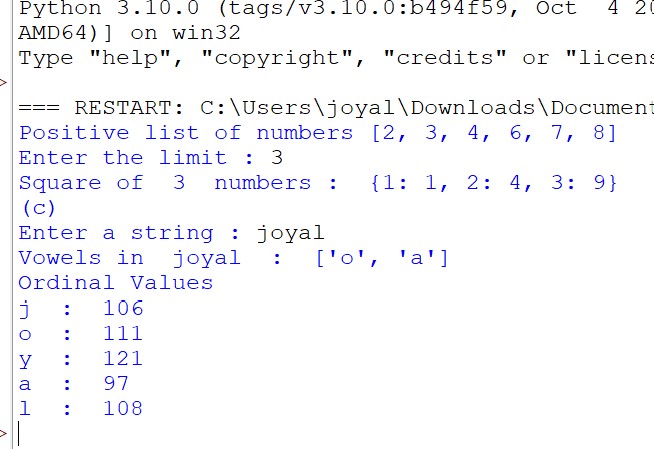
(c) Form a list of vowels selected from a given word

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p2.py** | l=[-1,2,3,4,-4,-5,6,7,8,-9,-10]  m=[x for x in l if x > 0 ]  print("Positive list of numbers",m)  c=int(input("Enter the limit : "))  n={x:x\*\*2 for x in range(1,c+1)}  print("Square of ",c," numbers : ",n)  print("(c)")  v=['a','e','i','o','u']  s=input("Enter a string : ")  o=[x for x in s if x in v]  print("Vowels in ",s," : ",o)  print("Ordinal Values")  for x in s:  print(x," : ",ord(x)) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 3**

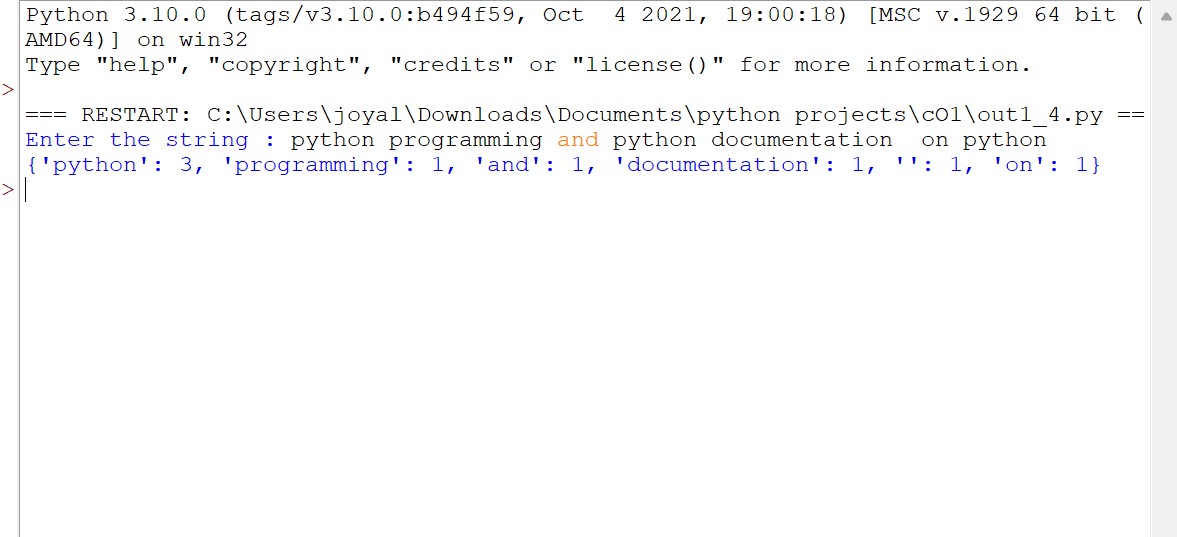
**AIM:**

Count the occurrences of each word in a line of text

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p3.py** | s1=input("Enter the string : ")  s2=s1.split( )  d={}  for i in s2:  if i in d:  d[i.lower()]+=1  else:  d[i.lower()]=1  print(d) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 4**

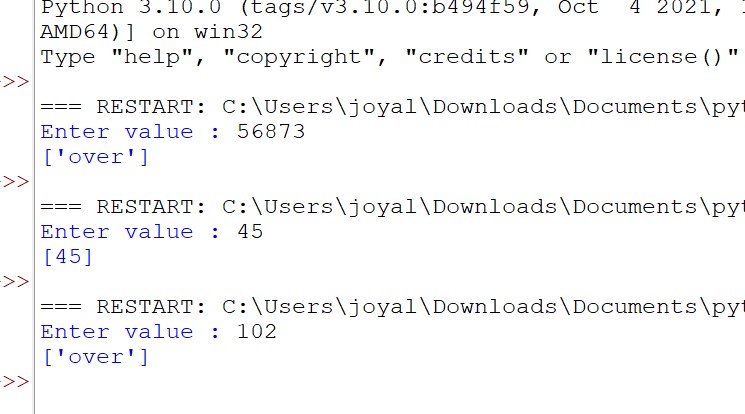
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p4.py** | s=input("Enter value : ")  s=s.split(",")  l=[]  c=0  for i in s:  l.append(int(i))  for i in l:  if i>100:  l[c]='over'  c+=1  print(l) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 5**

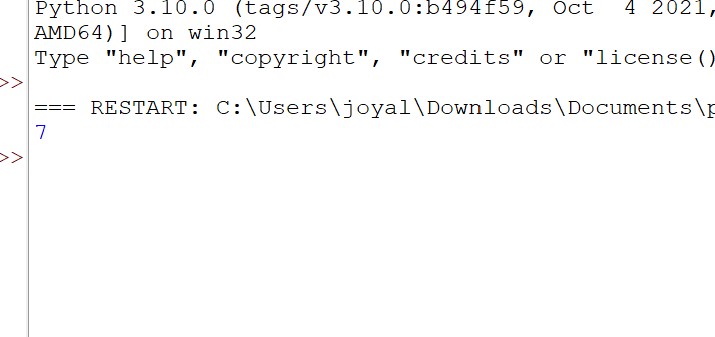
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p5.py** | l=['amal','akash','bharath','divin','rakesh']  c=0  for i in l:  if 'a' in i:  c+=i.count('a')  print(c) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 6**

**AIM:**

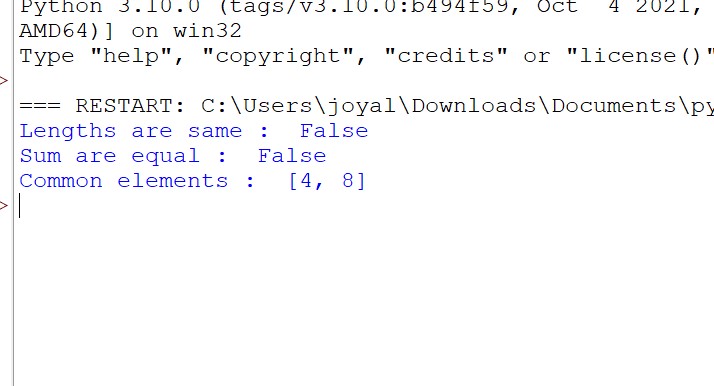
Enter 2 lists of integers. Check (a) Whether list are of same length (b) whether list sums

to same value (c) whether any value occur in both

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p6.py** | l1=[2,4,1,3,5,8,9]  l2=[4,6,0,6,8]  s=len(l1)==len(l2)  p=sum(l1)==sum(l2)  print("Lengths are same : ",s)  print("Sum are equal : ",p)  m=[ i for i in l1 if i in l2]  print("Common elements : ",m) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 7**

**AIM:**

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

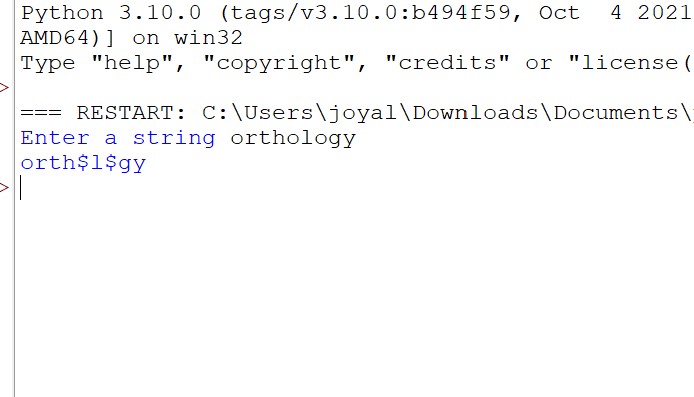
‘$’, except first character.

[eg: onion -> oni$n] **:**

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p7.py** | s1=input("Enter a string ")  s2=s1[0]  s3=s1[1:]  s4=s3.replace(s2,'$')  replaced=s2+s4  print(replaced) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 8**

**AIM:**

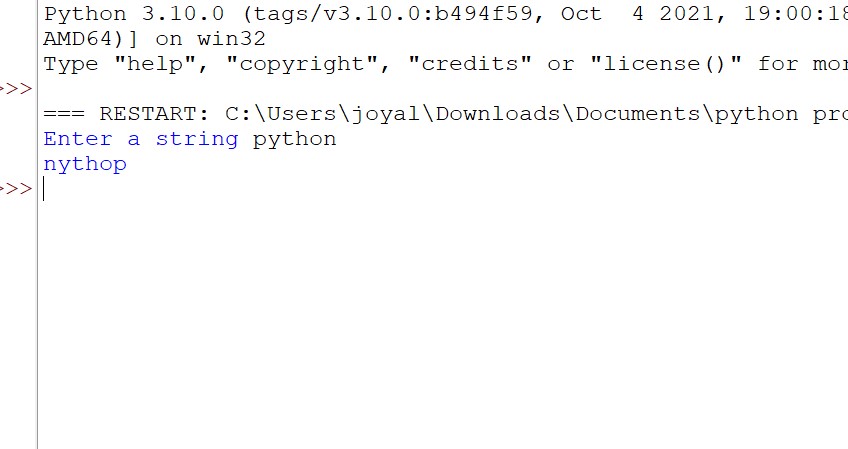
Create a string from given string where first and last characters exchanged. [eg: python -

> nythop]

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p8.py** | s1=input("Enter a string ")  s2=s1[0]  s3=s1[-1]  s4=s1.replace(s2,s3)  s5=s4[0]  s6=s4[1:]  s7=s6.replace(s3,s2)  s8=s5+s7  print(s8) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 9**

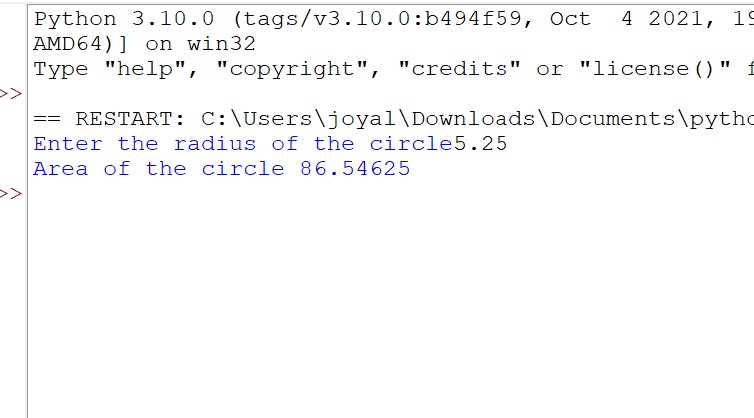
**AIM:**

Accept the radius from user and find area of circle

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p9.py** | a=float(input("Enter the radius of the circle"))  area=3.14\*a\*a  print("Area of the circle",area) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 10**

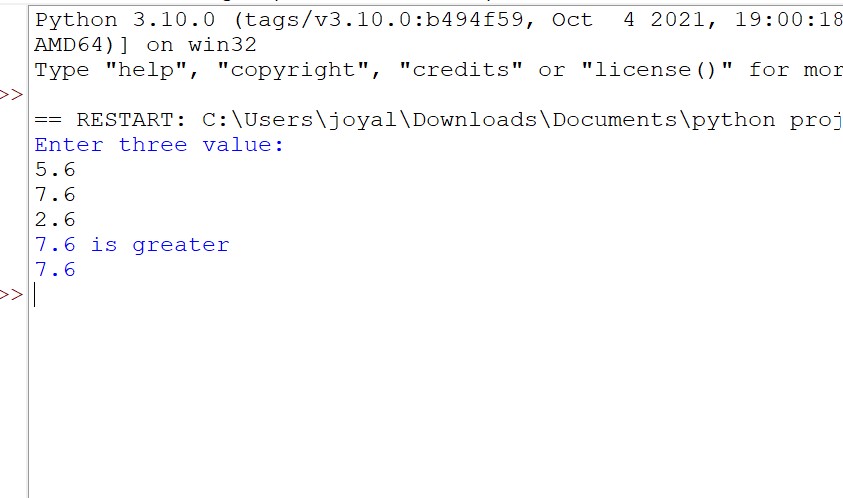
**AIM:**

Find biggest of 3 numbers entered

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p10.py** | print("Enter three value:")  a,b,c=input(),input(),input()  if((a>b) and (a>c)):  print(a, "is greater")  elif ((b>a) and (b>c)):  print(b, "is greater")  else:  print(c, "is greater")  print(max(a,b,c)) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 11**

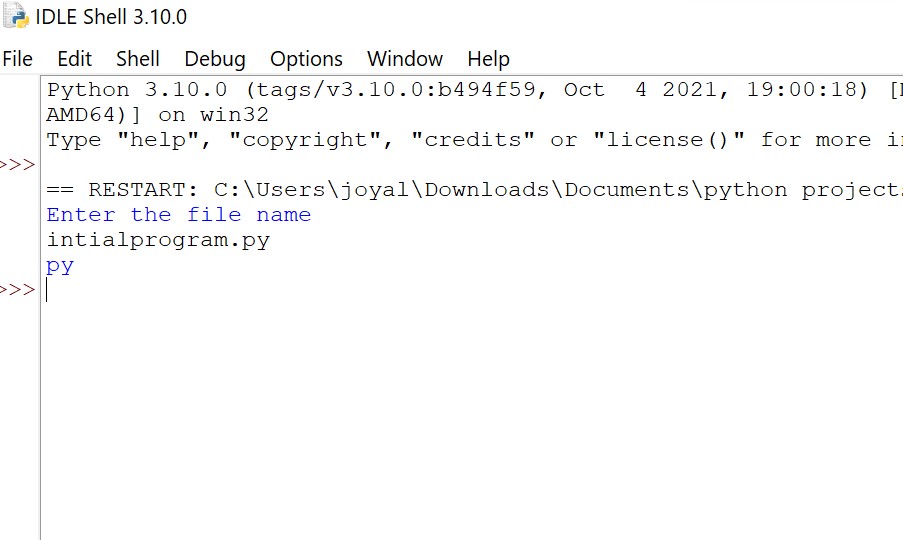
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p11.py** | filename=input("Enter the file name\n")  l=filename.split(".")  print(l[-1]) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 12**

**AIM:**

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

first and last colors.

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p12.py** | s=input("Enter comma separated colors : ")  s=s.split(",")  l=[]  c=0  for i in s:  l.append(i)  print("First color : ",l[0]," Last Color : ",l[-1]) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 13**

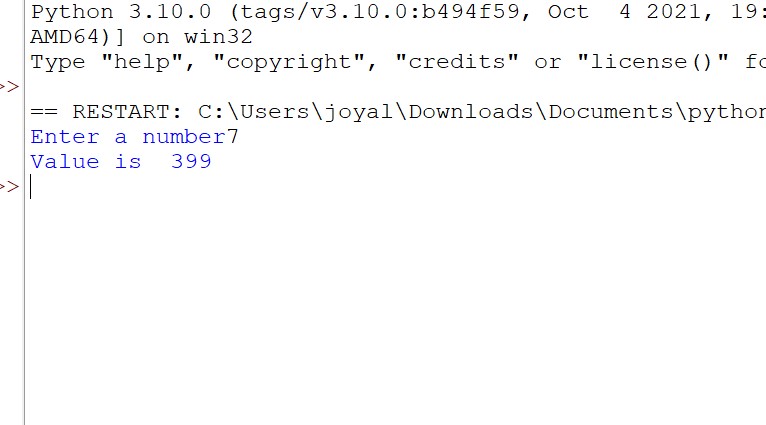
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p13.py** | n=int(input("Enter a number"))  sum=n+(n\*n)+(n\*n\*n)  print("Value is ",sum) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 14**

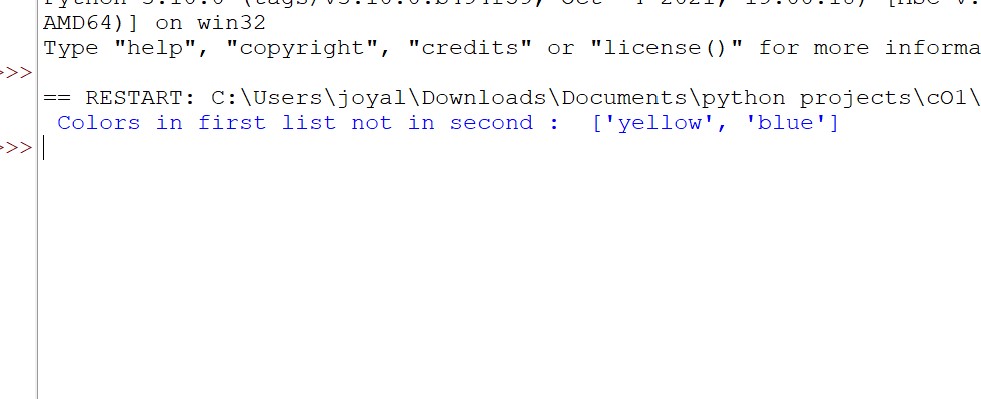
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p14.py** | c1=['yellow','green','blue','white']  c2=['white','green','violet','black']  m=[ i for i in c1 if i not in c2]  print(" Colors in first list not in second : ",m) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 15**

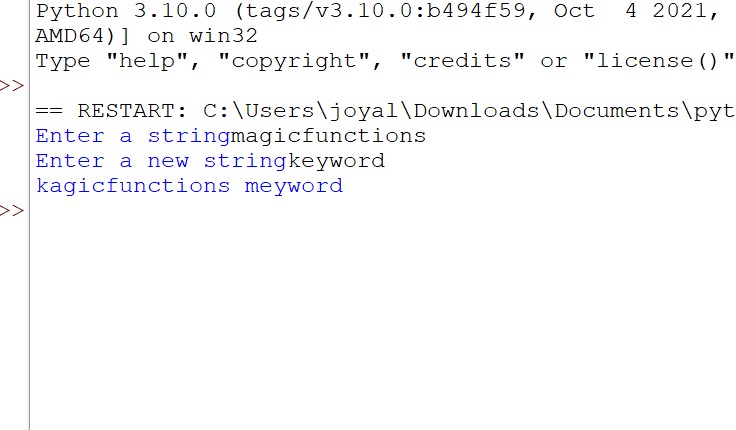
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p15.py** | s1=input("Enter a string")  s2=input("Enter a new string")  s3=s1[0]  s4=s2[0]  s5=s4+s1[1:]+" "+s3+s2[1:]  print(s5) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 16**

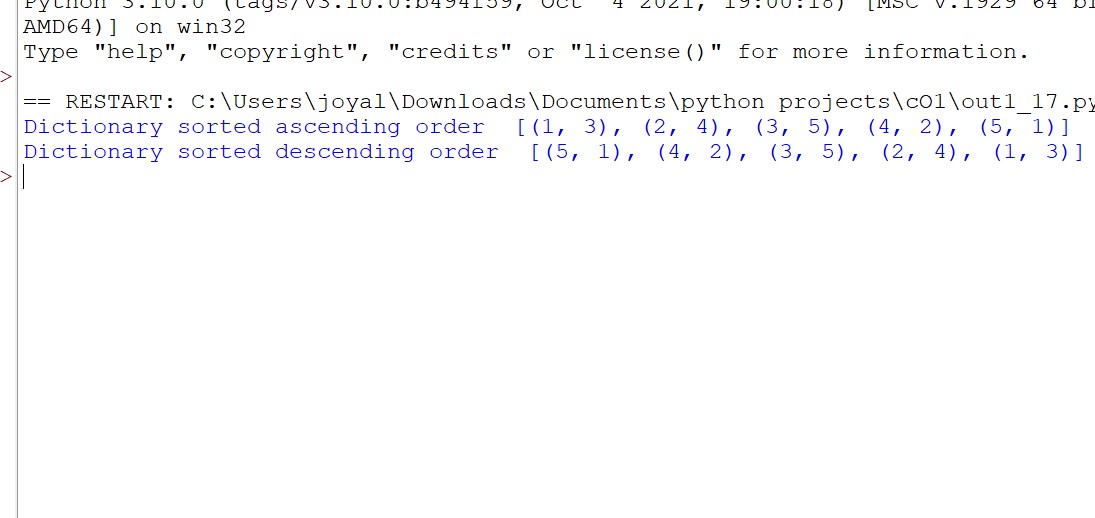
**AIM:**

Sort dictionary in ascending and descending order.

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p16.py** | import operator  d={3:5,1:3,4:2,5:1,2:4}  sort\_as=(sorted(d.items(), key=operator.itemgetter(0)))  sort\_dec=(sorted(d.items(), key=operator.itemgetter(0), reverse=True))  print("Dictionary sorted ascending order ",sort\_as)  print("Dictionary sorted descending order ",sort\_dec) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 17**

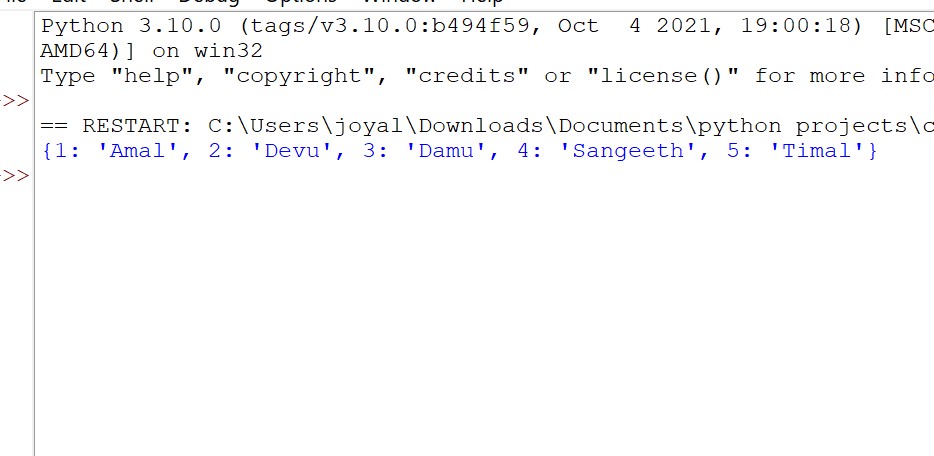
**AIM:**

Merge two dictionaries

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p17.py** | d1={1:'Amal',2:'Devu',3:'Damu'}  d2={4:'Sangeeth',5:'Timal'}  d1.update(d2)  print(d1) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 18**

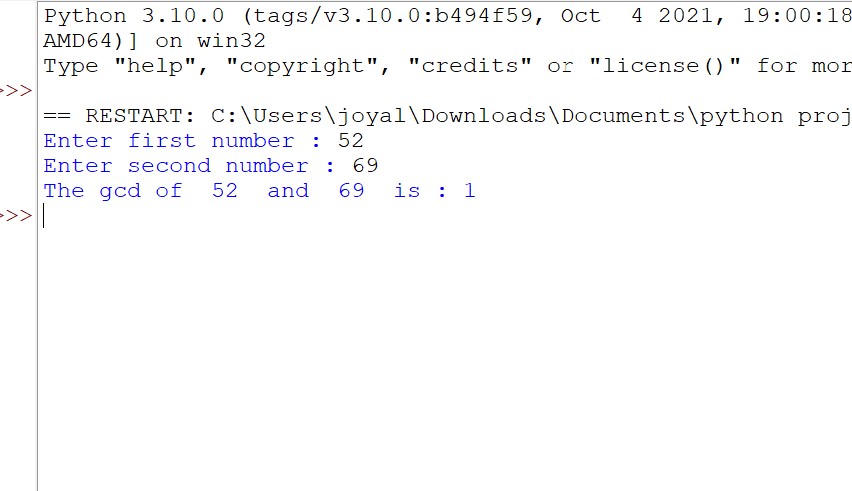
**AIM:**

Find gcd of 2 numbers

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p18.py** | import math    s=int(input("Enter first number : "))  t=int(input("Enter second number : "))  print("The gcd of ",s," and ",t," is : ", end="")  print(math.gcd(s, t)) |

**OUTPUT:**

****

**LABCYCLE 1 QUESTION 19**

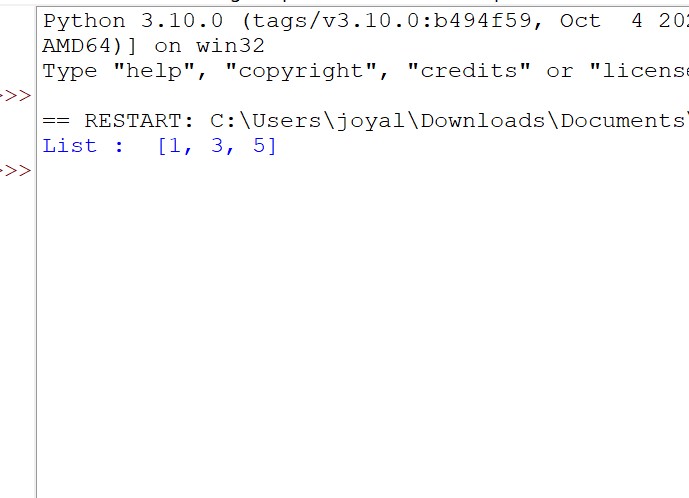
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out1p19.py** | l=[1,2,3,4,5]  odd=[]  for i in l:  if i%2!=0:  odd.append(i)  print("List : ",odd) |

**OUTPUT:**

****

**LABCYCLE 2 QUESTION 1**

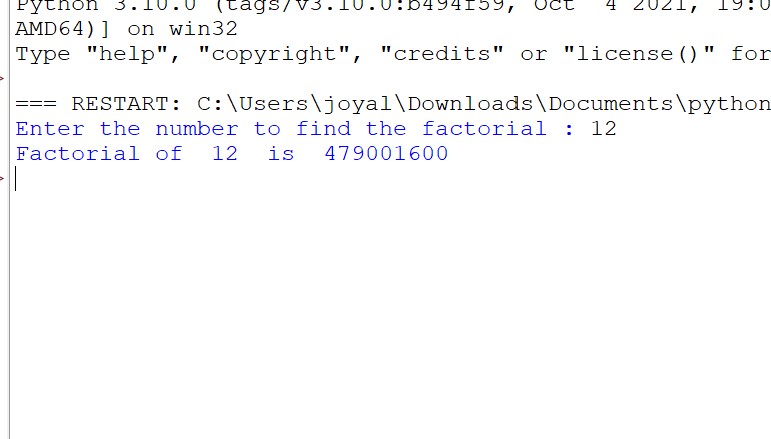
**AIM:**

Program to find the factorial of a number

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p1.py** | fact=1  n=int(input("Enter the number to find the factorial : "))  for i in range(1,n+1):  fact=fact\*i  print("Factorial of ",n," is ",fact) |

**OUTPUT:**

****

**LABCYCLE 2 QUESTION 2**

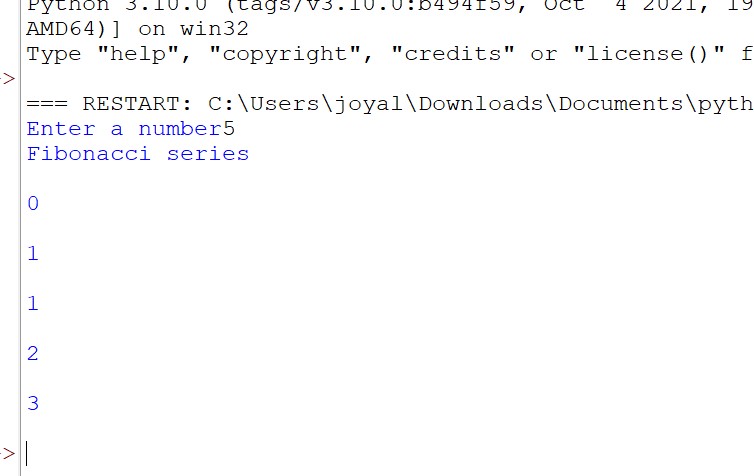
**AIM:**

Generate Fibonacci series of N terms

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p2.py** | num=int(input("Enter a number"))  a=0  b=1  sum=0  count=1  print("Fibonacci series\n")  while count<=num:  print(sum,"\n")  a=b  b=sum  sum=a+b  count+=1 |

**OUTPUT:**

****

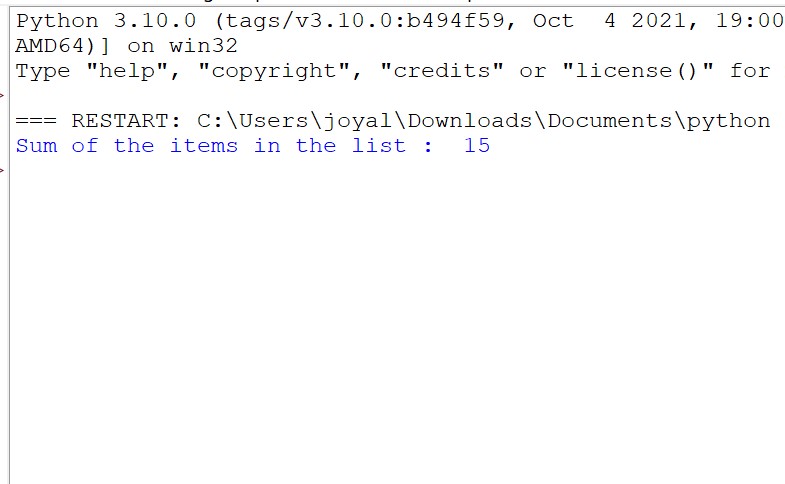
**LABCYCLE 2 QUESTION 3**

**AIM:**

Find the sum of all items in a list

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p3.py** | l=[1,2,3,4,5]  sum=0  for i in l:  sum=sum+i  print("Sum of the items in the list : ",sum) |

****

**LABCYCLE 2 QUESTION 4**

**AIM:**

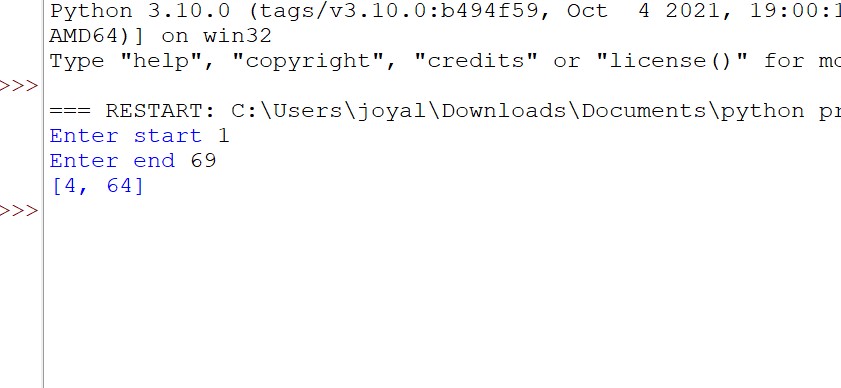
Generate a list of four digit numbers in a given range with all their digits even and the

number is a perfect square.

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p4.py** | import math  lists =[]  start=int(input("Enter start "))  end=int(input("Enter end "))  for a in range(start,end+1):  for b in str(a):  if int(b) % 2 != 0:  break  else:  root=math.sqrt(a)  if root % 1 == 0:  lists.append(a)  print(lists) |

**OUTPUT:**

****

**LABCYCLE 2 QUESTION 5**

**AIM:**

Display the given pyramid with step number accepted from user.

Eg: N=4

1

2 4

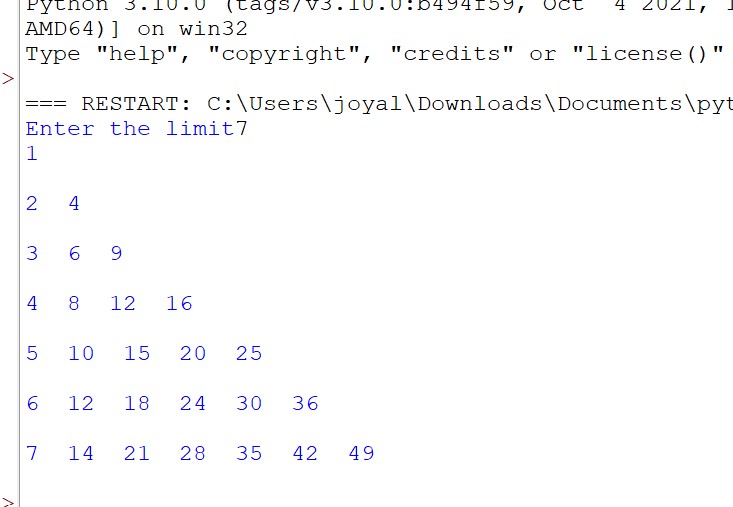
3 6 9

4 8 12 16

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p5.py** | num=int(input("Enter the limit"))  for i in range(1,num+1):  for j in range(1,i+1):  print(i\*j," ",end='')  print("\n") |

**OUTPUT:**

****

**LABCYCLE 2 QUESTION 6**

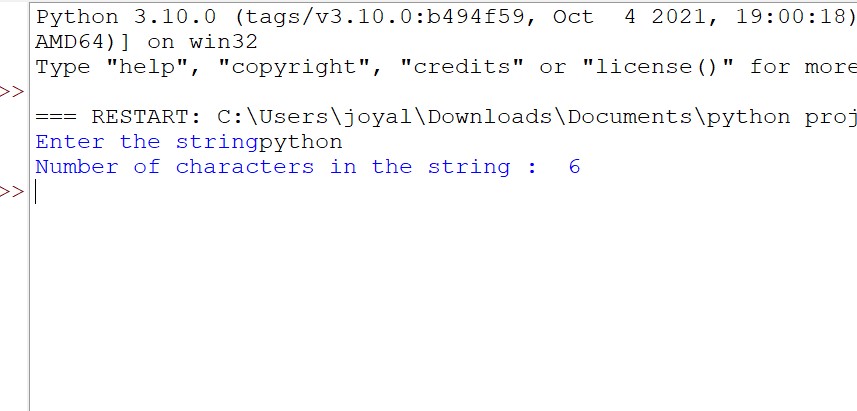
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p6.py** | char=input("Enter the string")  count=len(char)  print("Number of characters in the string : ",count) |

**OUTPUT:**



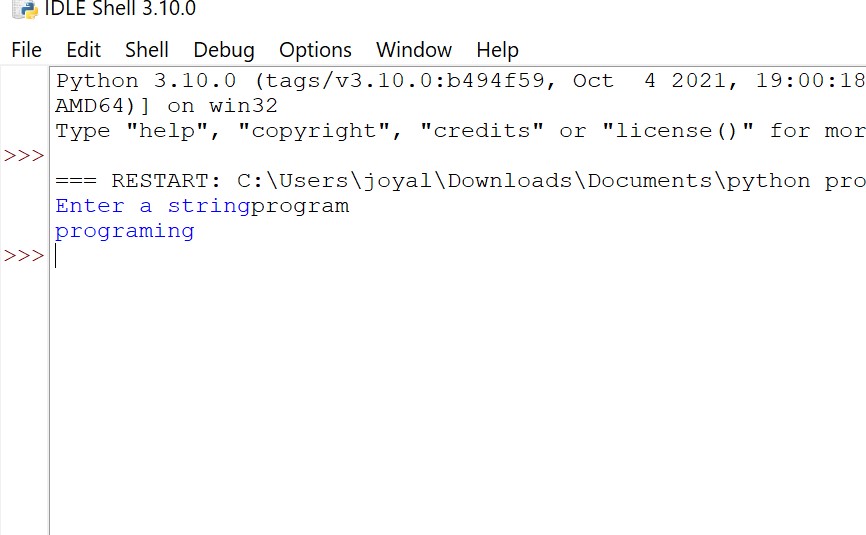
**LABCYCLE 2 QUESTION 7**

**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p7.py** | st=input("Enter a string")  if(st[-3:]=='ing'):  st=st[:]+'ly'  else:  st=st[:]+'ing'  print(st) |



**LABCYCLE 2 QUESTION 8**

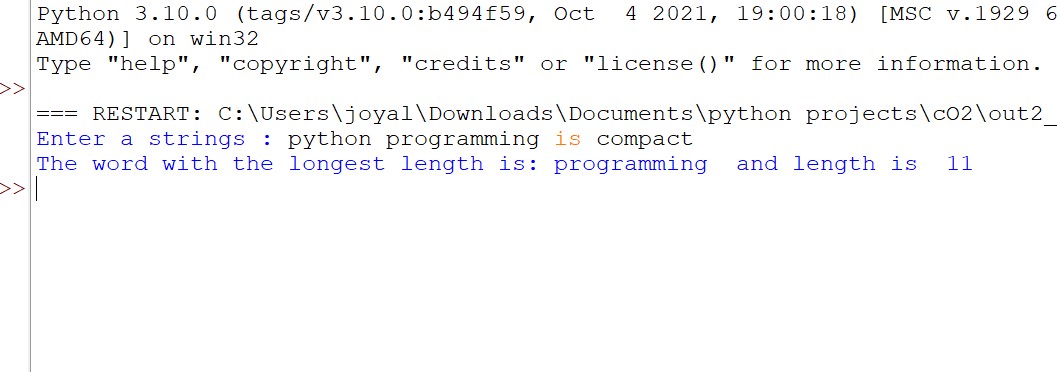
**AIM:**

Accept a list of words and return length of longest word

**PROGRAM CODE:**

|  |  |
| --- | --- |
| Out2p8**.py** | s=input("Enter a strings : ")  s=s.split(" ")  a=[]  for i in s:  a.append(i)  max = len(a[0])  temp = a[0]  for i in a:  if(len(i) > max):  max = len(i)  temp = i  print("The word with the longest length is:", temp," and length is ", max) |

**OUTPUT:**



**LABCYCLE 2 QUESTION 9**

**AIM:**

Construct following pattern using nested loop

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

\* \* \* \*

\* \* \*

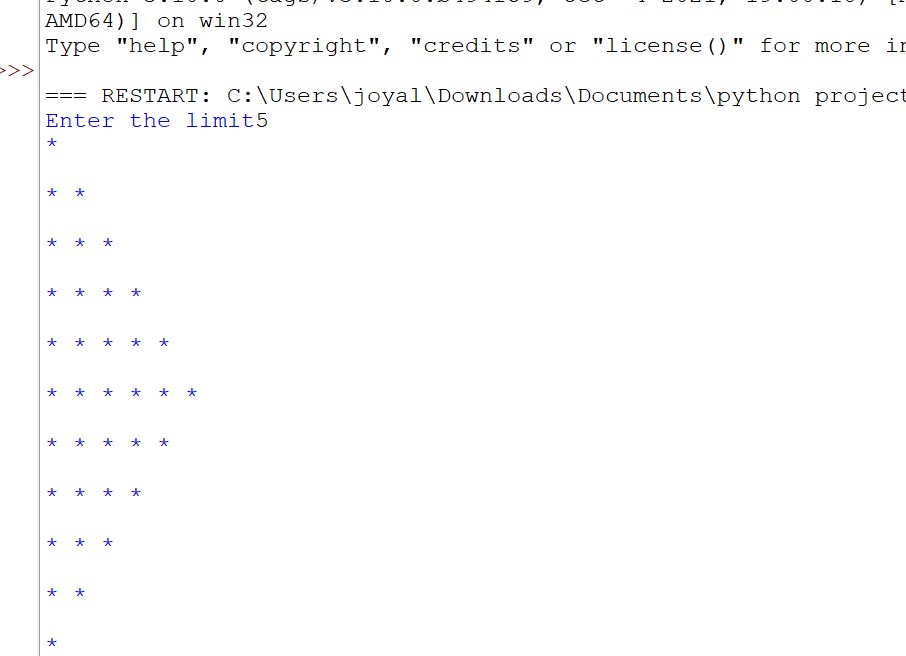
\* \*

\*

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p9.py** | num=int(input("Enter the limit"))  for i in range(1,num+1):  for j in range(1,i+1):  print("\* ",end='')  print("\n")  for i in range(num+1,0,-1):  for j in range(1,i+1):  print("\* ",end='')  print("\n") |

**OUTPUT:**

****

**LABCYCLE 2 QUESTION 10**

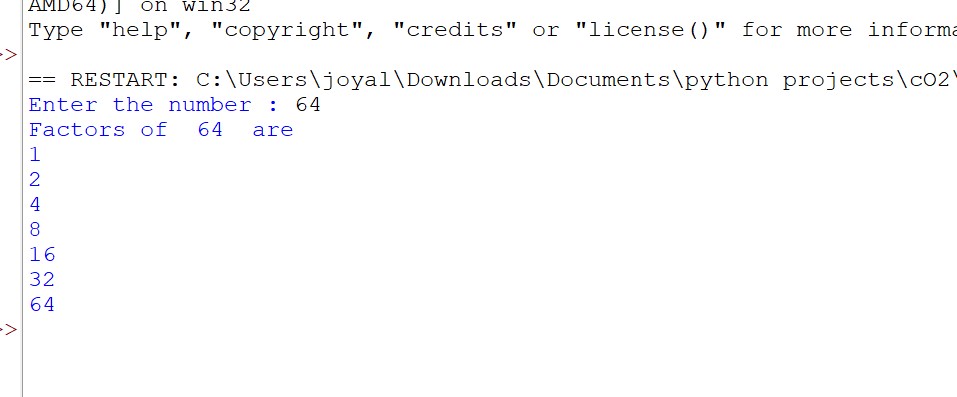
**AIM:**

Generate all factors of a number.

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p10.py** | num=int(input("Enter the number : "))  print("Factors of ",num," are")  for i in range(1,num+1):  if num % i ==0:  print(i) |

**OUTPUT:**

****

**LABCYCLE 2 QUESTION 11**

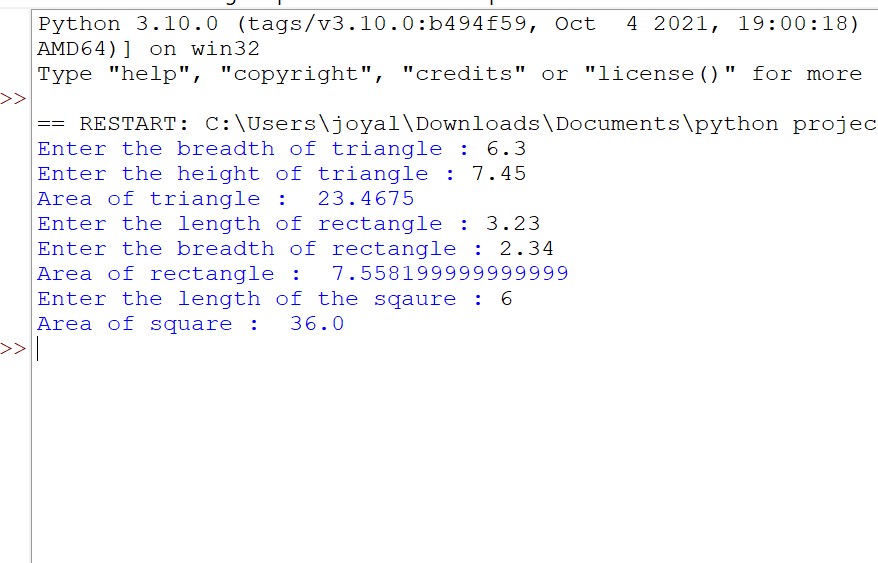
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out2p11.py** | import math  t\_area= lambda b,h : 1/2\*(b\*h)  r\_area= lambda l,b : l\*b  s\_area= lambda a : a\*a  s=float(input("Enter the breadth of triangle : "))  t=float(input("Enter the height of triangle : "))  print("Area of triangle : ",t\_area(s,t))  u=float(input("Enter the length of rectangle : "))  v=float(input("Enter the breadth of rectangle : "))  print("Area of rectangle : ",r\_area(u,v))  w=float(input("Enter the length of the sqaure : "))  print("Area of square : ",s\_area(w)) |

**OUTPUT:**

****

**LABCYCLE 3 QUESTION 1**

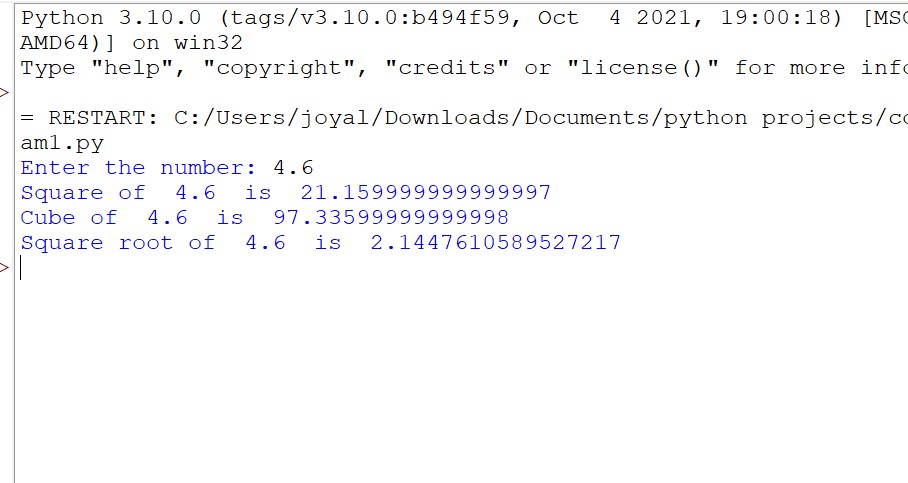
**AIM:**

Work with built-in packages

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out3p1.py** | import math  n=int(input("Enter the number: "))  print("Square of ",n," is ",pow(n,2))  print("Cube of ",n," is ",pow(n,3))  print("Square root of ",n," is ",math.sqrt(n)) |

**OUTPUT:**

****

**LABCYCLE 3 QUESTION 2**

**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 CODE:**

|  |  |
| --- | --- |
| **Out3p2.py** | from graphics.rectangle import \*  from graphics.\_3D\_graphics.cuboid import \*  from graphics.\_3D\_graphics.sphere import \*  from graphics.circle import \*  print("\*\*\*\*\*\*\*\*Rectangle\*\*\*\*\*\*\*\*\*\*")  l=float(input("Enter the length: "))  b=float(input("Enter the breadth: "))  arearect(l,b)  perirect(l,b)  h=float(input("Enter the height of cuboid: "))  cuboidarea(l,b,h)  cuboidperi(l,b,h)  print("\*\*\*\*\*\*\*\*Circle\*\*\*\*\*\*\*\*")  r=float(input("Enter the radius: "))  areac(r)  circumc(r)  sphere(r) |

GRAPHICS MODULE

|  |  |
| --- | --- |
| **Rectangle.py** | def arearect(l,b):  a=l\*b  print("Area of rectangle: ",a)  def perirect(l,b):  p=2\*(l+b)  print("Perimeter of rectangle: ",p) |

|  |  |
| --- | --- |
| **Circle.py** | def areac(r):  a=3.14\*r\*r  print("Area of circle: ",a)  def circumc(r):  c=2\*3.14\*r  c=round(c,2)  print("Circumference of circle: ",c) |

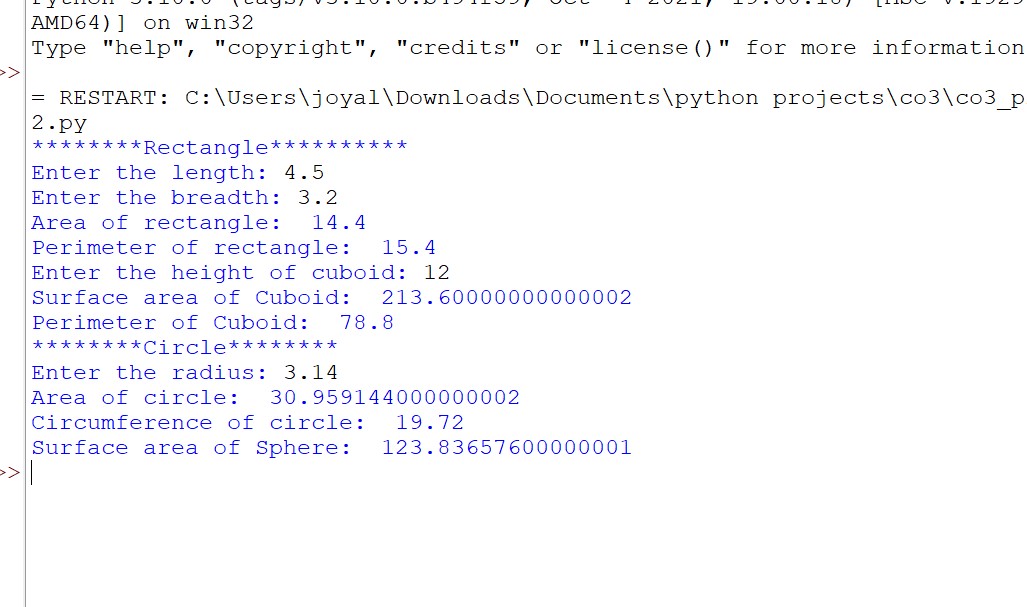
SUB MODULE

\_\_3D\_\_GRAPHICS

|  |  |
| --- | --- |
| **Cuboid.py** | def cuboidarea(l,b,h):  s=2\*((l\*b)+(b\*h)+(l\*h))  print("Surface area of Cuboid: ",s)  def cuboidperi(l,b,h):  p=4\*(l+b+h)  print("Perimeter of Cuboid: ",p) |

|  |  |
| --- | --- |
| **Sphere.py** | def sphere(r):  s=4\*3.14\*r\*r  print("Surface area of Sphere: ",s) |

**OUTPUT:**



**LABCYCLE 4 QUESTION 1**

**AIM:**

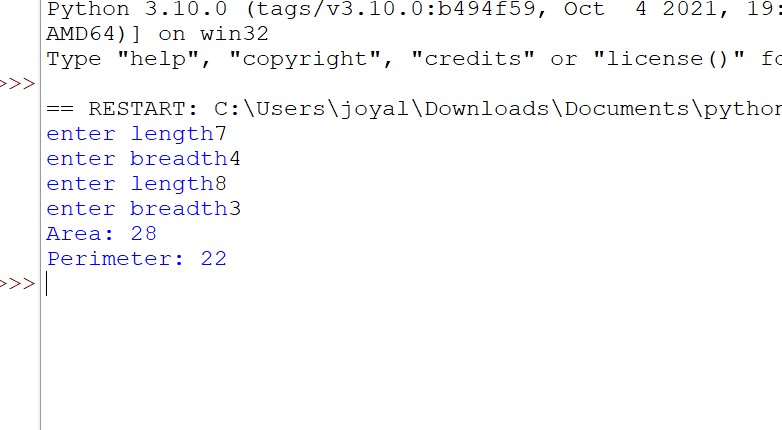
Create Rectangle class with attributes length and breadth and methods to find area and

perimeter. Compare two Rectangle objects by their area.

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out4p1.py** | class Rectangle:  def \_\_init\_\_(self,lenth,bread):  self.lenth=lenth  self.bread=bread  def area(self):  self.result=self.lenth\*self.bread  print("Area:",self.result)  def peri(self):  self.result=2\*(self.lenth+self.bread)  print("Perimeter:",self.result)  def compare(self):  print("Area of Rectangle1")    obj2=Rectangle(int(input("enter length")),int(input("enter breadth")))  obj1=Rectangle(int(input("enter length")),int(input("enter breadth")))  obj2.area()  obj1.peri() |

**OUTPUT:**

****

**LABCYCLE 4 QUESTION 2**

**AIM:**

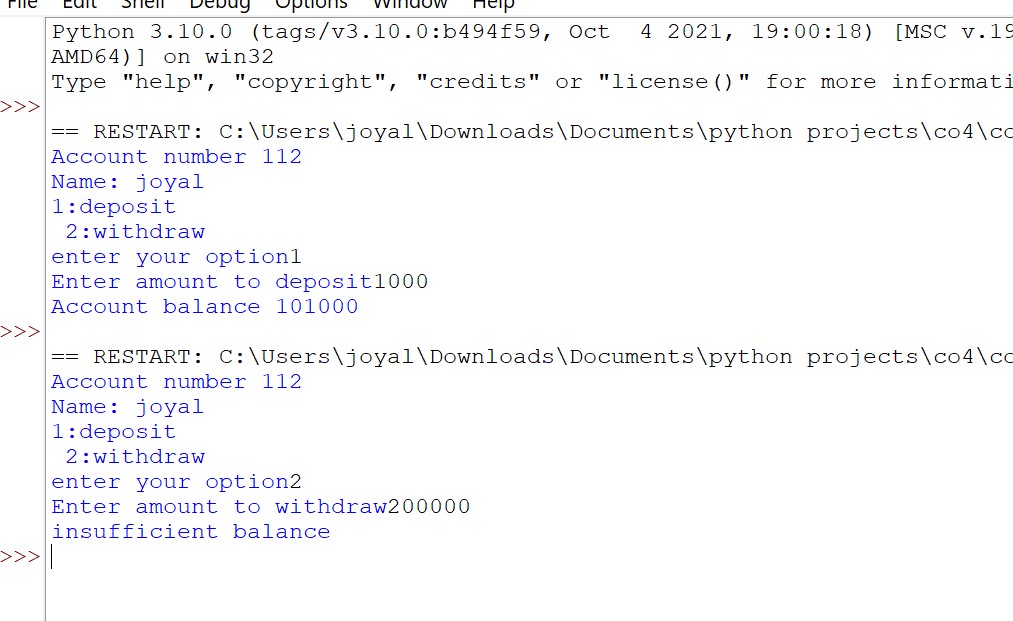
Create a Bank account with members account number, name, type of account and balance.

Write constructor and methods to deposit at the bank and withdraw an amount from the bank.

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out4p2.py** | class Account:  def \_\_init\_\_(self,ac,name,typeofac,balance):  self.ac=ac  self.name=name  self.typeofac=typeofac  self.balance=balance  def display(self):  print("Account number", self.ac)  print("Name:",str(self.name))  def withdraw(self):  if(self.balance==0):  print("Acount balance =",self.balance)  n=int(input("Enter amount to withdraw"))  if(n>self.balance):  print("insufficient balance")  else:  self.balance=self.balance-n  print("Account balance",self.balance)  def deposit(self):  n=int(input("Enter amount to deposit"))  self.balance=self.balance+n  print("Account balance",self.balance)  obj=Account(112,"joyal","savings",100000)  obj.display()  print("1:deposit\n 2:withdraw")  n=int(input("enter your option"))  if(n==2):  obj.withdraw()  elif(n==1):  obj.deposit() |

**OUTPUT:**



**LABCYCLE 4 QUESTION 3**

**AIM:**

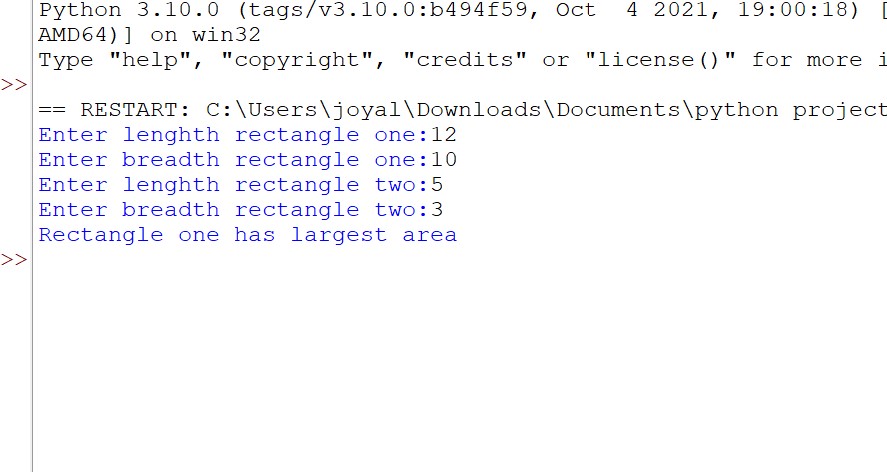
Create a class Rectangle with private attributes length and width. Overload ‘<’ operator to

compare the area of 2 rectangle

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out4p3.py** | class Rectangle:  def \_\_init\_\_(self,length,breadth):  self.\_\_length=length  self.\_\_breadth=breadth  self.\_\_area=length\*breadth  def \_\_lt\_\_(self,m):  return self.\_\_area<m.\_\_area  r=Rectangle(int(input("Enter lenghth rectangle one:")),int(input("Enter breadth rectangle one:")))  r1=Rectangle(int(input("Enter lenghth rectangle two:")),int(input("Enter breadth rectangle two:")))  if r<r1:  print("Rectangle two has largest area")  else:  print("Rectangle one has largest area") |

**OUTPUT:**

****

**LABCYCLE 4 QUESTION 4**

**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out4p4.py** | class Time:  def \_\_init\_\_(self,hour,minute,second):  self.\_\_hour=hour  self.\_\_minute=minute  self.\_\_second=second  def \_\_add\_\_(self,other):  h=self.\_\_hour+other.\_\_hour  m=self.\_\_minute+other.\_\_minute  s=self.\_\_second+other.\_\_second  return str(h)+str(m)+str(s)  t1=Time(11,34,10)  t2=Time(20,34,50)  t3=t1+t2  print(t3) |

**OUTPUT:**



**LABCYCLE 4 QUESTION 1**

**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

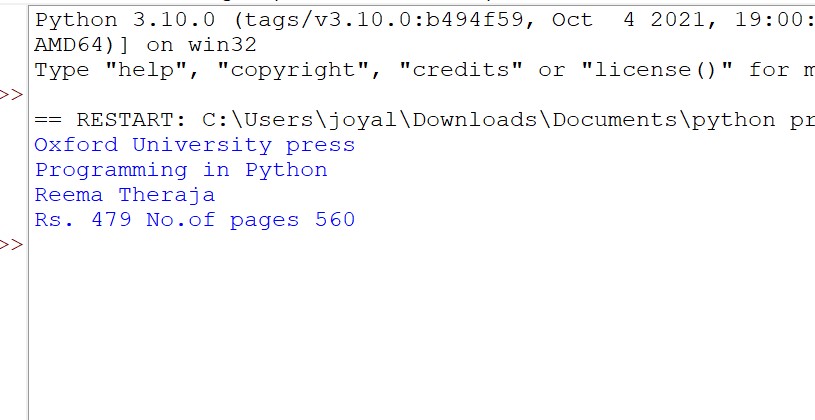
program that displays information about a Python book. Use base class constructor invocation and

method overriding. **:**

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out4p5.py** | class Publisher:  def \_\_init\_\_(self,name):  self.name=name  def disp(self):  print(self.name)  class Book(Publisher):  def \_\_init\_\_(self,name,title,auth):  Publisher.\_\_init\_\_(self,name)  self.title=title  self.auth=auth  def disp(self):  print(self.title,self.author)  class Python(Book):  def \_\_init\_\_(self,name,title,auth,price,nop):  Book.\_\_init\_\_(self,name,title,auth)  self.price=price  self.nop=nop  def disp(self):  print(self.name)  print(self.title)  print(self.auth)  print("Rs.",self.price,"No.of pages",self.nop)  obj=Python("Oxford University press","Programming in Python","Reema Theraja",479,560)  obj.disp() |

**OUTPUT:**



**LABCYCLE 5 QUESTION 1**

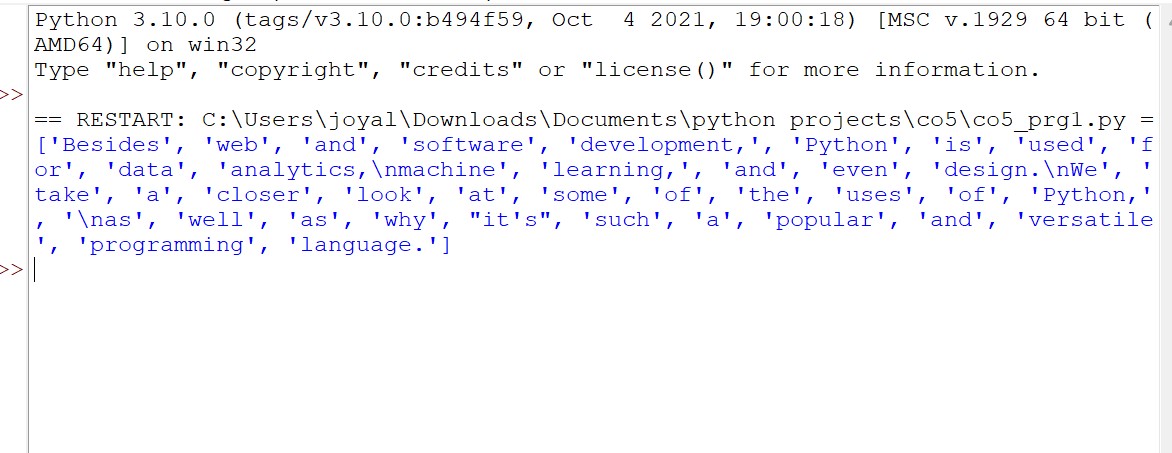
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out5p1.py** | fr=open("test.txt","r")  s=fr.read()  w=s.split(" ")  print(w) |

**OUTPUT:**

****

**LABCYCLE 5 QUESTION 2**

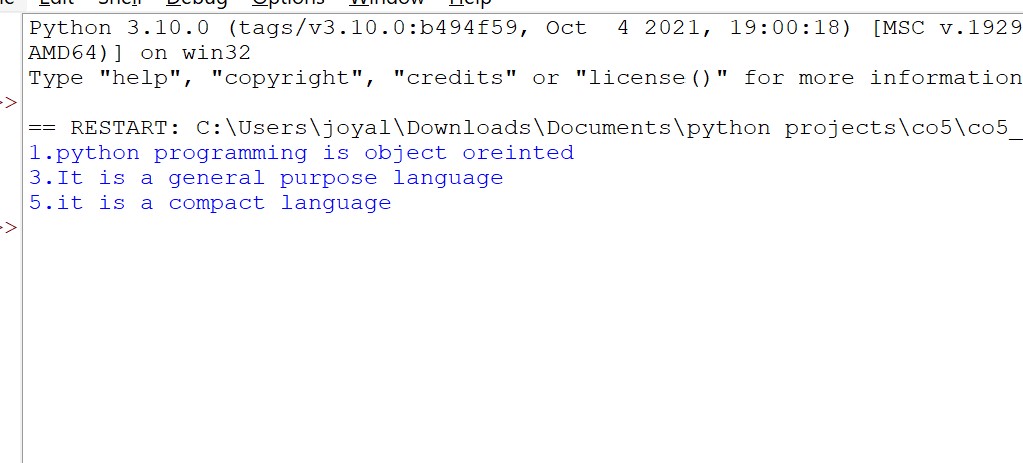
**AIM:**

Python program to copy odd lines of one file to other

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out5p2.py** | fn = open('test1.txt', 'r')  fn1 = open('nfile.txt', 'w')  cont = fn.readlines()  type(cont)  for i in range(0, len(cont)):  if(i%2!=0):  fn1.write(cont[i])  else:  pass  fn1.close()  fn1 = open('nfile.txt', 'r')  cont1 = fn1.read()  print(cont1)  fn.close()  fn1.close() |

**OUTPUT:**

****

**LABCYCLE 5 QUESTION 3**

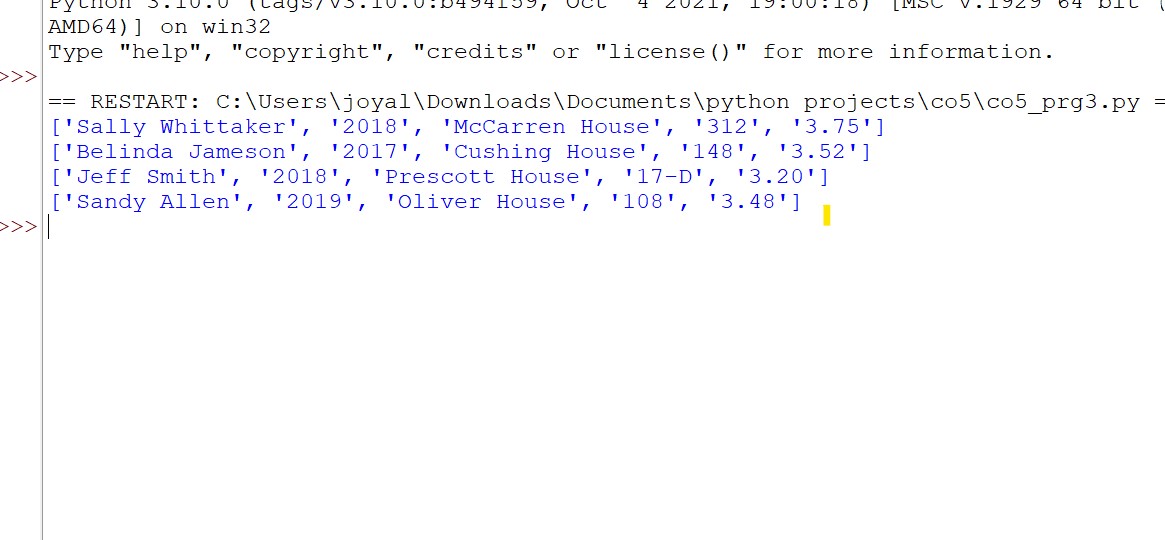
**AIM:**

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

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out5p3.py** | import csv  with open("csvtest.csv","r") as csv\_file:  csv\_reader=csv.reader(csv\_file)  for line in csv\_reader:  print(line) |

**OUTPUT:**

****

**LABCYCLE 5 QUESTION 4**

**AIM:**

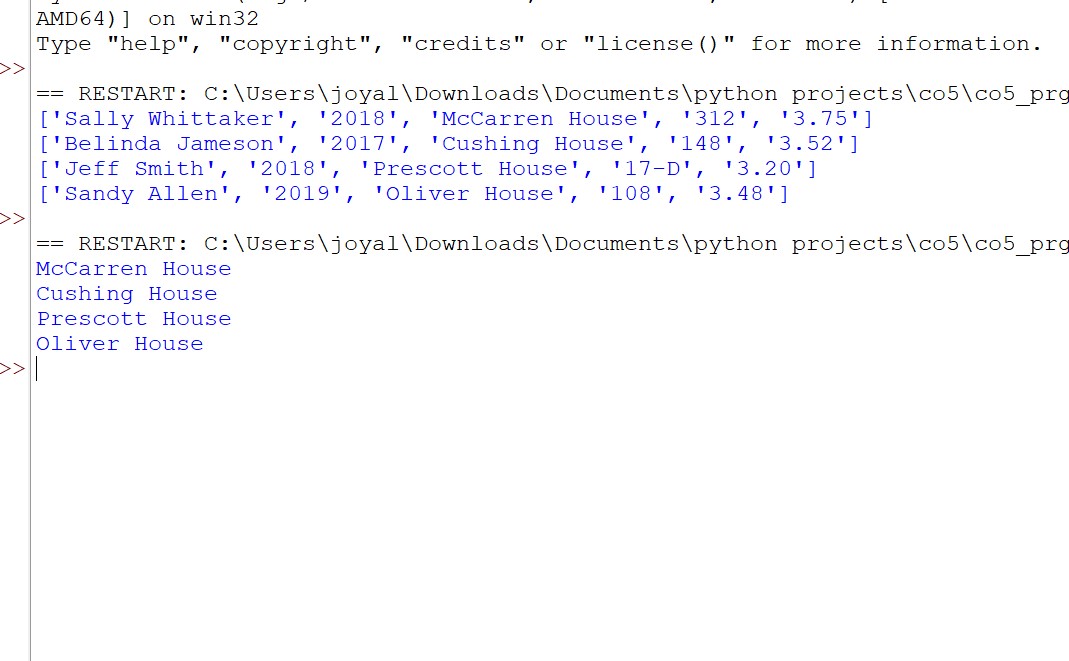
Write a Python program to read specific columns of a given CSV file and print the content

of the columns

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out5p4.py** | import csv  with open("csvtest.csv","r") as csv\_file:  csv\_reader=csv.reader(csv\_file)  for line in csv\_reader:  print(line[2])#column 3 |

**OUTPUT:**



**LABCYCLE 5 QUESTION 5**

**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.

**PROGRAM CODE:**

|  |  |
| --- | --- |
| **Out5p5.py** | #dictionary to csv  import csv  dict\_value = [  {"name":"Manas","age":27,"course":"MBA"},  {"name":"Biju","age":23,"course":"MCA"},  {"name":"Anandhu","age":20,"course":"BSC"}  ]  fields = ["name","age","course"]  with open('dictconverted.csv','w') as csvfile:  writer = csv.DictWriter(csvfile,fieldnames=fields)  writer.writeheader()  writer.writerows(dict\_value)  csvfile.close()  with open('dictconverted.csv','r') as csvfiles:  readerobj = csv.reader(csvfiles)  for rows in readerobj:  print(rows) |

**OUTPUT:**

