DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB MANUAL

SUBJECT: PYTHON PROGRAMMING LAB

YEAR AND SEM: II-I

SECTION: A, B, C, D

ACADEMIC YEAR: 2017-18

Submitted by

A.Leelavathi

SRI VASAVI ENGINEERING COLLEGE
PEDATADEPALLI

INDEX

COURSE OBJECTIVES AND OUTCOMES

1. Course Objectives and Course Outcomes

Course Objectives:

In this lab, students will learn:

1. How to use Control structures in Python
2. How to use objects in a program and How to invoke an object's methods
3. How to use Functions, Modules, Packages.
4. How to use OOP Concepts in Python
5. How to use methods of Turtle objects and Turtle Graphics

Course Outcomes:

1. Demonstrate the interactive python, Loop structures conditional programming (K2)
2. Apply the logic for loops programs like Fibonacci series, Data Structures (K3)
3. Illustrate file programs (K3)
4. Operate Scope of variables, Modules, Packages (K3)
5. Examine the concept of OOP, Exception handling (K3)
6. Compute the concept of Multithreading, GUI Programming and testing (K3).

INTRODUCTION TO THE LAB

2.Introduction to the Lab

Python is a powerful high-level, object-oriented programming language created by Guido van Rossum.It has simple easy-to-use syntax, making it the perfect language for someone trying to learn computer programming for the first time.This is a comprehensive guide on how to get started in Python, why you should learn it and how you can learn it.

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

Why Python was created?

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

Why the name Python?

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

Release Dates of Different Versions

Version	Release Data
Python 1.0 (first standard release) Python 1.6 (Last minor version)	January 1994 September 5, 2000
Python 2.0 (Introduced list comprehensions) Python 2.7 (Last minor version)	October 16, 2000 July 3, 2010
Python 3.0 (Emphasis on removing duplicative constructs and module) Python 3.5 (Last updated version)	December 3, 2008 September 13, 2015

Features of Python:

A simple language which is easier to learn ,Free and open-source, Portability, Extensible and Embeddable ,A high-level, interpreted language, Large standard libraries to solve common tasks, Object-oriented Language.

Applications:

Web Applications

You can create scalable Web Apps using frameworks and CMS (Content Management System) that are built on Python. Some of the popular platforms for creating Web Apps are: Django, Flask, Pyramid, Plone, Django CMS.

Sites like Mozilla, Reddit, Instagram and PBS are written in Python.

Scientific and Numeric Computing

There are numerous libraries available in Python for scientific and numeric computing. There are libraries like: SciPy and NumPy that are used in general purpose computing. And, there are specific libraries like: EarthPy for earth science, AstroPy for Astronomy and so on.

Also, the language is heavily used in machine learning, data mining and deep learning.

Creating software Prototypes

Python is slow compared to compiled languages like C++ and Java. It might not be a good choice if resources are limited and efficiency is a must.

However, Python is a great language for creating prototypes. For example: You can use Pygame (library for creating games) to create your game's prototype first. If you like the prototype, you can use language like C++ to create the actual game.

Good Language to Teach Programming

Python is used by many companies to teach programming to kids and newbies.

It is a good language with a lot of features and capabilities. Yet, it's one of the easiest language to learn because of its simple easy-to-use syntax.

LIST OF LAB EXERCISES

3.List of Lab Exercises

Exercise 1 - Basics

a) Running instructions in Interactive interpreter and a Python Script

b) Write a program to purpose fully raise Indentation Error and Correct it

Exercise 2 - Operations

a) Write a program to compute distance between two points taking input from the user (Pythagorean Theorem)

b) Write a program add.py that takes 2 numbers as command line arguments and prints its sum.

Exercise - 3 Control Flow

a) Write a Program for checking whether the given number is a even number or not.

b) Using a for loop, write a program that prints out the decimal equivalents of 1/2, 1/3, 1/4, . . . ,1/10

c) Write a program using a for loop that loops over a sequence. What is sequence ?

d) Write a program using a while loop that asks the user for a number, and prints a countdown from that number to zero.

Exercise 4 - Control Flow - Continued

a) Find the sum of all the primes below two million. Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...

b) By considering the terms in the Fibonacci sequence whose values do not exceed four million, find the sum of the even-valued terms.

Exercise - 5 - DS

a) Write a program to count the number of characters in the string and store them in a dictionary data structure

b) Write a program to use split and join methods in the string and trace a birthday with a dictionary data structure.

Exercise - 6 -DS

a) Write a program combine_lists that combines these lists into a dictionary.

b) Write a program to count frequency of characters in a given file. Can you use character frequency to tell whether the given file is a Python program file, C program file or a text file?

Exercise - 7 Files

a) Write a program to print each line of a file in reverse order.

b) Write a program to compute the number of characters, words and lines in a file.

Exercise - 8 Functions

a) Write a function ball_collide that takes two balls as parameters and computes if they are colliding. Your function should return a Boolean representing whether or not the balls are colliding.

Hint: Represent a ball on a plane as a tuple of (x, y, r), r being the radius

If (distance between two balls centers) <= (sum of their radii) then (they are colliding)

b) Find mean, median, mode for the given set of numbers in a list.

Exercise - 9 Functions - Continued

a) Write a function nearly_equal to test whether two strings are nearly equal. Two strings a and b are nearly equal when a can be generated by a single mutation on b.

b) Write a function dups to find all duplicates in the list.

c) Write a function unique to find all the unique elements of a list.

Exercise - 10 - Functions - Problem Solving

a) Write a function cumulative_product to compute cumulative product of a list of numbers.

b) Write a function reverse to reverse a list. Without using the reverse function.

c) Write function to compute gcd, lcm of two numbers. Each function shouldn't exceed one line.

Exercise 11 - Multi-D Lists

a) Write a program that defines a matrix and prints

b) Write a program to perform addition of two square matrices

c) Write a program to perform multiplication of two square matrices

Exercise - 12 - Modules

a) Install packages requests, flask and explore them. using (pip)

b) Write a script that imports requests and fetch content from the page. Eg. (Wiki)

c) Write a simple script that serves a simple HTTP Response and a simple HTML Page

Exercise - 13 OOP

a) Class variables and instance variable

i) Robot

ii) ATM Machine

Exercise - 14 GUI, Graphics

1. Write a GUI for an Expression calculator using Tk

2. Write a program to implementing the following figures using Turtle

Exercise - 15 - Testing

a) Write a test-case to check the even numbers which return T on passing a list of all Even numbers

b) Write a test-case to check the function reverse-string which returns the reversed string

Exercise - 16 – Advanced DS

a) Build any one classical data structure.

b) Write a program to solve knapsack problem.

SOLUTIONS FOR PROGRAMS

4. Solutions for Programs

Exercise 1 – Basics

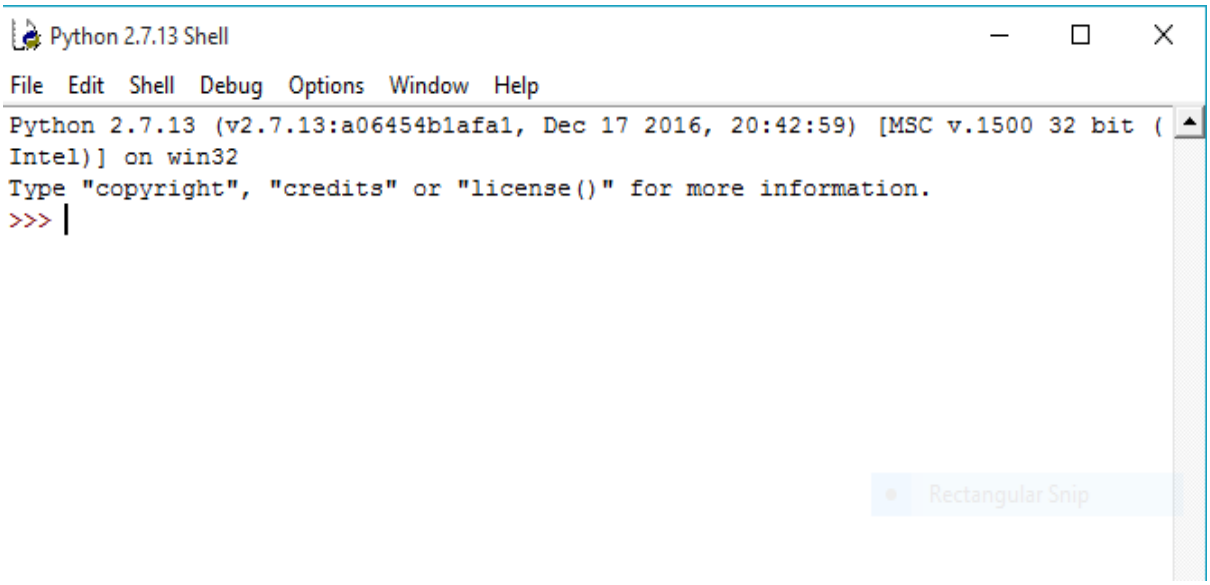
Question:

1a) Running instructions in Interactive interpreter and a Python Script

Description:

Interactive Mode:

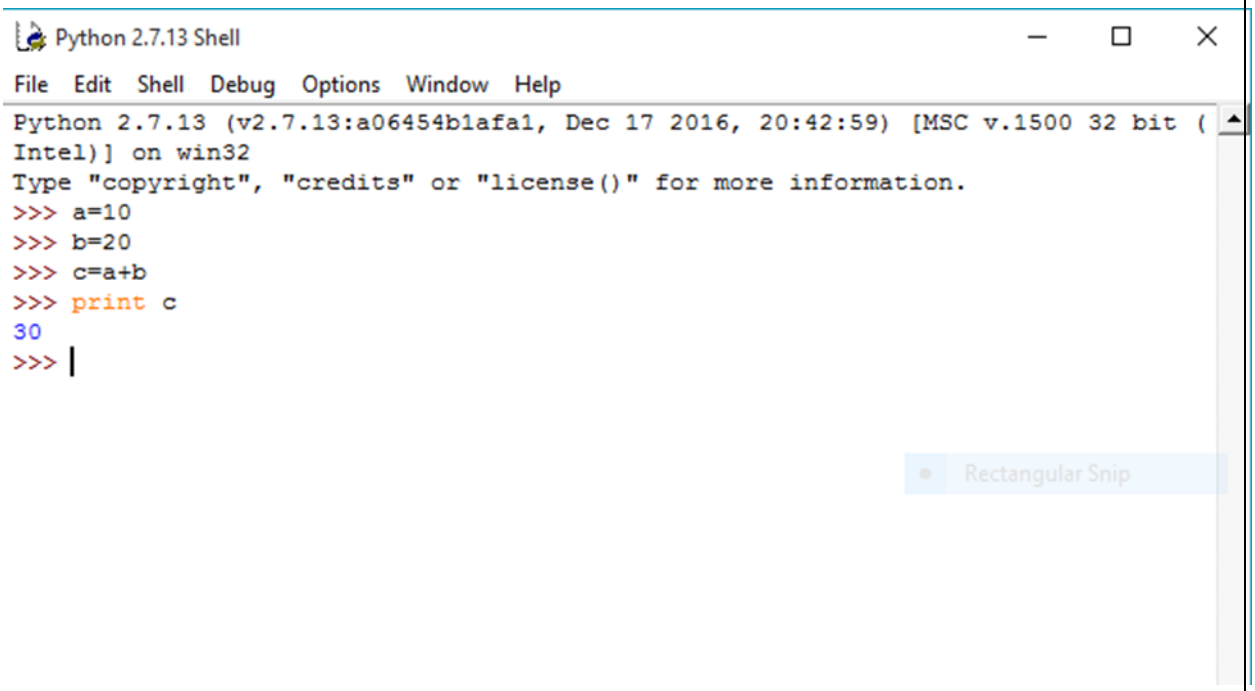
- When commands are read from a tty, the interpreter is said to be in interactive mode.
- In this mode it prompts for the next command with the primary prompt, usually 3 greater than signs(>>>); for continuation lines it prompts with the secondary prompt, by default 3 dots(…).
- The interpreter prints a welcome message stating its version number and a copyright notice before printing the first prompt:



Running python programs using interactive mode:

Step-1: Click on start-> All programs->python2.7->IDLE (python GUI).

Step-2: Type the sample code in IDLE as follows:



(or)

Step-1: Click on start-> All programs->python2.7->IDLE.
Step-2: Click on file menu, click on New file.
Step-3: Type the following code

```
a=10
b=20
c=a+b
print c
```

Step-4: Save this file (c:\python2.7\addex.py)
Step-5: Press F5 (to run the program).

Running python programs using command prompt

Step-1: Open the notepad editor.
Step-2: Type the following code

```
a=10
b=20
c=a+b
print c
```

Step-3: Save the file with .py extension (ex: d:\vasavi\addex.py).
Step-4: Open the command prompt (start->Run->cmd).
Step-5: Go to your folder location where you want to save the python program (ex:d:\vasavi).
Step-6: Type the following command for execution

```
python addex.py
```

Question:

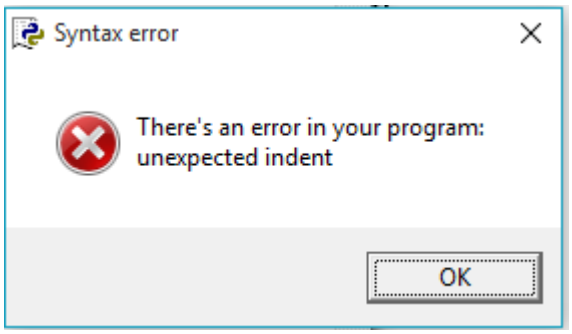
1b)Write a program to purposefully raise Indentation Error and Correct it

Program:

```
a=input("enter a value:")
b=input("enter b value:")
```

```
        c=a*b #unexpected indent
print c
```

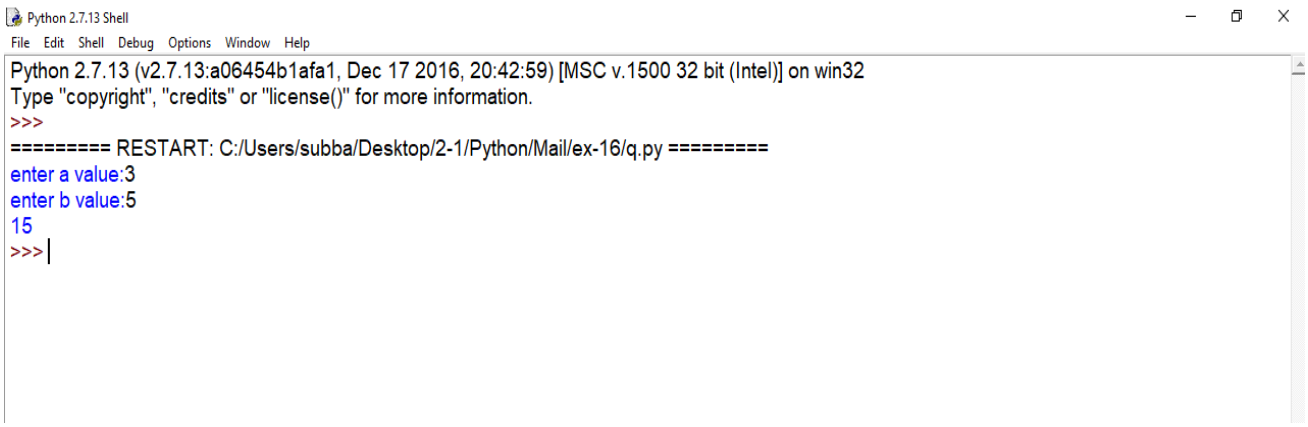
Output:



Correction:

```
a=input("enter a value:")
b=input("enter b value:")
c=a*b #corrected indent
print c
```

Output:



Exercise 2 - Operations

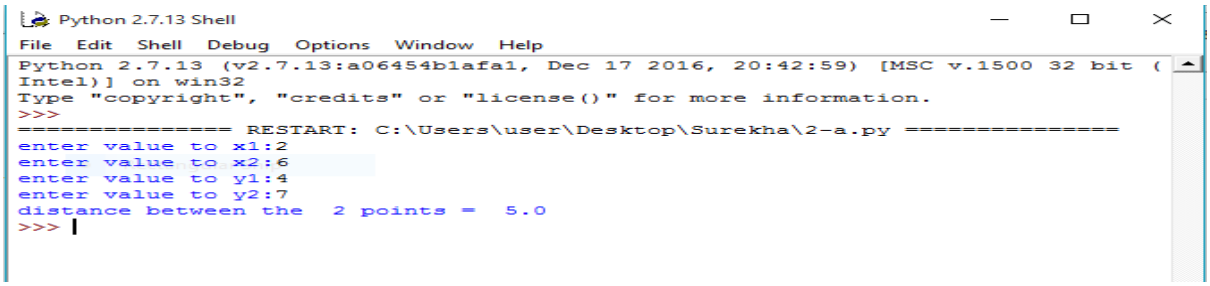
Question:

2a)Write a program to compute distance between two points taking input from the user (Pythagorean Theorem)

Program:

```
import math
x1=input("enter value to x1:")
x2=input("enter value to x2:")
y1=input("enter value to y1:")
y2=input("enter value to y2:")
x=x2-x1
y=y2-y1
distance=math.sqrt((y**2)+(x**2))
print "distance between the  2 points = ",distance
```

Output:



Question:

2b)Write a program add.py that takes 2 numbers as command line arguments and prints its sum.

Program:

```
import sys
a=int(sys.argv[1])
b=int(sys.argv[2])
c=a+b
print c
```

Output:

Exercise - 3 Control Flow

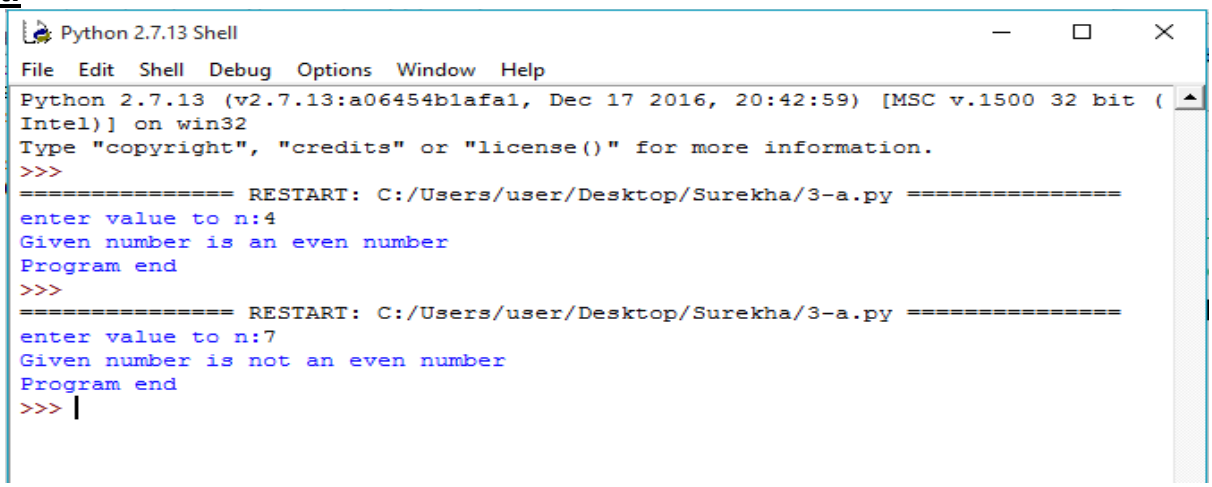
Question:

3a)Write a Program for checking whether the given number is a even number or not.

Program:

```
n=input("enter value to n:")
if n%2==0:
    print("Given number is an even number")
else:
    print("Given number is not an even number")
print("Program end")
```

Output:



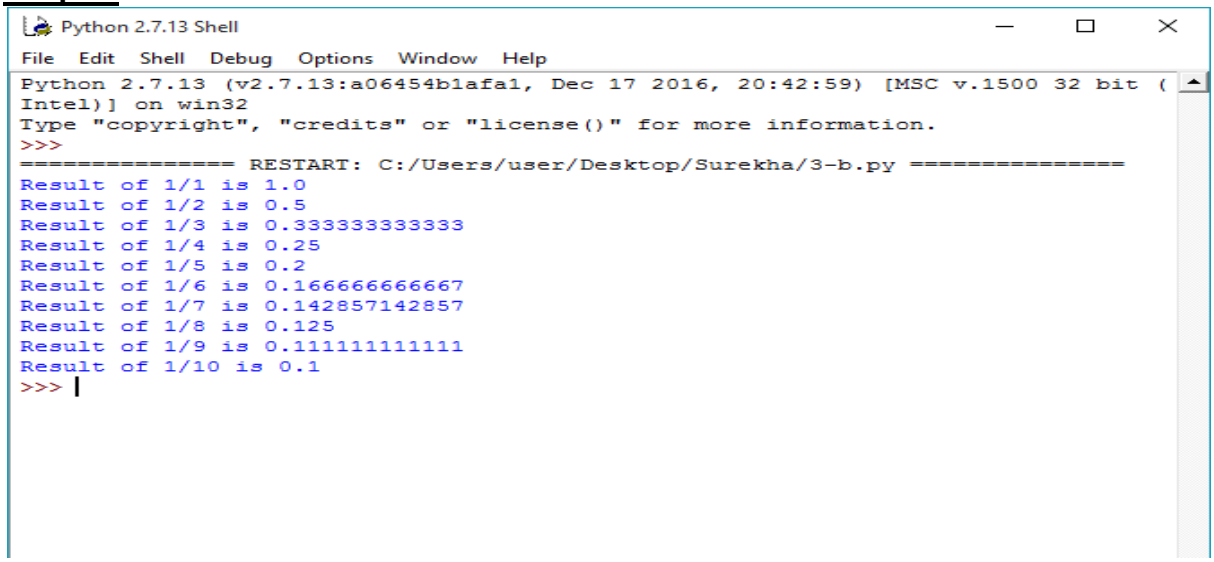
Question:

3b)Using a for loop, write a program that prints out the decimal equivalents of 1/2, 1/3, 1/4, . . . , 1/10

Program:

```
for i in range(1,11,1):
  n=(1.0/i)
  print("Result of {}/{} is {}".format(1,i,n))
```

Output:



Question:

3c)Write a program using a for loop that loops over a sequence. What is sequence ?

Description:

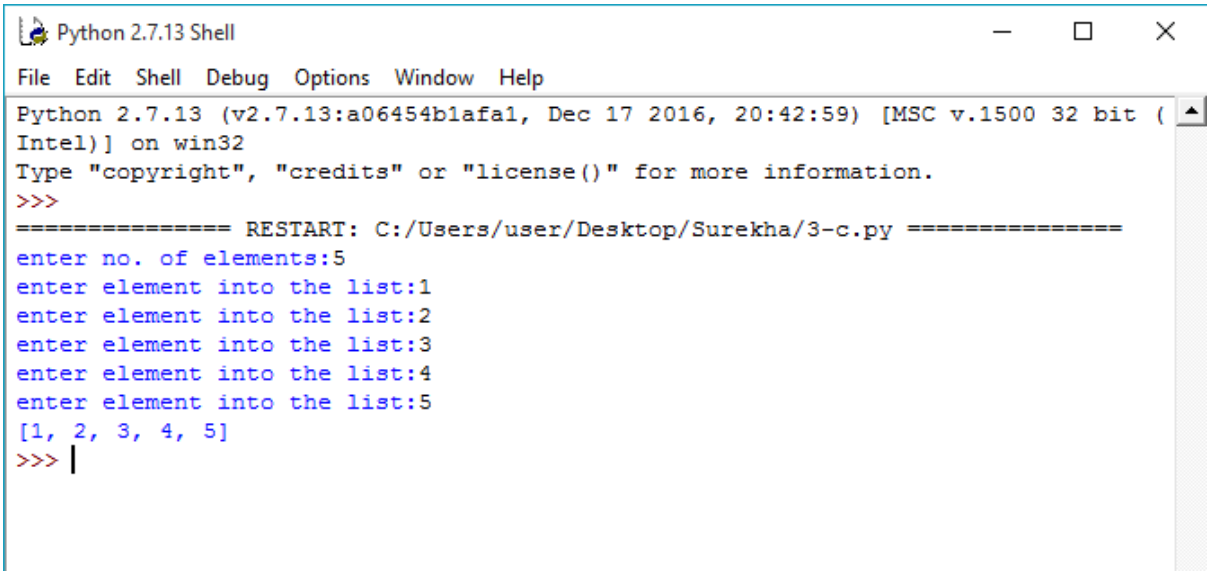
Sequence: In python, sequence is the generic term for an ordered set. There are several types of sequences in python. The following 3 are most important among them.

- ✓ **List:** It is a type of sequence – we can use it to store multiple values, and access them sequentially, by their position, or index, in the list. List is a mutable data structure.
- ✓ **Tuple**: Python has another sequence type which is called tuple. Tuples are similar to lists in many ways, but they are immutable.
- ✓ **String**: String is a special type of sequence that can only store characters and they have a special notation.

Program:

```
n=input("enter no. of elements:")
                list=[]
                for i in range(0,n):
    x=input("enter element into the list:")
    list.append(x)
                print list
```

Output:



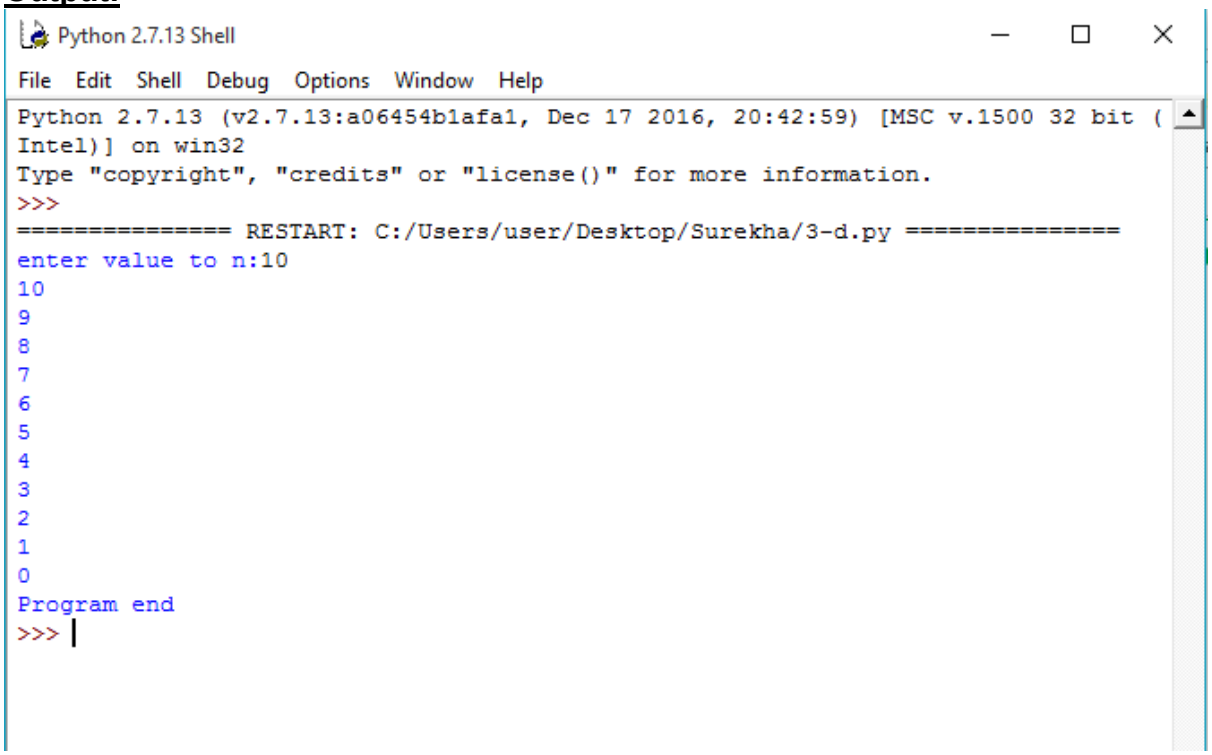
Question:

3d)Write a program using a while loop that asks the user for a number, and prints a countdown from that number to zero

Program:

```
n=input("enter value to n:")
while n>=0:
    print n
    n=n-1
print("Program end")
```

Output:


Python 2.7.13 Shell
File Edit Shell Debug Options Window Help
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/user/Desktop/Surekha/3-d.py ==========
enter value to n:10
10
9
8
7
6
5
4
3
2
1
0
Program end
>>>

Exercise 4 - Control Flow - Continued

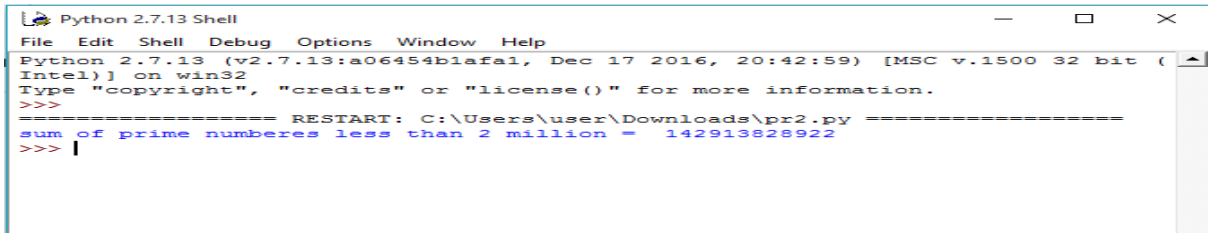
Question:

4a) Find the sum of all the primes below two million

Program:

```
m = 2000000
numbers = set(xrange(3, m + 1, 2))
for number in xrange(3, int(m ** 0.5) + 1):
if number not in numbers:
    continue
                        num = number
                while num < m:
                                num += number
        if num in numbers:
          numbers.remove(num)
print "sum of prime numberes less than 2 million = ",2 + sum(numbers)
```

Output:



Question:

4b)Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be:

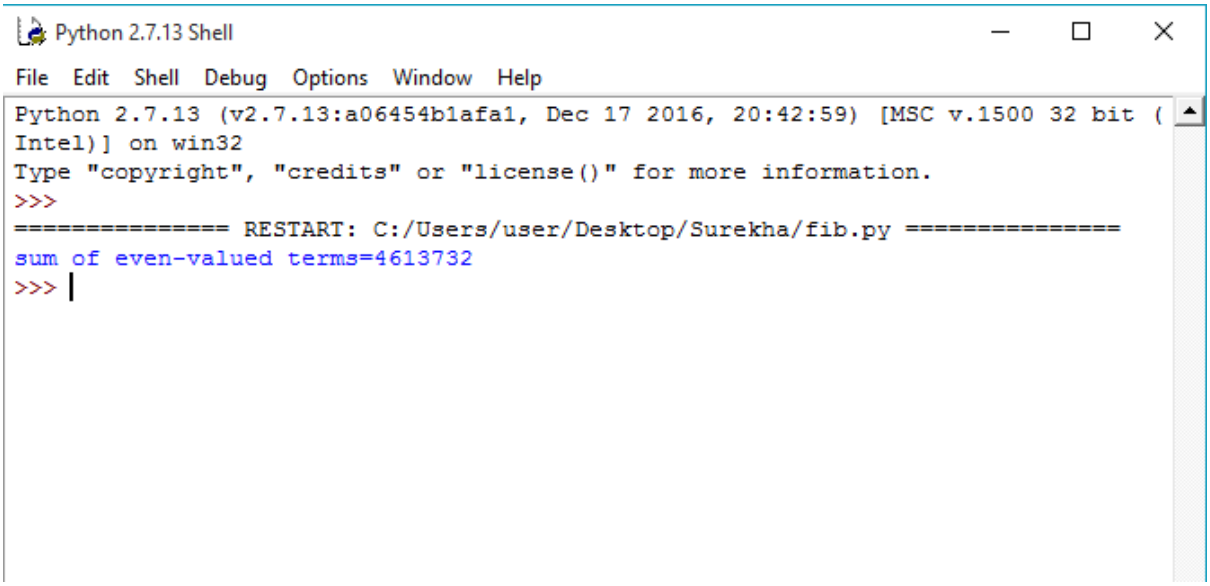
1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...

By considering the terms in the Fibonacci sequence whose values do not exceed four million, find the sum of the even-valued terms.

Program:

```
f1=1
f2=0
f3=0
fsum=0
while f3<4000000:
      f3=f1+f2
f1=f2
f2=f3
if f3%2==0:
  fsum+=f3
print("sum of even-valued terms={}".format(fsum))
```

Output:


Python 2.7.13 Shell
File Edit Shell Debug Options Window Help
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/user/Desktop/Surekha/fib.py ==========
sum of even-valued terms=4613732
>>>

Exercise - 5 - DS

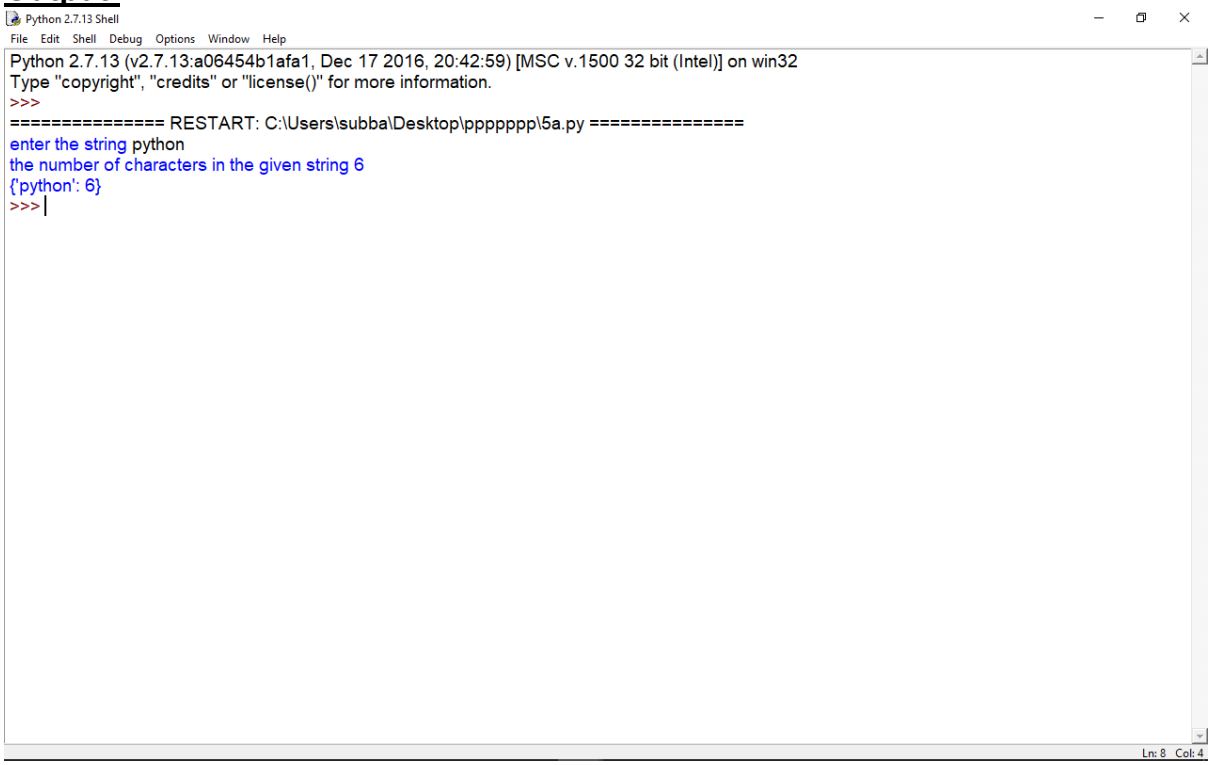
Question:

5a)Write a program to count the numbers of characters in the string and store them in a dictionary data structure

Program:

```
s=raw_input("enter the string ")
no_char=0
for x in s :
    no_char+=1
#d={s:no_char}
d=dict([(s,no_char)])
print "the number of characters in the given string",no_char
print d
```

Output:



Question:

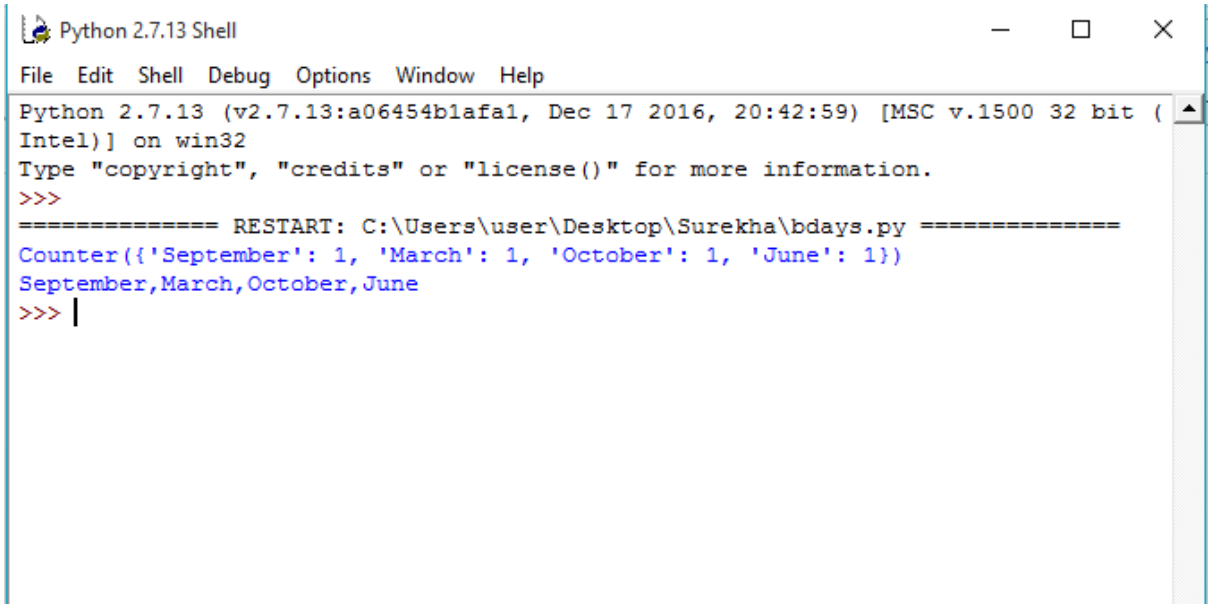
5b)Write a program to use split and join methods in the string and trace a birthday with a dictionary data structure.

Program:

```
from collections import Counter
birthdays={
"surekha":"6/22/1999",
"divya":"10/13/1999",
"sushma":"9/17/1998",
"prasanna":"03/4/1999"
}
#print birthdays.split("/")
```

```
num_to_string = {
  1: "January",
  2: "February",
  3: "March",
  4: "April",
  5: "May",
  6: "June",
  7: "July",
  8: "August",
  9: "September",
  10: "October",
  11: "November",
  12: "December"
}
months = []
mn=","
for name, birthday_string in birthdays.items():
  month = int(birthday_string.split("/")[0])
  months.append(num_to_string[month])
print(Counter(months))
print mn.join(Counter((months)))
```

Output:



Exercise - 6 DS - Continued

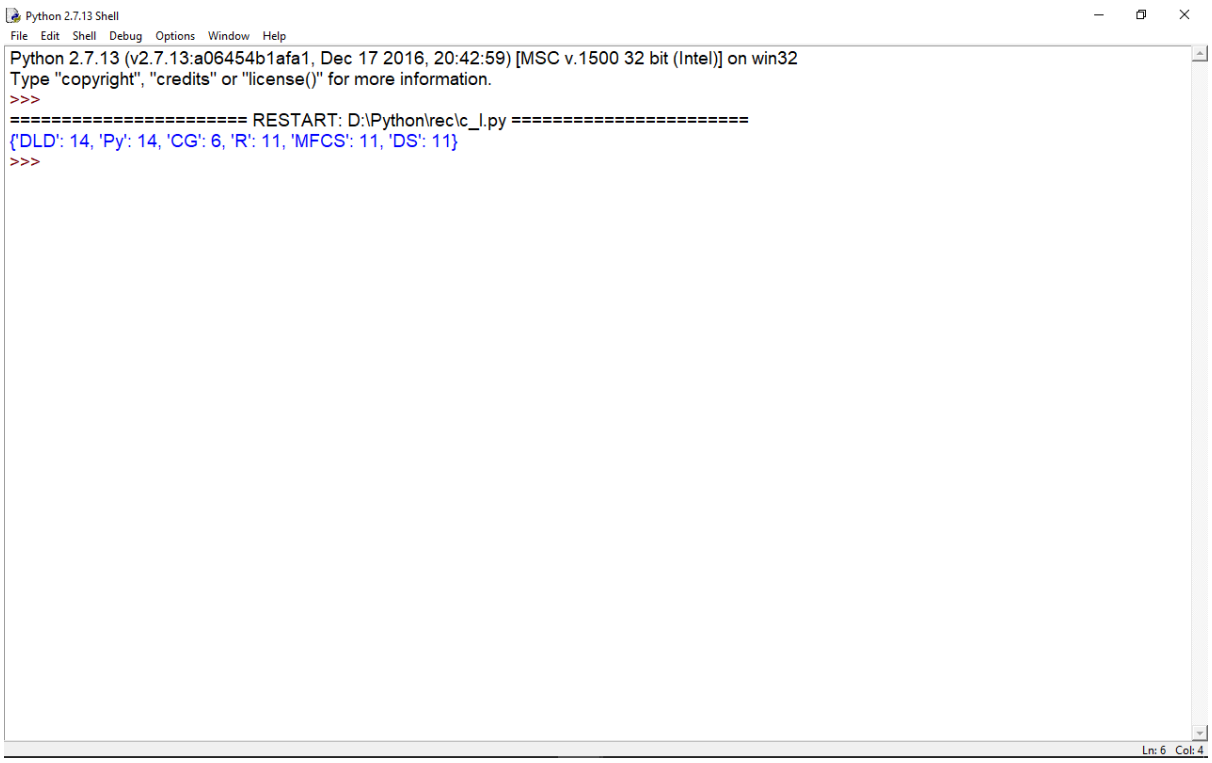
Question:

6a)Write a program combine_lists that combines these lists into a dictionary.

Program:

```
list1=['Py','R','CG','MFCS','DLD','DS']
list2=[14,11,6,11,14,11]
combine_lists=dict(zip(list1,list2))
print combine_lists
```

Output:



Question:

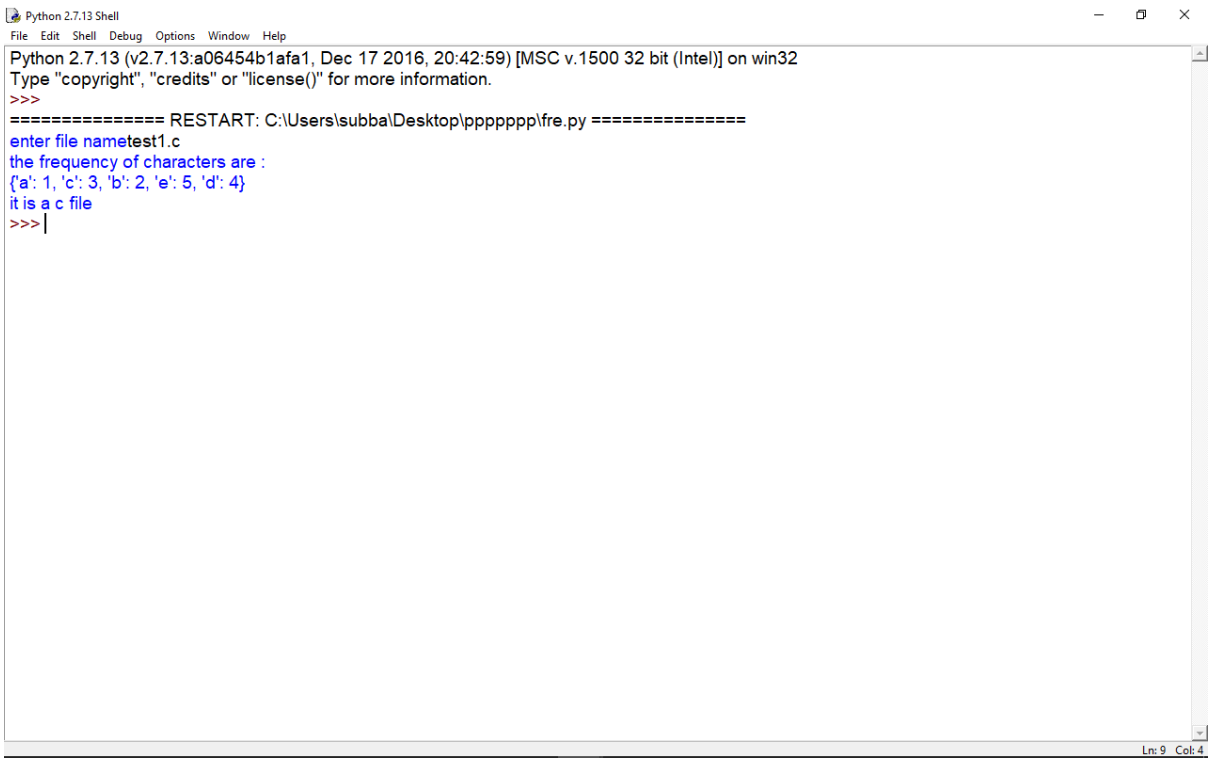
6b)Write a program to count frequency of characters in a given file. Can you use character frequency to tell whether the given file is a Python program file, C program file or a text file?

Program:

```
fname=raw_input("enter file name")

f=open(fname,'r')

c=0

s=f.read()

d={x:c for x in s}

for i in d :

    d[i] = s.count(i)
```

```
if ' ' in d :
    del d[' ']
if '\n' in d :
    del d['\n']
print "the frequency of characters are :"
print d
if fname.endswith(".py") :
    print "it is a python file"
elif fname.endswith(".c") :
    print "it is a c file "
elif fname.endswith(".txt") :
    print "it is a text file"
else :
    print "it is a file rather than python,c,test file"
f.close()
```

Output:



Exercise - 7 Files

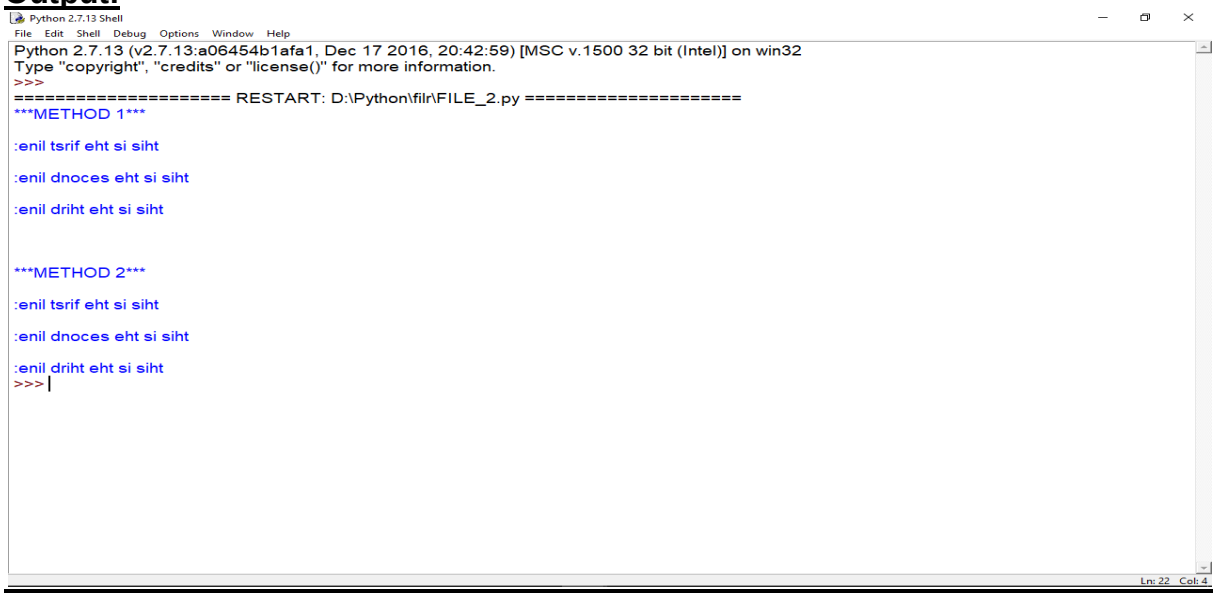
Question:

7a)Write a program to print each line of a file in reverse order

Program:

```
f=open("sample.txt","w")
line1="this is the first line:\n"
line2="this is the second line:\n"
line3="this is the third line:\n"
f.write(line1)
f.write(line2)
f.write(line3)
f.close()
print('***METHOD 1***')
f=open("sample.txt",'r')
count=0
while count < 4:
    l1=f.readline()
    print(l1[::-1])
    count+=1
f.close()
#2
print("\n\n***METHOD 2***")
with open('sample.txt','r') as f2 :
    for x in f2:
        print(x[::-1])\
```

Output:



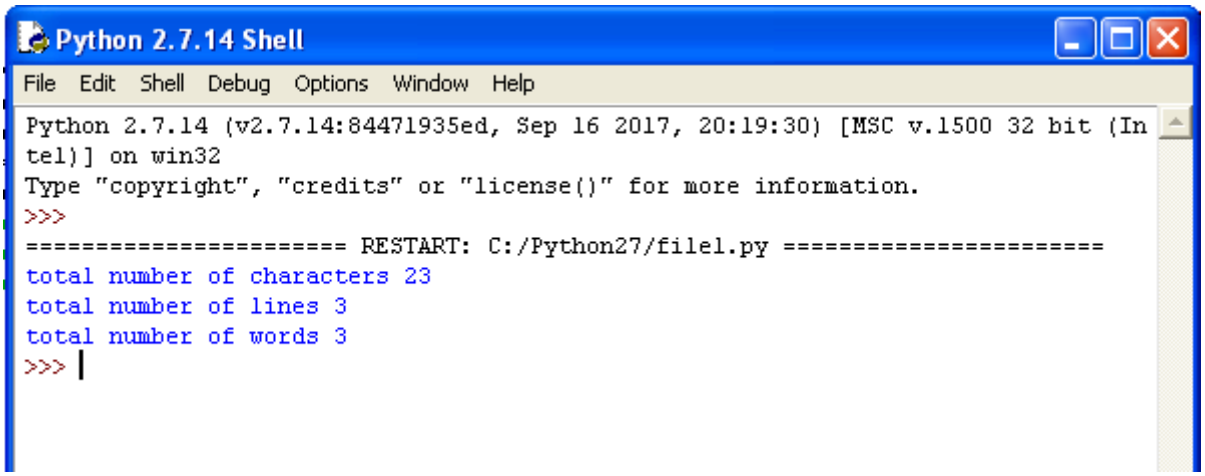
Question:

7b)Write a program to compute the number of characters, words and lines in a file.

Program:

```
myfile=open("student.txt","r")
numchars=0
numlines=0
numwords=0
for line in myfile:
    numchars=numchars+len(line)
    numlines=numlines+1
    wordslist=line.split(",")
    numwords=numwords+len(wordslist)
print "total number of characters",numchars
print "total number of lines",numlines
print "total number of words",numwords
myfile.close()
```

Output:



Exercise - 8 Functions

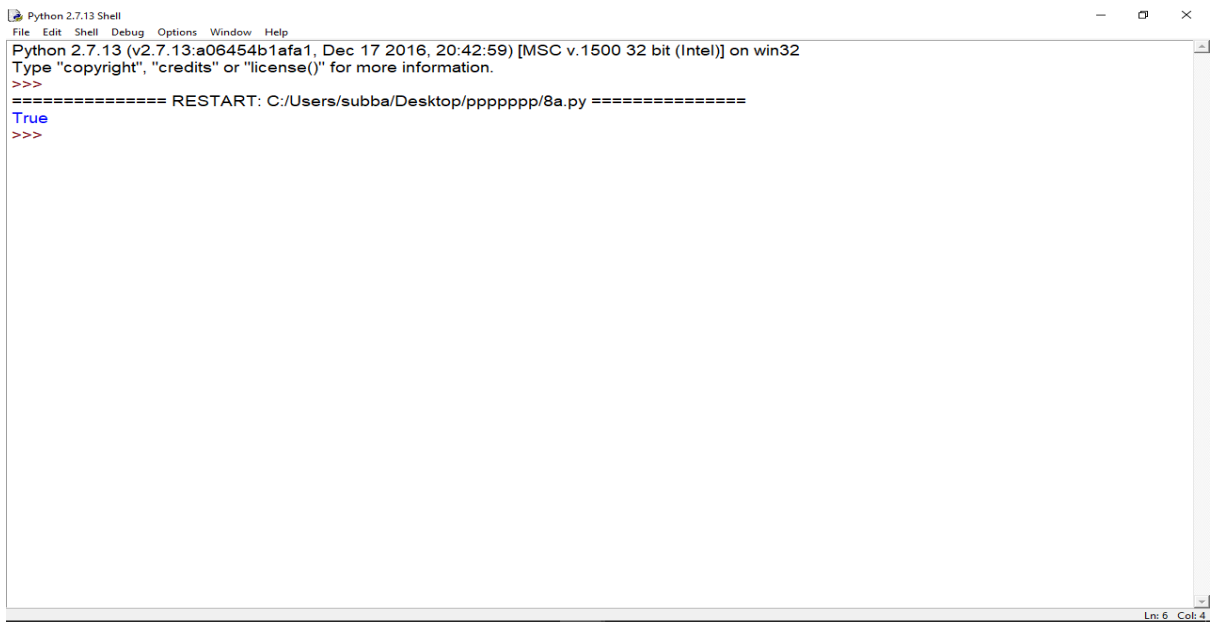
Question:

8a)write a function ball_collide that takes two balls as parameters and computes if they are colliding.your function should return a Boolean representing whether or not the balls are colliding

Program:

```
import math

def ball_collide(b1,b2):
    x=b2[0]-b1[0]
    y=b2[1]-b1[1]
    d=math.sqrt(x**2+y**2)
    r=b1[2]+b2[2]
    if(d<=r):
        return True
    else:
        return False
b1=(4,5,6)
b2=(7,8,9)
print ball_collide(b1,b2)
```

Output:



Question:

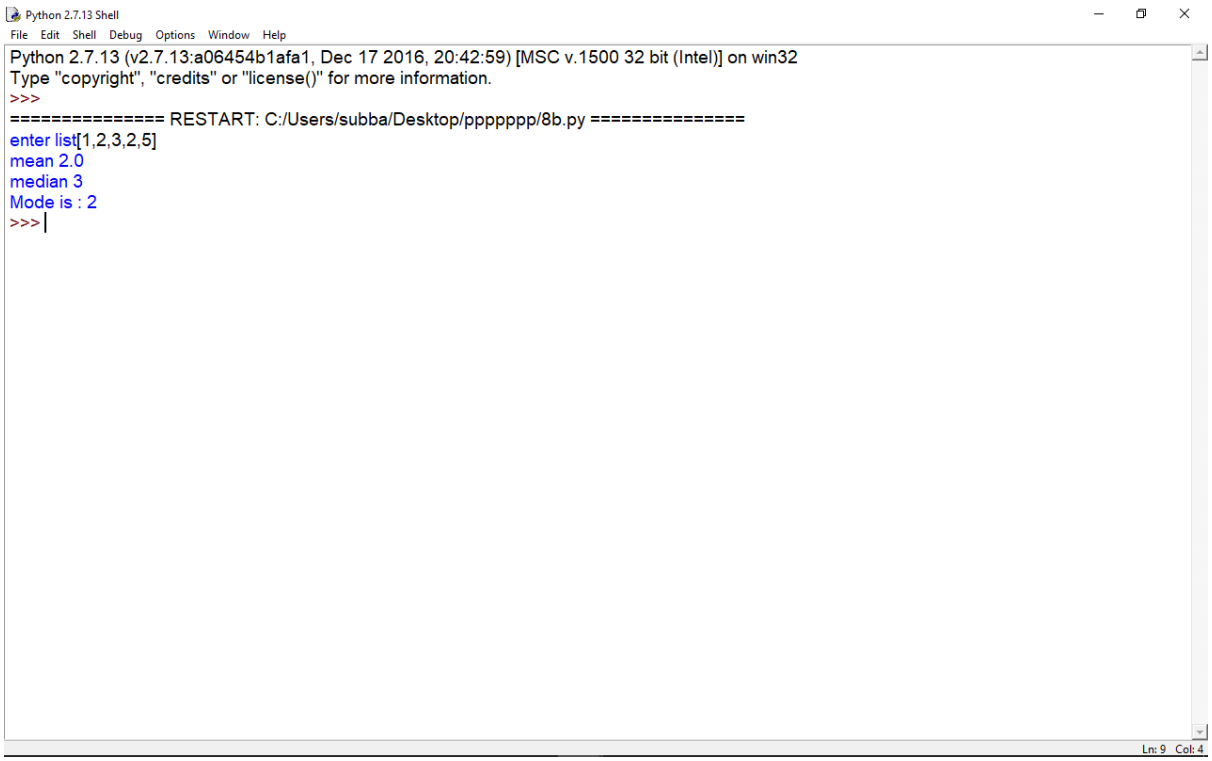
8b)Find mean, median, mode for the given set of numbers in a list

Program:

```
def mean(l):
    sum=0
    for i in l:
        sum+=i
    n=len(l)
    m=float(sum/n)
    print "mean",m
def median(l):
    n=len(l)
    sum=0
    l.sort()
    if(n%2==0):
        sum=l[n/2]+l[(n/2)+1]
        m=float(sum/2)
    else:
        m=l[(n+1)/2]
    print"median",m
def mode(l) :
    count=0
    s=set(l)
    d={x:count for x in s}
    for k in d :
        d[k]=l.count(k)
    m=max(d.values())
    for i,j in d.items() :
        if j==m :
            print "Mode is :",i
```

```
listx=input("enter list")
mean(listx)
median(listx)
mode(listx)
```

Output:



Exercise - 9 Functions – Continued

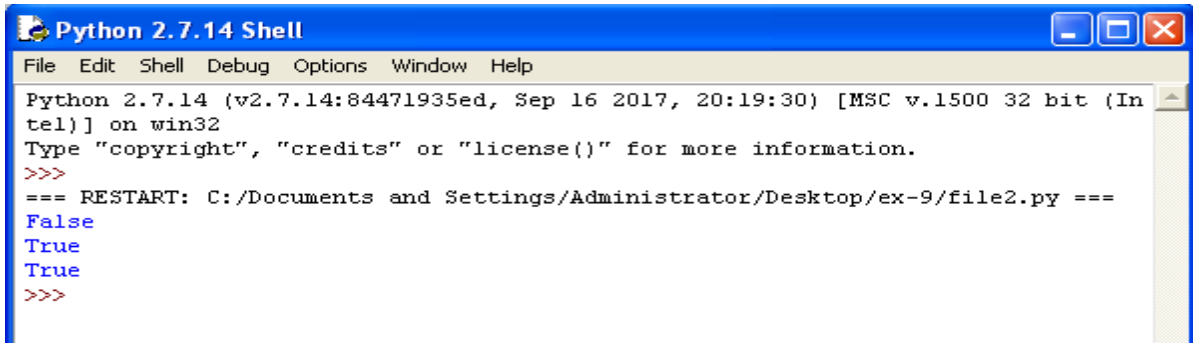
Question:

9a)Write a function nearly_equal to test whether two strings are nearly equal. Two strings a and b are nearly equal when a can be generated by a single mutation on b

Program:

```
def mutate(word):
    out_list = []
    letters = 'abcdefghijklmnopqrstuvwxyz'
    for i in range(len(word) + 1):
        for j in range(26):
            out_list.append(word[:i] + letters[j] + word[i:])
    for i in range(len(word)):
        out_list.append(word[:i] + word[i + 1:])
    for i in range(len(word)):
        for j in range(26):
            out_list.append(word[:i] + letters[j] + word[i + 1:])
    current_word = []
    out_word = ''
    for i in range(len(word) - 1):
        for j in range(i + 1, len(word)):
            current_word = []
            for symbol in word:
                current_word.append(symbol)
            current_word[i], current_word[j] = current_word[j], current_word[i]
            for symbol in current_word:
                out_word += symbol
            out_list.append(out_word)
            out_word = ''
    return out_list
def nearly_equal(word1, word2):
    return word1 in mutate(word2)
print nearly_equal('vasavi', 'sasi')
print nearly_equal('vasavi', 'vasaki')
print nearly_equal('see','sea')
```

Output:



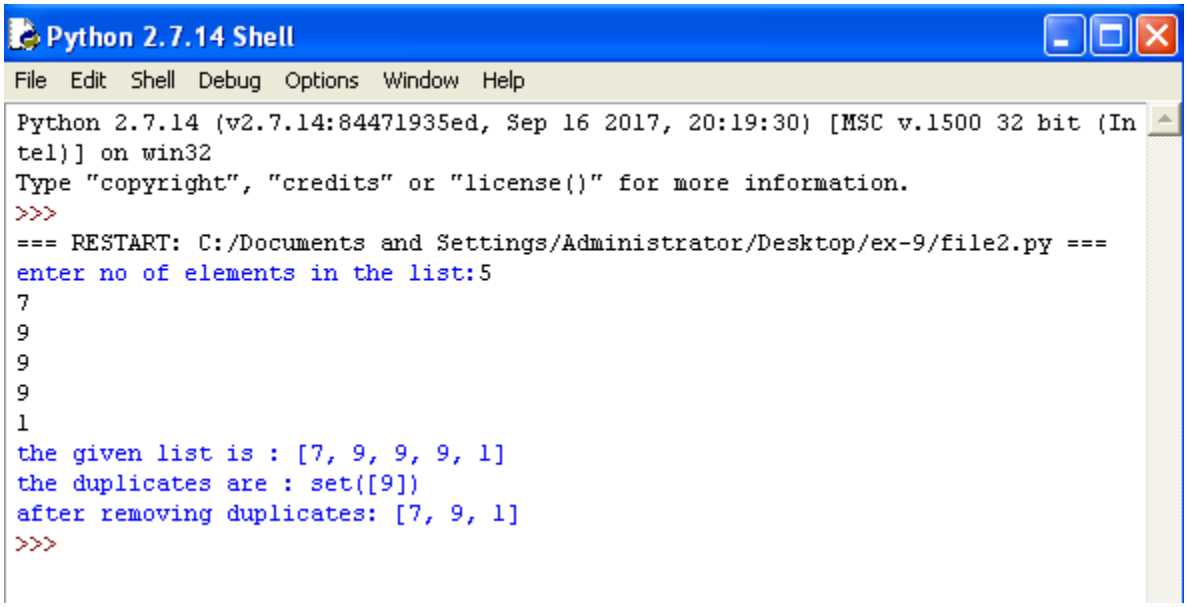
Question:

9b)Write a function dups to find all duplicates in the list

Program:

```
def dubs(l) :
    dup_s=set()
    for x in l:
        if l.count(x)>1 :
            dup_s.add(x)
            l.remove(x)
    print "the duplicates are :",dup_s
    print "after removing duplicates:",l
li=[]
n=input("enter no of elements in the list:")
for i in range(n) :
    e=input()
    li.append(e)
print "the given list is :",li
dubs(li)
```

Output:



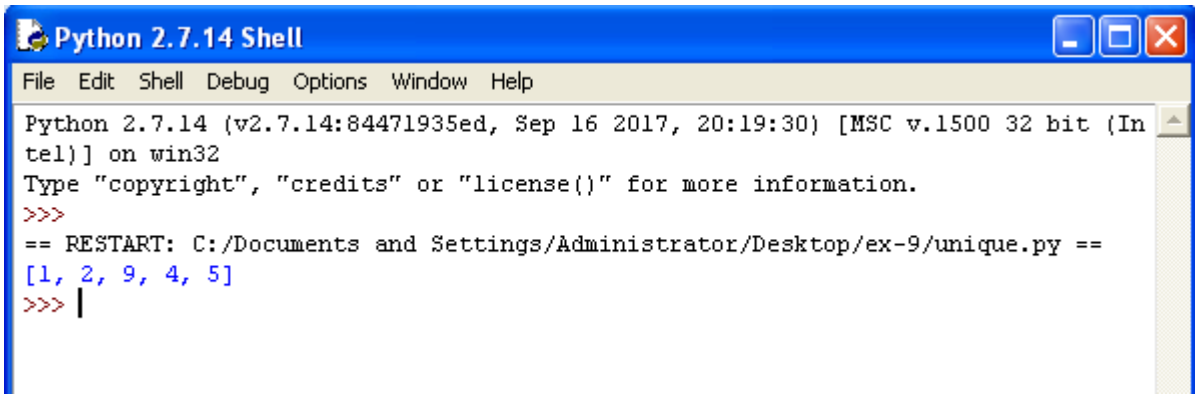
Question:

9c)Write a function unique to find all the unique elements of a list.

c

```
def unique_list(l):
  x = []
  for a in l:
    if a not in x:
      x.append(a)
  return x
print(unique_list([1,2,9,9,9,4,5]))
```

Output:


Python 2.7.14 Shell
File Edit Shell Debug Options Window Help
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Documents and Settings/Administrator/Desktop/ex-9/unique.py ==
[1, 2, 9, 4, 5]
>>>

Exercise - 10 - Functions - Problem Solving

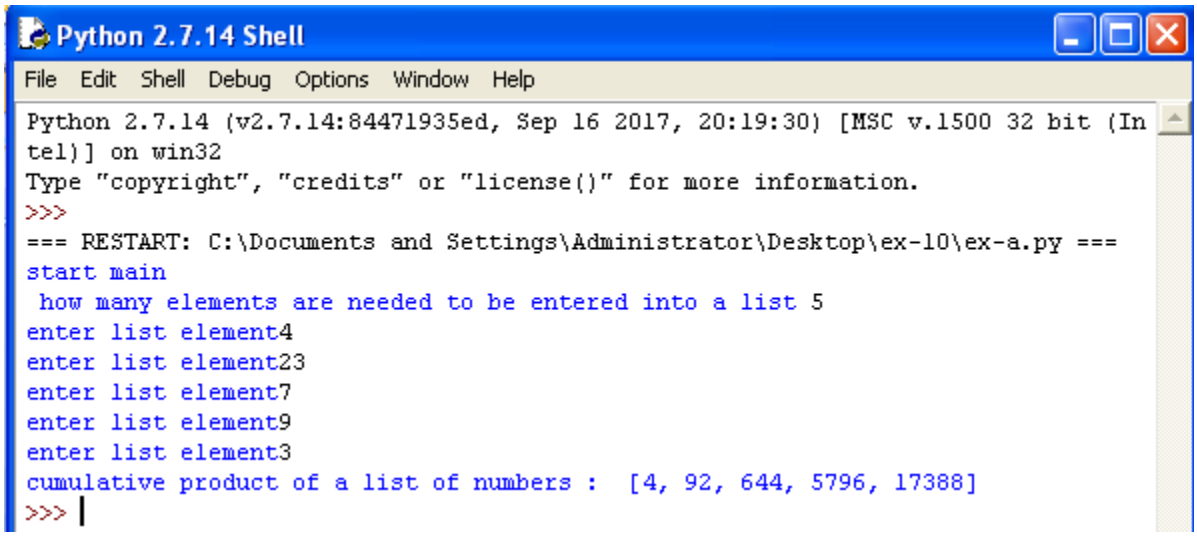
Question:

10a)Write a function cumulative_product to compute cumulative product of a list of numbers

Program:

```
def cproduct_list(mylist):
    product = 1
    cproduct = []
    for i in mylist:
        product = product * i
        cproduct.append(product)
    return cproduct
print "start main"
n=input(" how many elements are needed to be entered into a list")
mylist=[]
for i in range(0,n):
    l=input("enter list element")
    mylist.append(l)
print "cumulative product of a list of numbers : ",cproduct_list(mylist)
```

Output:



Question:

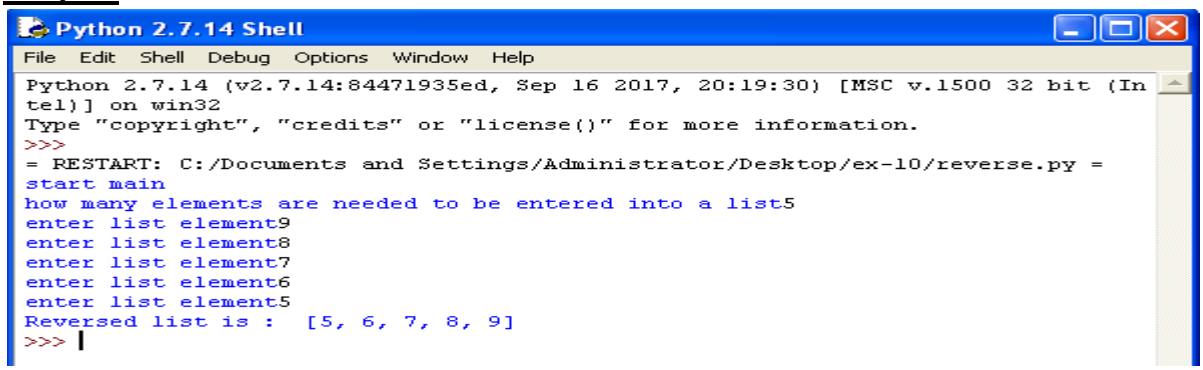
10b)Write a function reverse to reverse a list. Without using the reverse function

Program:

```
def reverse(list):
    reversed_list= []
    for i in range(1,len(list)+1):
        x = list[len(list)-i]
        reversed_list.append(x)
    return reversed_list
```

```
print "start main"
n=input("how many elements are needed to be entered into a list")
mylist=[]
for i in range(0,n):
    l=input("enter list element")
    mylist.append(l)
print "Reversed list is : ",reverse(mylist)
```

Output:



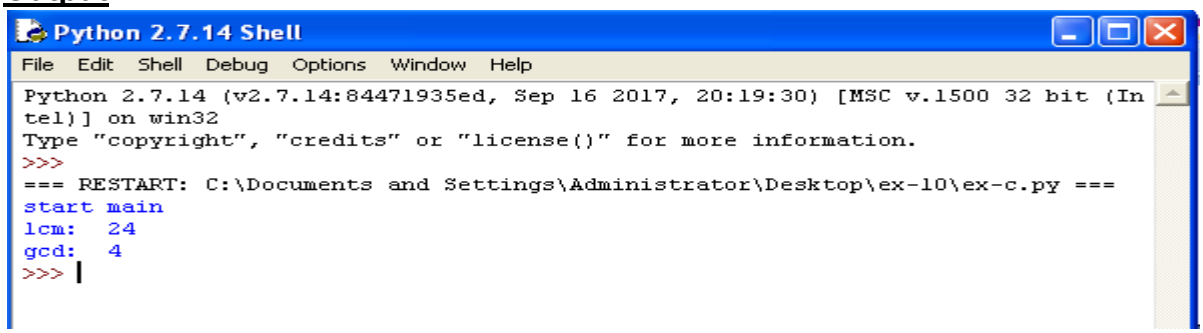
Question:

10c)Write function to compute gcd, lcm of two numbers. Each function shouldn't exceed one line.

Program:

```
def gcd(*numbers):
   from fractions import gcd
   return reduce(gcd, numbers)
def lcm(*numbers):
    def lcm(a, b):
        return (a * b) // gcd(a, b)
    return reduce(lcm, numbers, 1)
print "start main"
print "lcm: ",lcm(12,4,8)
print "gcd: ", gcd(12,4,8)
```

Output



Exercise 11 - Multi-D Lists

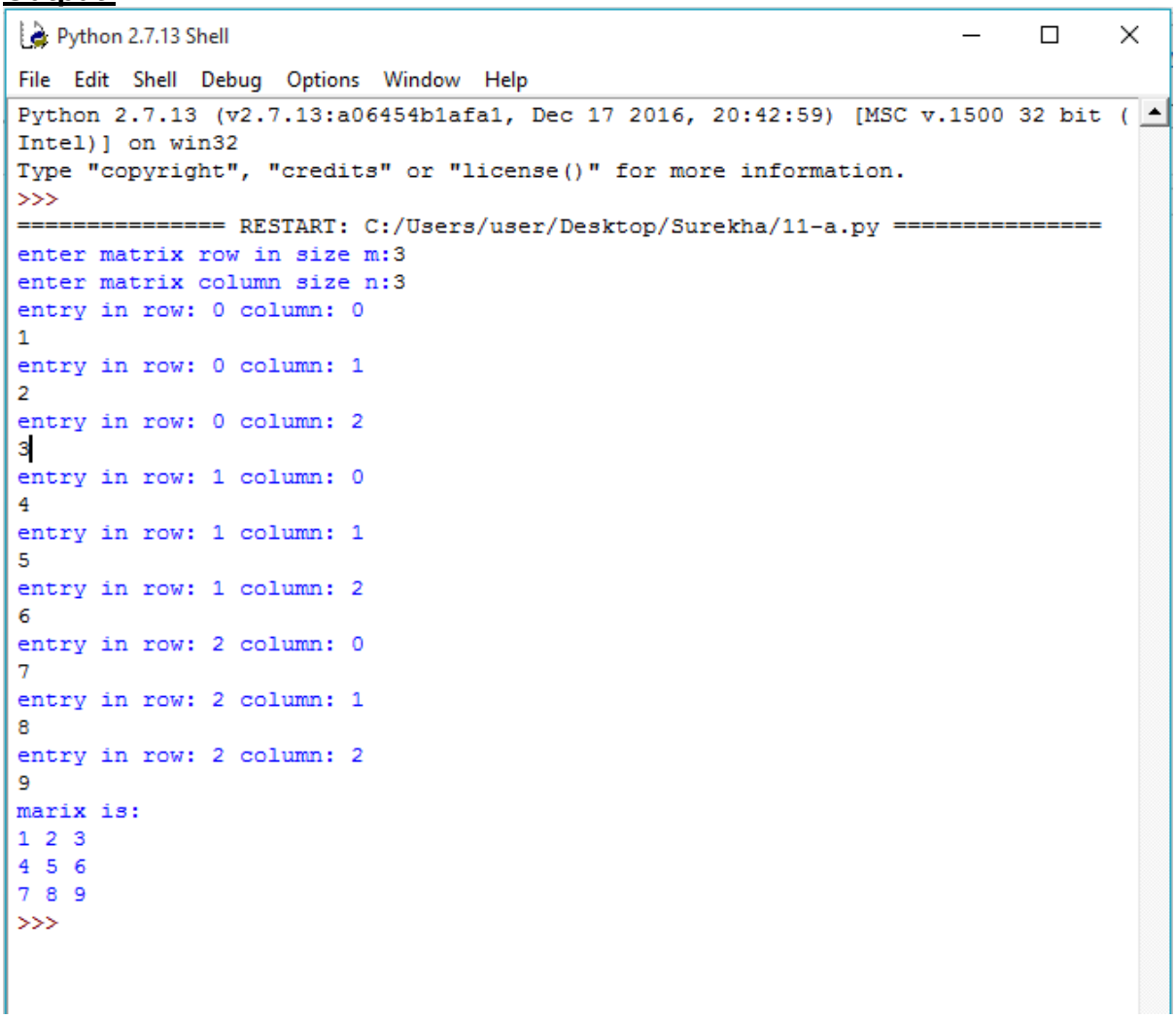
Question:

11a)Write a program that defines a matrix and prints

Program:

```
from __future__ import print_function
m=int(input("enter matrix row in size m:"))
n=int(input("enter matrix column size n:"))
x=[[0]*n for i in range(m)]
for i in range(m):
  for j in range(n):
    print("entry in row:",i,"column:",j)
    x[i][j]=int(input())
print("marix is:")
for i in range(m):
  for j in range(n):
    print(x[i][j],end=' ')
  print()
```

Output:



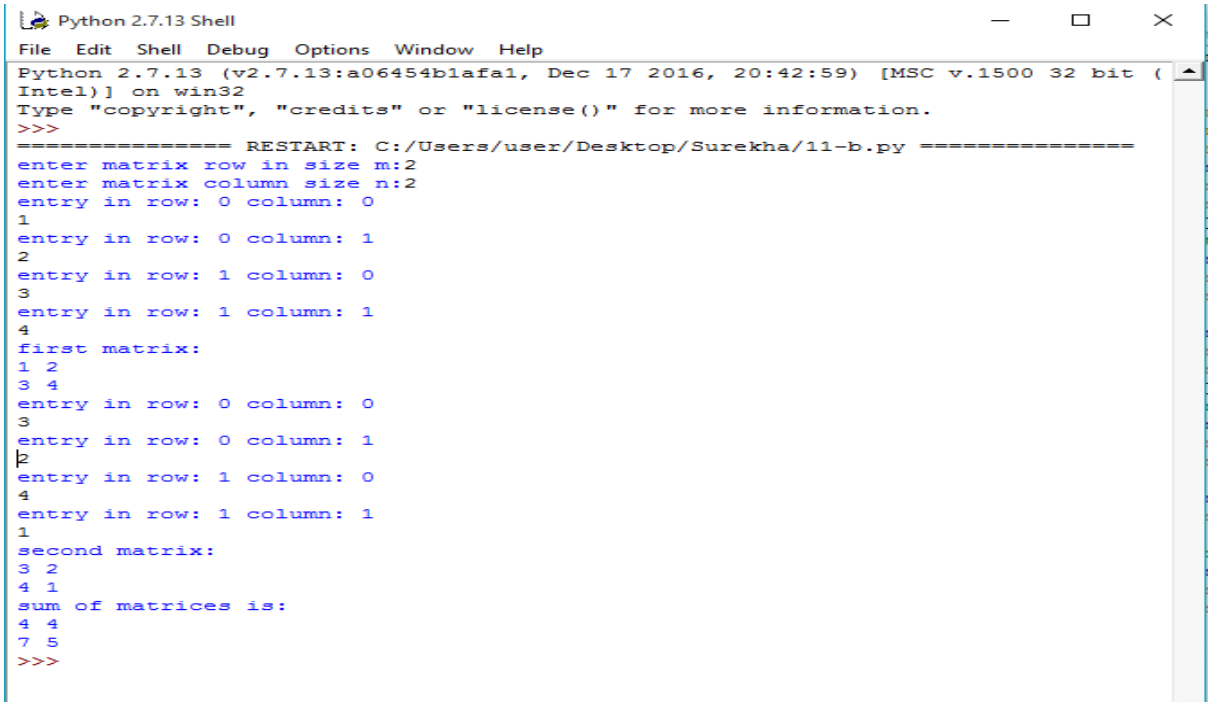
Question:

11b)Write a program to perform addition of two square matrices

Program:

```
from __future__ import print_function
m=int(input("enter matrix row in size m:"))
n=int(input("enter matrix column size n:"))
x=[[0]*n for i in range(m)]
y=[[0]*n for i in range(m)]
res=[[0]*n for i in range(m)]
for i in range(m):
  for j in range(n):
    print("entry in row:",i,"column:",j)
    x[i][j]=int(input())
print("first matrix:")
for i in range(m):
  for j in range(n):
    print(x[i][j],end=' ')
  print()
for i in range(m):
  for j in range(n):
    print("entry in row:",i,"column:",j)
    y[i][j]=int(input())
print("second matrix:")
for i in range(m):
  for j in range(n):
    print(y[i][j],end=' ')
  print()
for i in range(len(x)):
  for j in range(len(x[0])):
    res[i][j]=x[i][j]+y[i][j]
print("sum of matrices is:")
for i in range(m):
  for j in range(n):
    print(res[i][j],end=' ')
  print()
```

Output:


Python 2.7.13 Shell
File Edit Shell Debug Options Window Help
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/user/Desktop/Surekha/11-b.py ==========
enter matrix row in size m:2
enter matrix column size n:2
entry in row: 0 column: 0
1
entry in row: 0 column: 1
2
entry in row: 1 column: 0
3
entry in row: 1 column: 1
4
first matrix:
1 2
3 4
entry in row: 0 column: 0
3
entry in row: 0 column: 1
2
entry in row: 1 column: 0
4
entry in row: 1 column: 1
1
second matrix:
3 2
4 1
sum of matrices is:
4 4
7 5
>>>

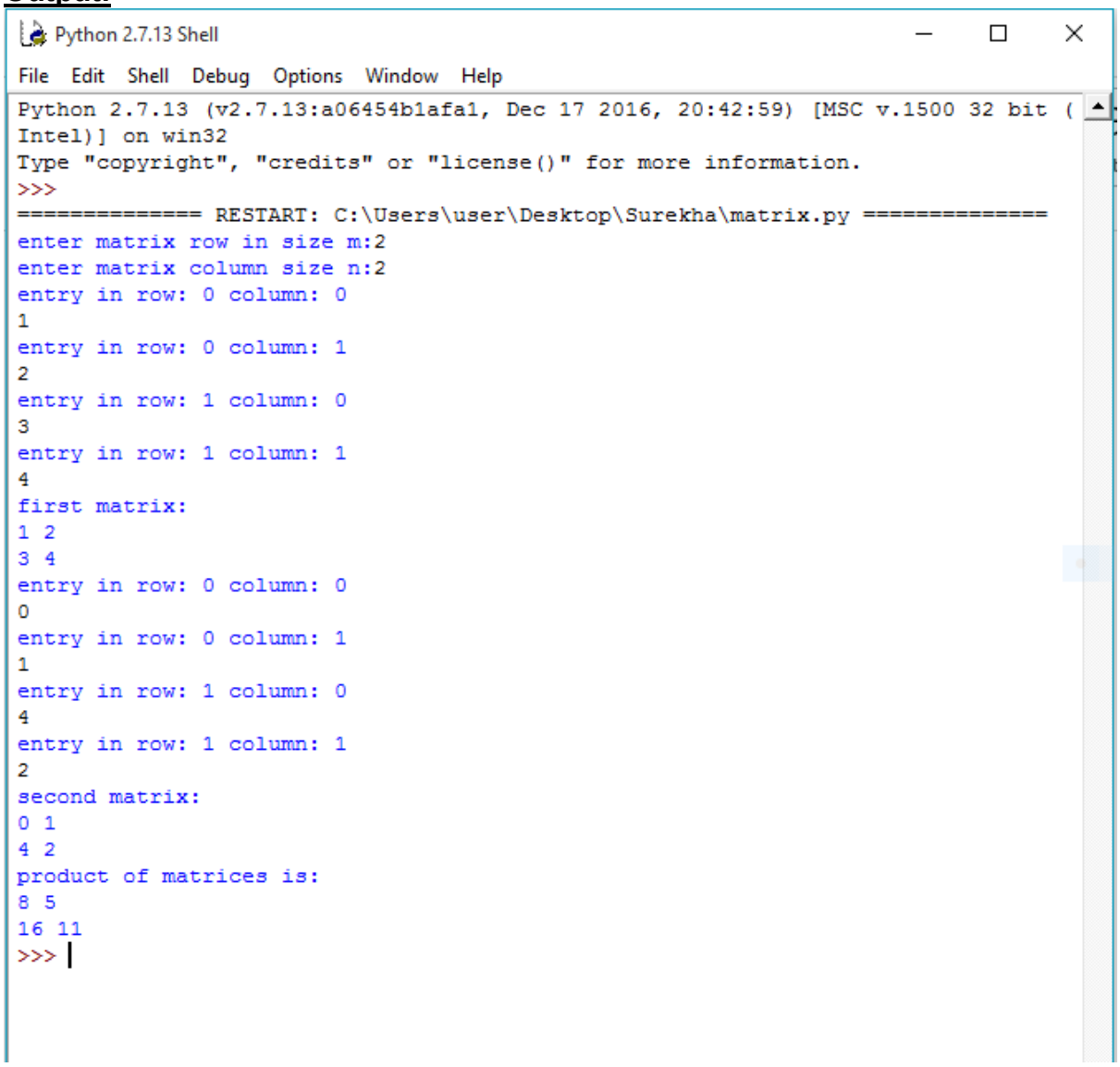
Question:

11c)Write a program to perform multiplication of two square matrices.

Program:

```
from __future__ import print_function
m=int(input("enter matrix row in size m:"))
n=int(input("enter matrix column size n:"))
x=[[0]*n for i in range(m)]
y=[[0]*n for i in range(m)]
res=[[0]*n for i in range(m)]
for i in range(m):
    for j in range(n):
        print("entry in row:",i,"column:",j)
        x[i][j]=int(input())
print("first matrix:")
for i in range(m):
    for j in range(n):
        print(x[i][j],end=' ')
    print()
for i in range(m):
    for j in range(n):
        print("entry in row:",i,"column:",j)
        y[i][j]=int(input())
print("second matrix:")
for i in range(m):
    for j in range(n):
        print(y[i][j],end=' ')
    print()
for i in range(len(x)):
    for j in range(len(x[0])):
        for k in range(len(x[0])):
            res[i][j]=res[i][j]+(x[i][k]*y[k][j])
print("product of matrices is:")
for i in range(m):
    for j in range(n):
        print(res[i][j],end=' ')
    print()
```

Output:


Python 2.7.13 Shell
File Edit Shell Debug Options Window Help
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\user\Desktop\Surekha\matrix.py ==========
enter matrix row in size m:2
enter matrix column size n:2
entry in row: 0 column: 0
1
entry in row: 0 column: 1
2
entry in row: 1 column: 0
3
entry in row: 1 column: 1
4
first matrix:
1 2
3 4
entry in row: 0 column: 0
0
entry in row: 0 column: 1
1
entry in row: 1 column: 0
4
entry in row: 1 column: 1
2
second matrix:
0 1
4 2
product of matrices is:
8 5
16 11
>>>

Exercise - 12 – Modules

Question:

12a)Install packages requests,flask and explore them using pip

Pip:

- Pip is a package management system(tool) used to install and manage software packages written in python
- Many packages can be found in the python package index(pyPI)
- Python 2.7.9 and late(on the python series) and python 3.4 and later include pip(pip3 for puthon3) by default
- One major advantage of pip is the ease of its command-line interface which makes installing pip software packages as eay as issuing one command
  Pip install some-package-name
  Users can also easily remove the package

Pip uninstall some-package-name

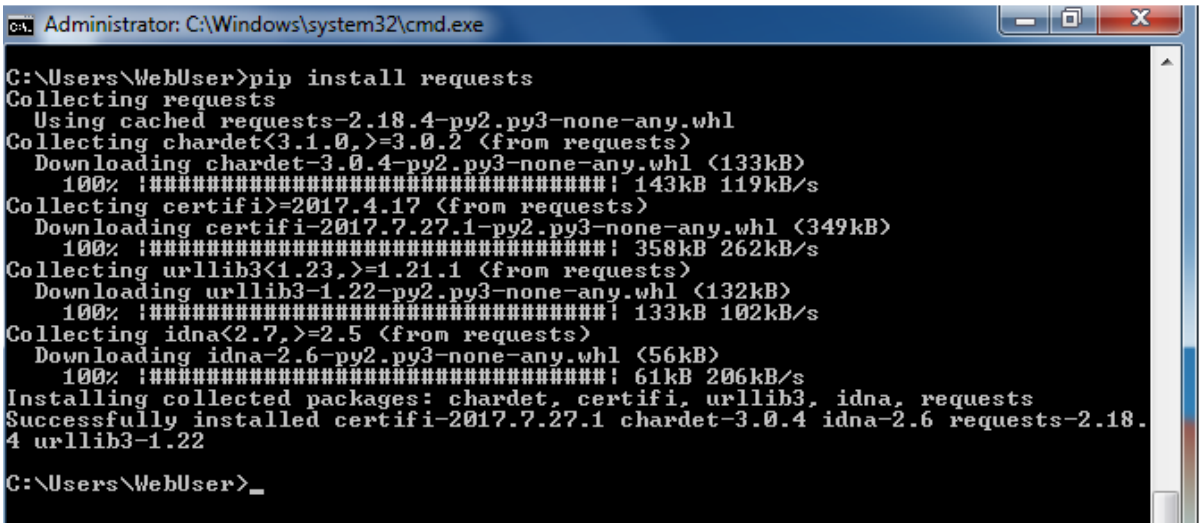
Downloading and installing pip:

Step-1:goto https://pip.pypa.io/en/stable/installing/
Step2:click on get-pip.py
Step3:save it on desktop with same name
Step4:double click on get-pip.py on your desktop
Step5:goto command prompt
Step6:type the following command for pip installed or not
pip

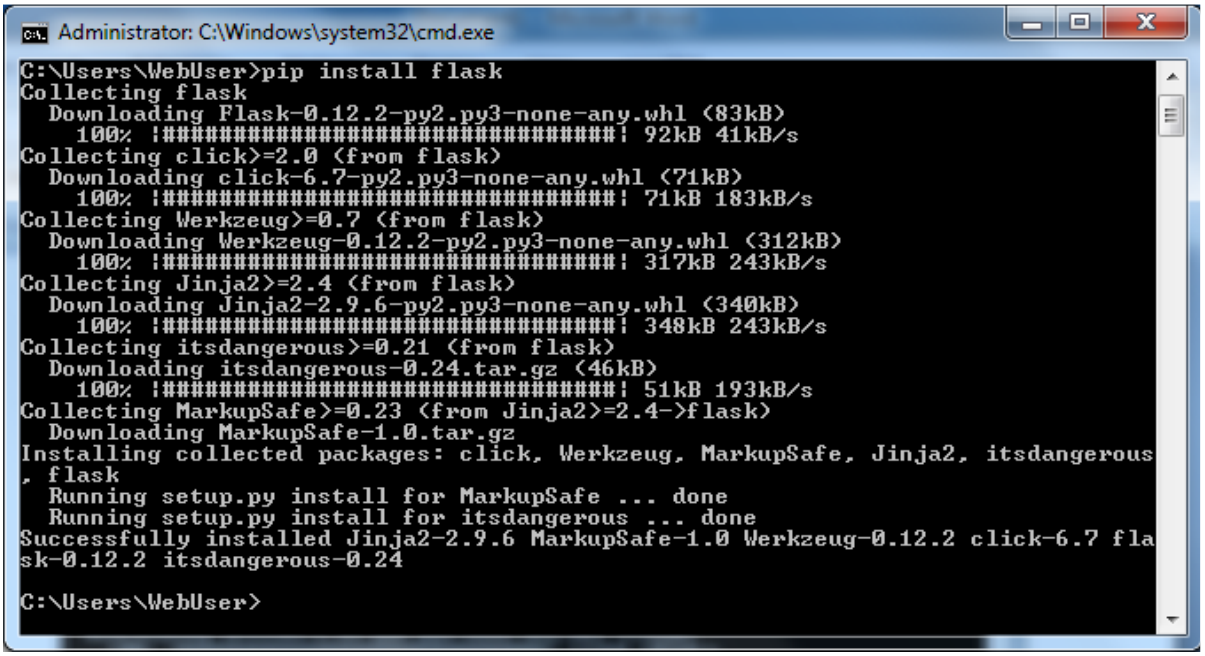
To install requests package
pip install requests
To install flask package
pip install flask

Output:

Install pip requests package



Install flask package



Question:

12b)Write a script that imports requests and fetch content from the page. Eg. (Wiki)

Installing Wikipedia:

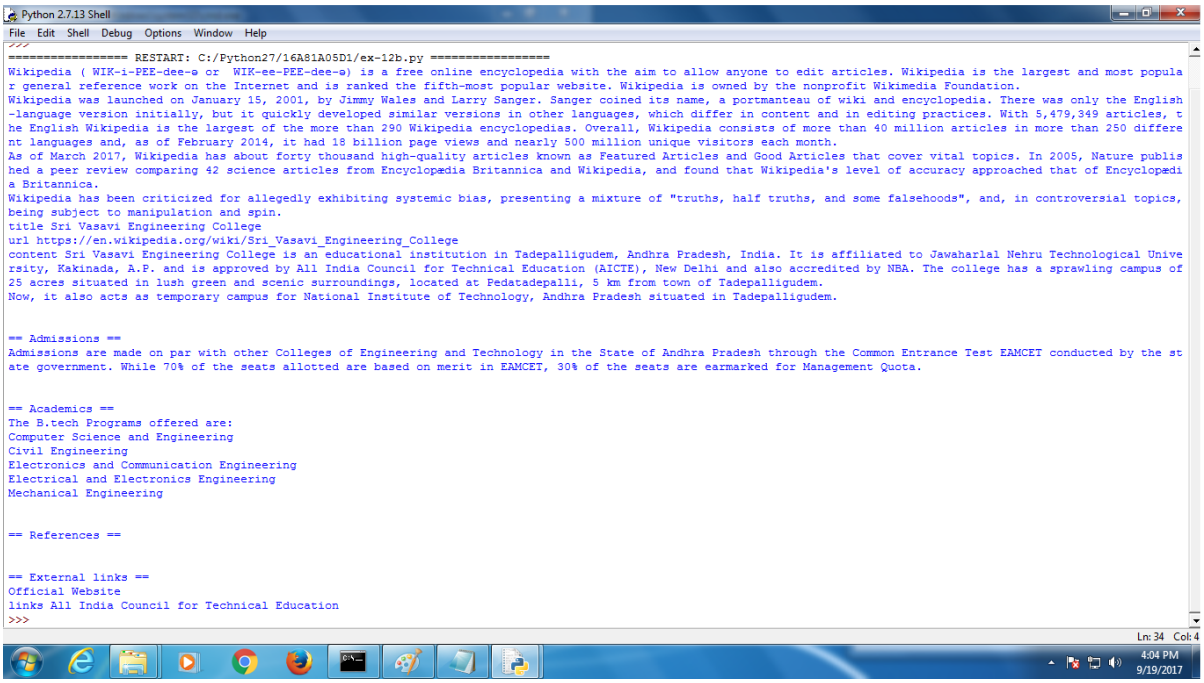
Step1:download Wikipedia 1.4.0 from https://pypi.python.org/pypi/wikipedia
Step2:place the Wikipedia module in python27/lib folder
Step3:install the Wikipedia module as follows:

- Open command prompt
- Type c:/python27
- Type pip install Wikipedia

Step4:type program as with the filename as wikiex.py

```
Wikiex.py
import wikipedia
print wikipedia.summary("wikipedia")
ny=wikipedia.page("sri_vasavi_engineering_college")
print "title",ny.title
print"url",ny.url
print "content",ny.content
    print "links",ny.links[0]
```

Output:



Question:

12c)Write a simple script that serves a simple HTTPResponse and a simple HTML Page

Program:

Server.py:

```
from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer
import os
class KodeFunHTTPRequestHandler(BaseHTTPRequestHandler):
    def do_GET(self):
        rootdir = 'c:/xampp/htdocs/'
        try:
            if self.path.endswith('.html'):
                f = open(rootdir + self.path)
                self.send_response(200)
                self.send_header('Content-type','text-html')
                self.end_headers()
                self.wfile.write(f.read())
                f.close()
                return
        except IOError:
            self.send_error(404, 'file not found')
    def run():
        print('http server is starting...')
        server_address = ('127.0.0.1', 80)
        httpd = HTTPServer(server_address, KodeFunHTTPRequestHandler)
        print('http server is running...')
        httpd.serve_forever()
if __name__ == '__main__':
    run()
```

client.py:

```
import httplib
import sys
http_server = sys.argv[1]
conn = httplib.HTTPConnection(http_server)
while 1:
    cmd = raw_input('input command (ex. GET index.html): ')
    cmd = cmd.split()
    if cmd[0] == 'exit':
        break
    conn.request(cmd[0], cmd[1])
    rsp = conn.getresponse()
    print(rsp.status, rsp.reason)
    data_received = rsp.read()
    print(data_received)
conn.close()
```

vasavi.html

```
<html>
        <head><title>CSE-C ROCKS</title></head>
        <body>
          <h1>sri vasavi engineering college</h1>
                <h2>pedatadepalli</h2>
                <h3>tadepalligudem</h3>
  </body>
<html>
```

HOW TO RUN SERVER PROGRAM

Step1:open command prompt to start the server
Step2:goto python27(c:/python27)
Step3:type command as follows
Python server.py

HOW TO RUN CLIENT PROGRAM

Step1:open command prompt
Step2:goto python27(c:/python27)
Step3:type the following command to execute the client program
Python client.py 127.0.0.1
Step4:type the following command to display the content
GET vasavi.html

Exercise - 13 OOP

Question:

13)Class variables and instance variable and illustration of the self variable

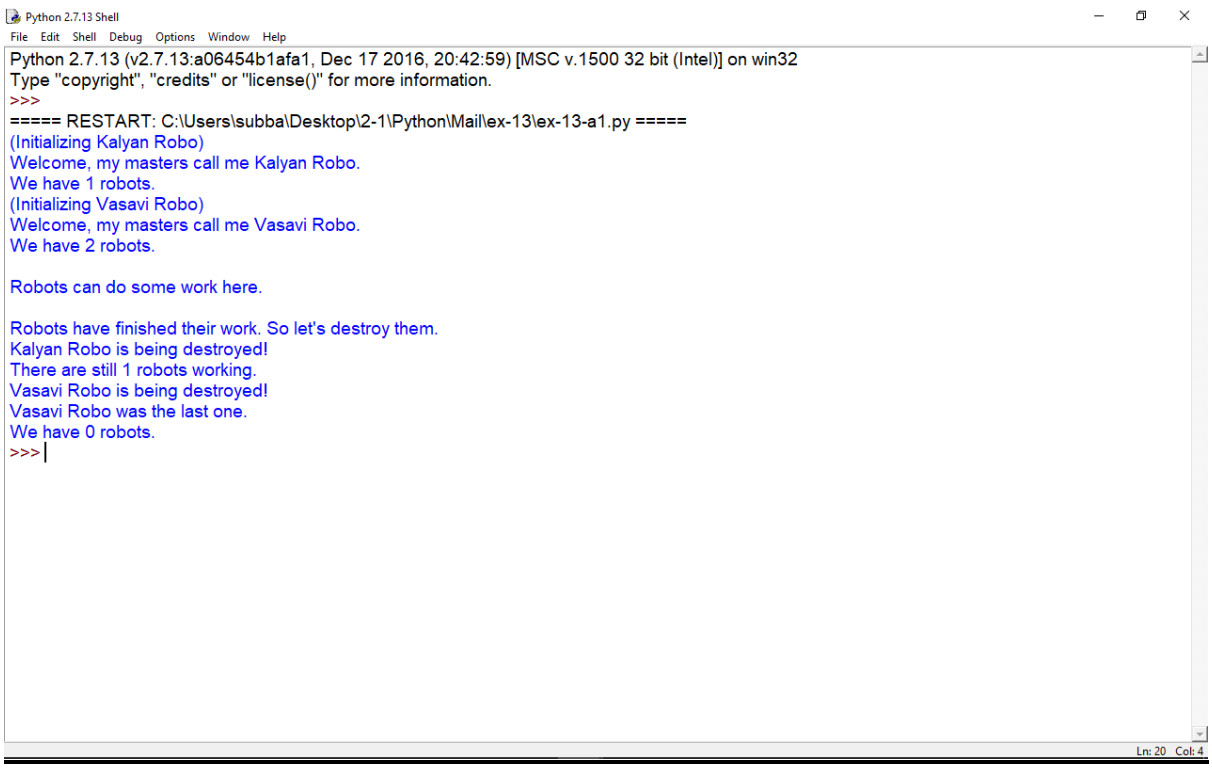
i) Robot

ii)) ATM Machine

i) Program1:Robot

```
class Robot:

    population = 0

    def __init__(self, name):
        self.name = name
        print('(Initializing {0})'.format(self.name))
        Robot.population += 1

    def __del__(self):
        print('{0} is being destroyed!'.format(self.name))
        Robot.population -= 1
        if Robot.population == 0:
            print('{0} was the last one.'.format(self.name))
        else:
            print('There are still {0:d} robots working.'.format(Robot.population))

    def welcome(self):
        print('Welcome, my masters call me {0}.'.format(self.name))

    def howMany():
        print('We have {0:d} robots.'.format(Robot.population))
    howMany = staticmethod(howMany)

droid1 = Robot('Kalyan Robo')
droid1.welcome()
Robot.howMany()

droid2 = Robot('Vasavi Robo')
droid2.welcome()
Robot.howMany()

print("\nRobots can do some work here.\n")

print("Robots have finished their work. So let's destroy them.")
del droid1
del droid2

Robot.howMany()
```

Output:

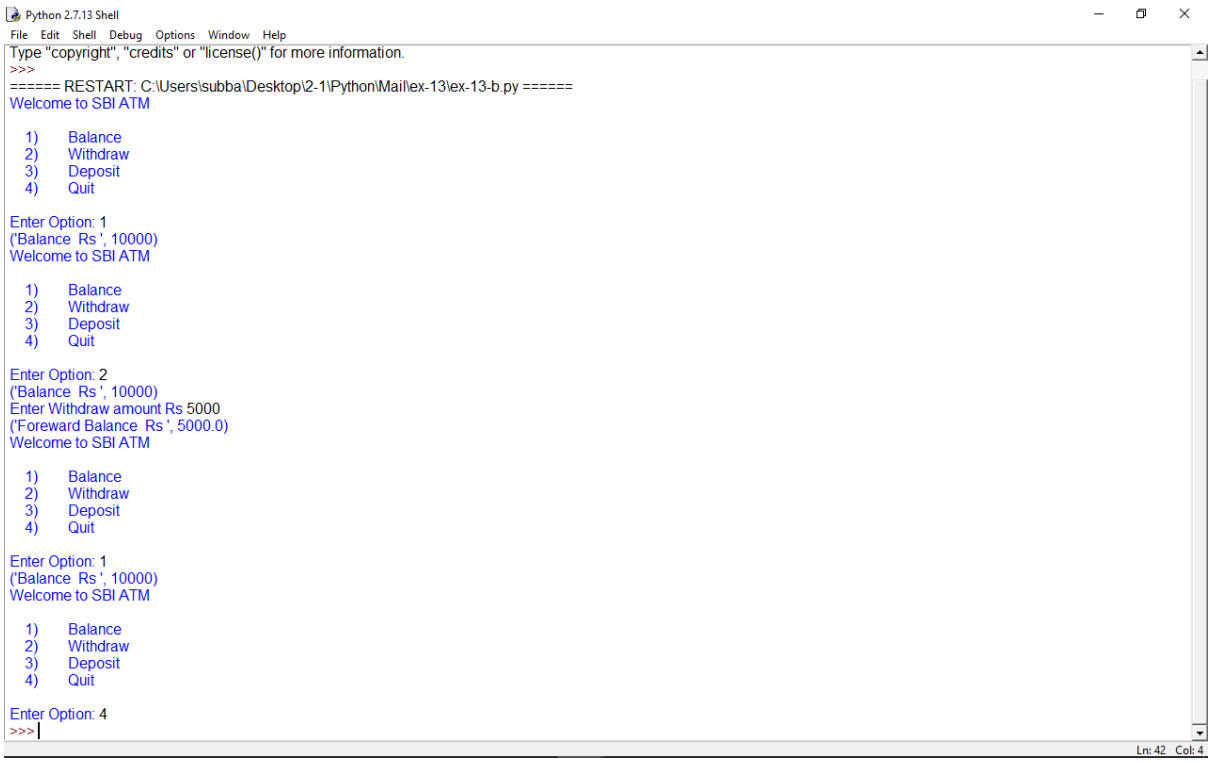


ii) ATM Machine

```
class ATM:
    balance=10000
    def showBalance(self):
        print("Balance  Rs ",ATM.balance)
    def withdraw(self):
        print("Balance  Rs ",ATM.balance)
        self.Withdraw=float(input("Enter Withdraw amount Rs "))
        if self.Withdraw>0:
            self.forewardbalance=(ATM.balance-self.Withdraw)
            print("Foreward Balance  Rs ",self.forewardbalance)
        elif self.Withdraw>ATM.balance:
            print("No funs in account")
        else:
            print("None withdraw made")
    def deposit(self):
         print("Balance  RS ",ATM.balance)
         self.Deposit=float(input("Enter deposit amount Rs "))
         if self.Deposit>0:
            self.forewardbalance=(ATM.balance+self.Deposit)
            print("Forewardbalance  Rs ",self.forewardbalance)
         else:
            print("None deposit made")
    def quit(self):
         exit()
```

```
while(True):
    print "Welcome to SBI ATM"

    print("""
    1)      Balance
    2)      Withdraw
    3)      Deposit
    4)      Quit
    """)
    Option=int(input("Enter Option: "))
    obj=ATM()
    if Option==1:
      obj.showBalance()
    if Option==2:
       obj.withdraw()
    if Option==3:
       obj.deposit()
    if Option==4:
       obj.quit()
```

Output:



Exercise - 14 GUI, Graphics

Question:

14a)Write a GUI for an Expression Calculator using tk

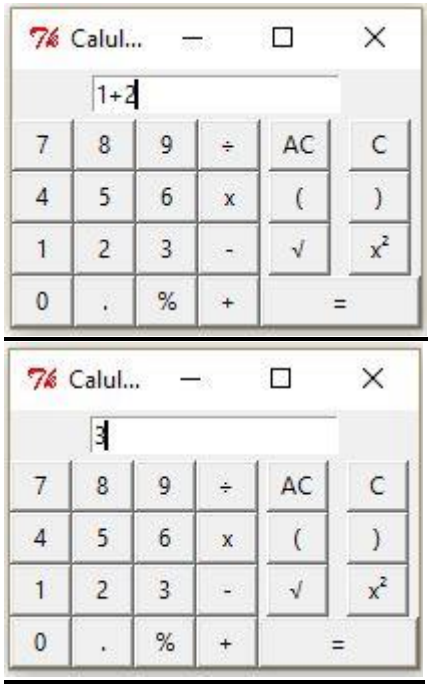
Program:

```
from Tkinter import *
import math

class calc:
 def getandreplace(self):
  """replace x with * and ÷ with /"""

  self.expression = self.e.get()
  self.newtext=self.expression.replace(self.newdiv,'/')
  self.newtext=self.newtext.replace('x','*')

 def equals(self):
  """when the equal button is pressed"""

  self.getandreplace()
  try:
   self.value= eval(self.newtext) #evaluate the expression using the eval function
  except SyntaxError or NameErrror:
   self.e.delete(0,END)
   self.e.insert(0,'Invalid Input!')
  else:
   self.e.delete(0,END)
   self.e.insert(0,self.value)

 def squareroot(self):
  """squareroot method"""

  self.getandreplace()
  try:
   self.value= eval(self.newtext) #evaluate the expression using the eval function
  except SyntaxError or NameErrror:
   self.e.delete(0,END)
   self.e.insert(0,'Invalid Input!')
  else:
   self.sqrtval=math.sqrt(self.value)
   self.e.delete(0,END)
   self.e.insert(0,self.sqrtval)

 def square(self):
  """square method"""

  self.getandreplace()
  try:
   self.value= eval(self.newtext) #evaluate the expression using the eval function
```

```
  except SyntaxError or NameErrror:
   self.e.delete(0,END)
   self.e.insert(0,'Invalid Input!')
  else:
   self.sqval=math.pow(self.value,2)
   self.e.delete(0,END)
   self.e.insert(0,self.sqval)

 def clearall(self):
  """when clear button is pressed,clears the text input area"""
  self.e.delete(0,END)

 def clear1(self):
  self.txt=self.e.get()[:-1]
  self.e.delete(0,END)
  self.e.insert(0,self.txt)
 def action(self,argi):
  """pressed button's value is inserted into the end of the text area"""
  self.e.insert(END,argi)

 def __init__(self,master):
  """Constructor method"""
  master.title('Calulator')
  master.geometry()
  self.e = Entry(master)
  self.e.grid(row=0,column=0,columnspan=6,pady=3)
  self.e.focus_set() #Sets focus on the input text area
  self.div='÷'
  self.newdiv=self.div.decode('utf-8')
  #Generating Buttons
  Button(master,text="=",width=10,command=lambda:self.equals()).grid(row=4,
column=4,columnspan=2)
  Button(master,text='AC',width=3,command=lambda:self.clearall()).grid(row=1,
column=4)
  Button(master,text='C',width=3,command=lambda:self.clear1()).grid(row=1, column=5)
  Button(master,text="+",width=3,command=lambda:self.action('+')).grid(row=4,
column=3)
  Button(master,text="x",width=3,command=lambda:self.action('x')).grid(row=2,
column=3)
  Button(master,text="-",width=3,command=lambda:self.action('-')).grid(row=3, column=3)
  Button(master,text="÷",width=3,command=lambda:self.action(self.newdiv)).grid(row=1,
column=3)
  Button(master,text="%",width=3,command=lambda:self.action('%')).grid(row=4,
column=2)
  Button(master,text="7",width=3,command=lambda:self.action('7')).grid(row=1,
column=0)
  Button(master,text="8",width=3,command=lambda:self.action(8)).grid(row=1, column=1)
  Button(master,text="9",width=3,command=lambda:self.action(9)).grid(row=1, column=2)
  Button(master,text="4",width=3,command=lambda:self.action(4)).grid(row=2, column=0)
  Button(master,text="5",width=3,command=lambda:self.action(5)).grid(row=2, column=1)
```

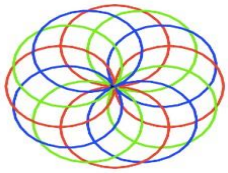
```
  Button(master,text="6",width=3,command=lambda:self.action(6)).grid(row=2, column=2)
  Button(master,text="1",width=3,command=lambda:self.action(1)).grid(row=3, column=0)
  Button(master,text="2",width=3,command=lambda:self.action(2)).grid(row=3, column=1)
  Button(master,text="3",width=3,command=lambda:self.action(3)).grid(row=3, column=2)
  Button(master,text="0",width=3,command=lambda:self.action(0)).grid(row=4, column=0)
  Button(master,text=".",width=3,command=lambda:self.action('.')).grid(row=4, column=1)
  Button(master,text="(",width=3,command=lambda:self.action('(')).grid(row=2, column=4)
  Button(master,text=")",width=3,command=lambda:self.action(')')).grid(row=2, column=5)
  Button(master,text="√",width=3,command=lambda:self.squareroot()).grid(row=3,
column=4)
  Button(master,text="x²",width=3,command=lambda:self.square()).grid(row=3, column=5)
#Main
root = Tk()
obj=calc(root) #object instantiated
root.mainloop()
```

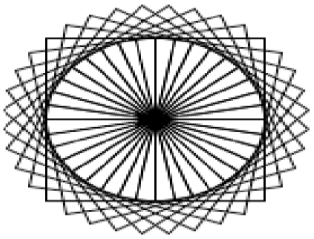
Output:



Question:

1. 14b)Write a program to implement the following figures using turtle

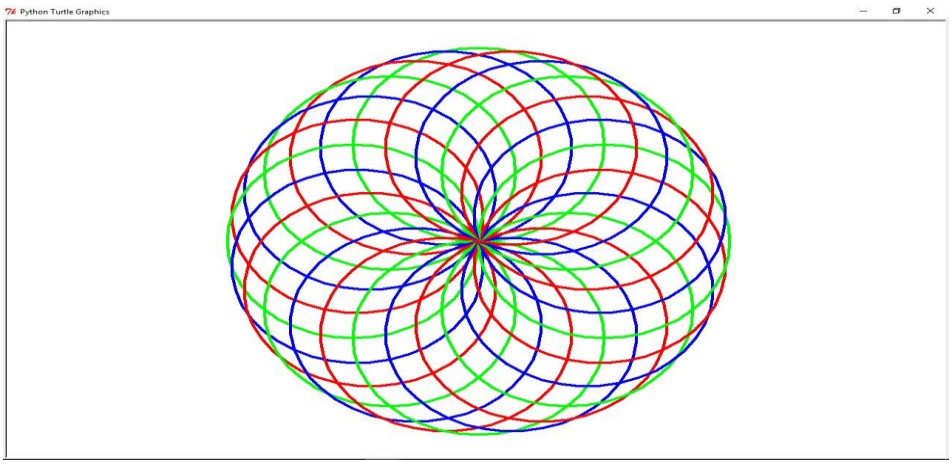




Program1:

```
from turtle import Turtle, Screen
from itertools import cycle
ANGLE = 15
colors = ["green", "blue", "red"]
def spiral(turtle, radius, color_names):
    colors = cycle(color_names)
    for i in range(360 // ANGLE):
        turtle.pensize(4)
        turtle.color(next(colors))
        turtle.circle(radius)
        turtle.left(ANGLE)
yertle = Turtle(visible=False)
yertle.speed("fastest")
spiral(yertle, 170, colors)
screen = Screen()
screen.exitonclick()
```

Output:

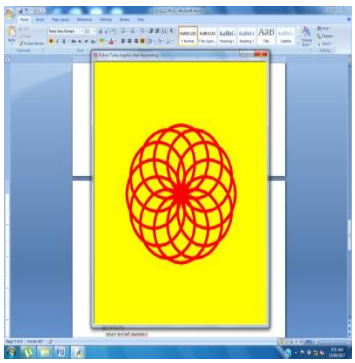


(**OR)**

Program1:

```
import turtle
turtle.bgcolor("yellow")
turtle.color("red")
turtle.pensize(10)
for angle in range(0,360,30):
    turtle.seth(angle)
    turtle.circle(100)
```

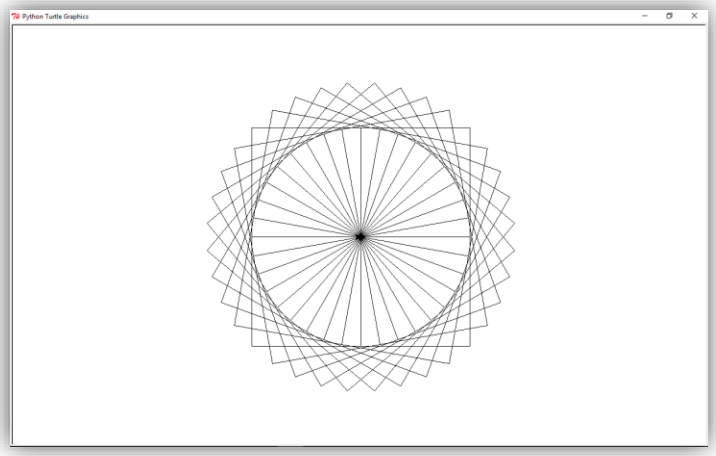
OUTPUT:



Program2:

```
import turtle
my_turtle = turtle.Turtle()
my_turtle.speed(0)
def square(length, degree):
    for i in range(4):
        my_turtle.forward(length)
        my_turtle.left(degree)
for i in range(36):
    square(200,90)
    my_turtle.left(10)
```

Output:



Exercise - 15 – Testing

Question:

15a)Write a test-case to check the function even_numbers which return True on passing a list of all even numbers

Program:

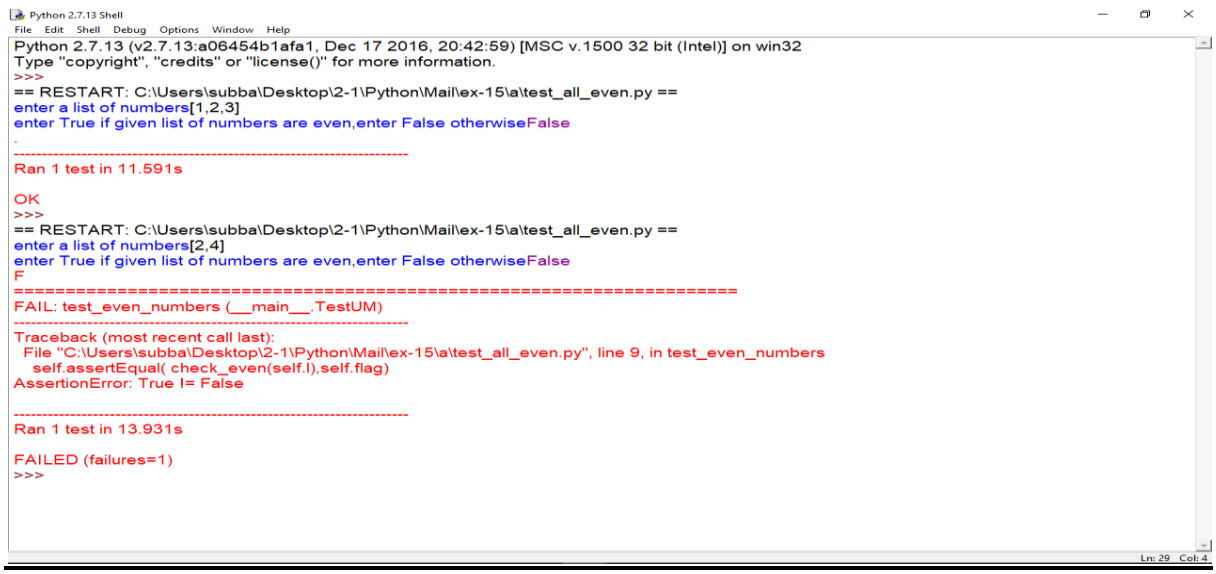
Evenno.py:

```
def check_even(numbers):
    for m in numbers:
        if m%2 == 0 :
            return True
        else :
            return False
```

test_all_even.py:

```
import unittest
from evenno import check_even

class TestUM(unittest.TestCase):

        def test_even_numbers(self) :
            self.l=input("enter a list of numbers")
            self.flag=input("enter True if given list of numbers are even,enter False otherwise")
            self.assertEqual( check_even(self.l),self.flag)
if __name__ == '__main__':
    unittest.main()
```

Output:



Question:

15b)Write a test-case to check the function reverse_string which returns the reversed string

Program:

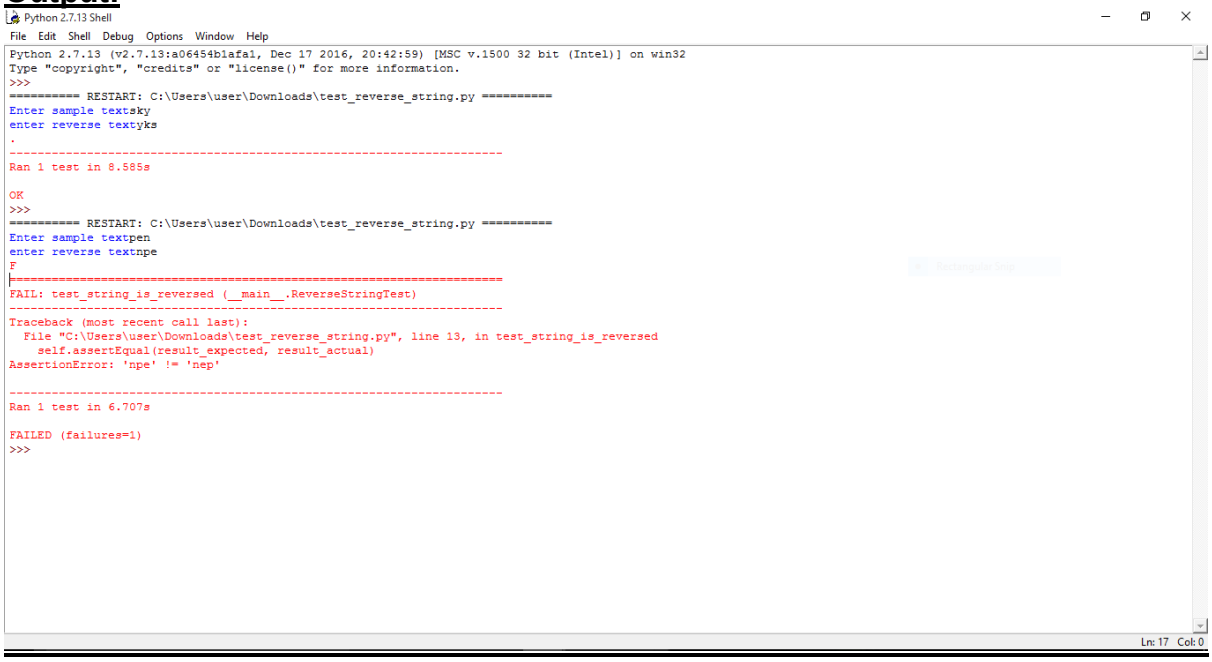
reverse_string.py:

```
def reverse(str_to_reverse):
    return str_to_reverse[::-1]
```

test_reverse_string.py:

```
from reverse_string import reverse

import unittest

class ReverseStringTest(unittest.TestCase):

    def test_string_is_reversed(self):

        test_str=raw_input("Enter sample text")

        result_expected=raw_input("enter reverse text")

        result_actual = reverse(test_str)

        self.assertEqual(result_expected, result_actual)

if __name__ == '__main__':

    unittest.main()
```

Output:



Exercise - 16 – Advanced

Question:

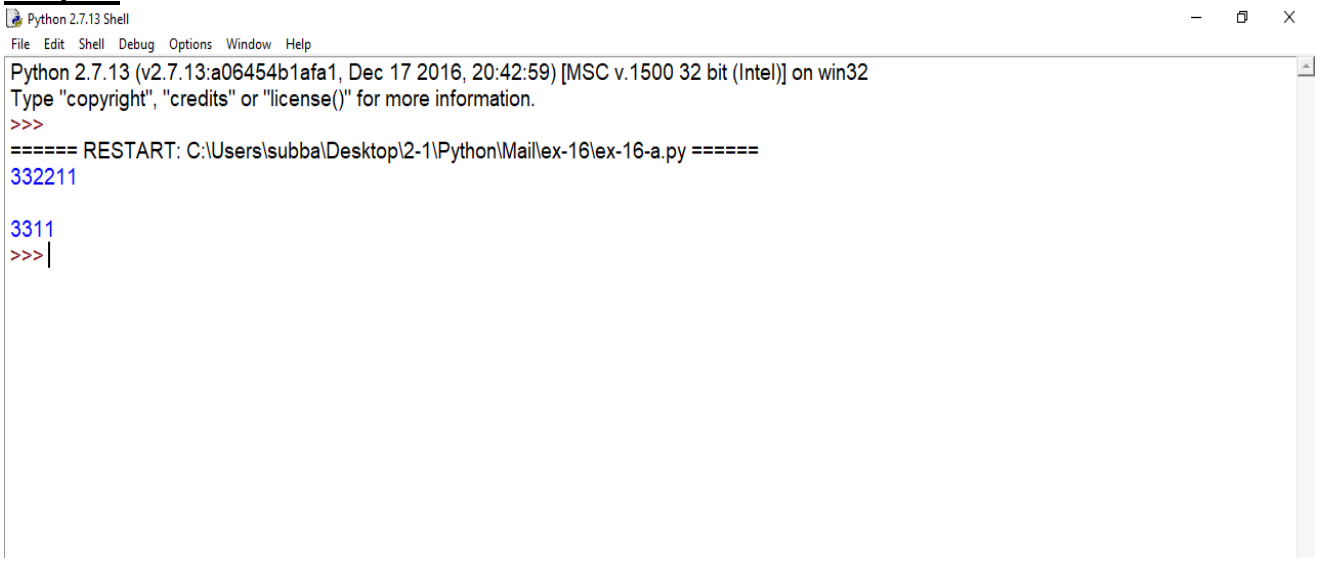
16a)Build any one classical data structure.

Program:

```
class Node :
  def __init__( self, data ) :
    self.data = data
    self.next = None
    self.prev = None

class LinkedList :
  def __init__( self ) :
    self.head = None

  def add( self, data ) :
    node = Node( data )
    if self.head == None :
      self.head = node
    else :
      node.next = self.head
      node.next.prev = node
      self.head = node

  def search( self, k ) :
    p = self.head
    if p != None :
      while p.next != None :
        if ( p.data == k ) :
          return p
        p = p.next
      if ( p.data == k ) :
        return p
    return None

  def remove( self, p ) :
    tmp = p.prev
    p.prev.next = p.next
    p.prev = tmp

  def __str__( self ) :
    s = ""
    p = self.head
    if p != None :
      while p.next != None :
        s += p.data
        p = p.next
      s += p.data
    return s
```

```
# example code
l = LinkedList()

l.add( '11' )
l.add('22')
l.add( '33' )

print l
l.remove( l.search('22') )
print
print l
```

Output:



Question:

b) Write a program to solve knapsack problem.

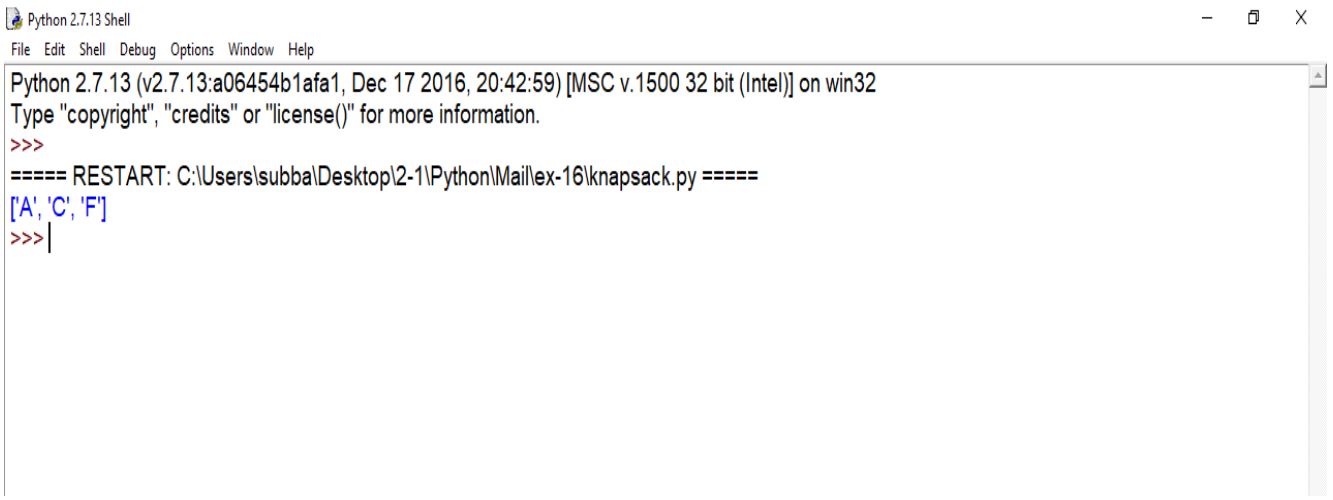
Program:

```
def itemSize(item): return item[0]
def itemValue(item): return item[1]
def itemName(item): return item[2]

exampleItems = [(3,3,'A'),
    (4,1,'B'),
    (8,3,'C'),
    (10,4,'D'),
    (15,3,'E'),
    (20,6,'F')]

exampleSizeLimit = 32

def pack1(items,sizeLimit):
  if len(items) == 0:
    return 0
  elif itemSize(items[-1]) > sizeLimit:
    return pack(items[:-1],sizeLimit)
  else:
```

```
    return max(pack(items[:-1],sizeLimit),
         pack(items[:-1],sizeLimit-itemSize(items[-1])) +
        itemValue(items[-1]))

def pack2(items,sizeLimit):
  def recurse(nItems,lim):
    if nItems == 0:
      return 0
    elif itemSize(items[nItems-1]) > lim:
      return recurse(nItems-1,lim)
    else:
      return max(recurse(nItems-1,lim),
           recurse(nItems-1,lim-itemSize(items[nItems-1])) +
          itemValue(items[nItems-1]))
  return recurse(len(items),sizeLimit)
def pack3(items,sizeLimit):
  P = {}

  def recurse(nItems,lim):
    if not P.has_key((nItems,lim)):
      if nItems == 0:
        P[nItems,lim] = 0
      elif itemSize(items[nItems-1]) > lim:
        P[nItems,lim] = recurse(nItems-1,lim)
      else:
        P[nItems,lim] = max(recurse(nItems-1,lim),
            recurse(nItems-1,lim-itemSize(items[nItems-1])) +
          itemValue(items[nItems-1]))
    return P[nItems,lim]

  return recurse(len(items),sizeLimit)

def pack4(items,sizeLimit):
  P = {}
  for nItems in range(len(items)+1):
    for lim in range(sizeLimit+1):
      if nItems == 0:
        P[nItems,lim] = 0
      elif itemSize(items[nItems-1]) > lim:
        P[nItems,lim] = P[nItems-1,lim]
      else:
        P[nItems,lim] = max(P[nItems-1,lim],
            P[nItems-1,lim-itemSize(items[nItems-1])] +
          itemValue(items[nItems-1]))
  return P[len(items),sizeLimit]
def pack5(items,sizeLimit):
  P = {}
  for nItems in range(len(items)+1):
    for lim in range(sizeLimit+1):
      if nItems == 0:
```

```
        P[nItems,lim] = 0
      elif itemSize(items[nItems-1]) > lim:
        P[nItems,lim] = P[nItems-1,lim]
      else:
        P[nItems,lim] = max(P[nItems-1,lim],
           P[nItems-1,lim-itemSize(items[nItems-1])] +
          itemValue(items[nItems-1]))

  L = []
  nItems = len(items)
  lim = sizeLimit
  while nItems > 0:
    if P[nItems,lim] == P[nItems-1,lim]:
      nItems -= 1
    else:
      nItems -= 1
      L.append(itemName(items[nItems]))
      lim -= itemSize(items[nItems])

  L.reverse()
  return L

print pack5(exampleItems,exampleSizeLimit)
```

Output:



ADD-ON EXPERIMENTS

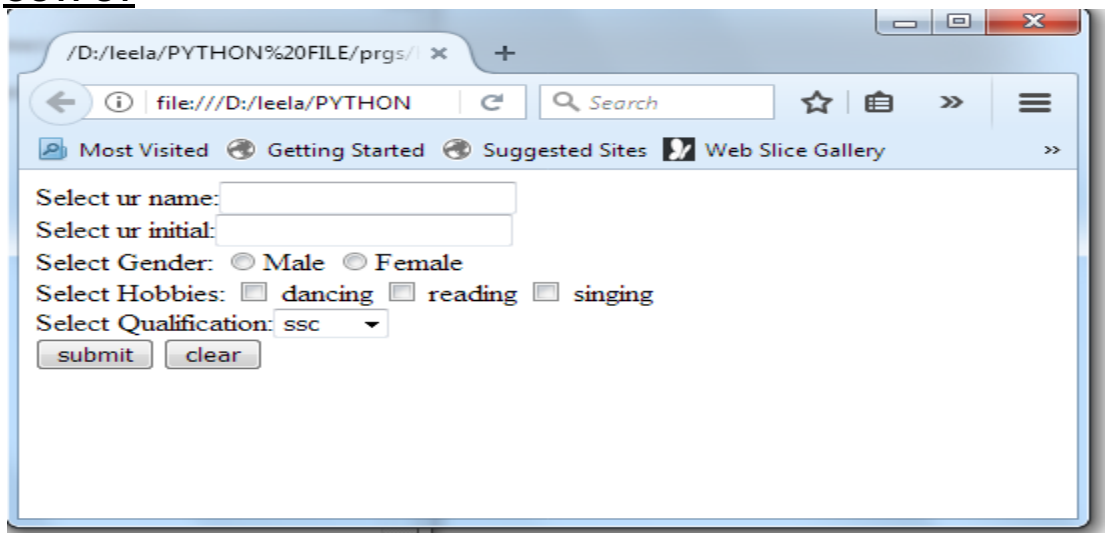
5.Add-on Experiments

1) Write a simple script to display a simple HTML Page for APPLICATION FORM.
2) Swapping of 2 numbers with parameterized Constructors.

1) Program: Write a simple script to display a simple HTML Page for APPLICATION FORM

```
import webbrowser
f = open('hello2.html','w')
message = """<html>
<body>
<form action="">
Select ur name:<input type="text" name="userid">
<br>
Select ur initial:<input type="text" name="usersurname">
<br>
Select Gender:
<input type="radio">Male
<input type="radio">Female
<br>
Select Hobbies:
<input type="checkbox"> dancing
<input type="checkbox"> reading
<input type="checkbox"> singing<br>
Select Qualification:<select>
<option>ssc</option>
<option>inter</option>
<option>btech</option>
</select><br>
<input type="Submit" name=b1 value="submit">
<input type="Submit" name=b1 value="clear">
</form>
</body>
</html>"""
f.write(message)
f.close()
webbrowser.open_new_tab('hello2.html')
```

OUTPUT



2) Swapping of 2 numbers with parameterized Constructors:

```
class MyClass:
  def __init__(self,var1,var2):
    self.a=var1
    self.b=var2
    print("before swap")
    print(self.a,self.b)
  def swapping(self):
    global t
    t=self.a
    self.a=self.b
    self.b=t
    print("after swap")
    print(self.a,self.b)
m=MyClass(10,20)
m.swapping()
```

OUTPUT

```
before swap
(10, 20)
after swap
(20, 10)
```

REFERENCES

6.References

1. Python Programming: A Modern Approach, Vamsi Kurama, Pearson

2. Learning Python, Mark Lutz, Orielly

3.Python Programming,Problem Solving Approach,Oxford Publications-Reema Thareja

4. www.tutorialspoint.com

5. https://www.programiz.com/python-programming/examples

6. http://www.sanfoundry.com/python-problems-solutions/