**14.8. Python Assignment Problems**

1. Write a Python program to find whether a given number (accept from the user) is prime or not,print out an appropriate message to the user.

**Code:**

x=int(raw\_input("Enter a number: "))

if(x<1):

print "Please, Enter a number greater than 1"

elif (x<=3):

print "Prime"

for i in range(2,x/2):

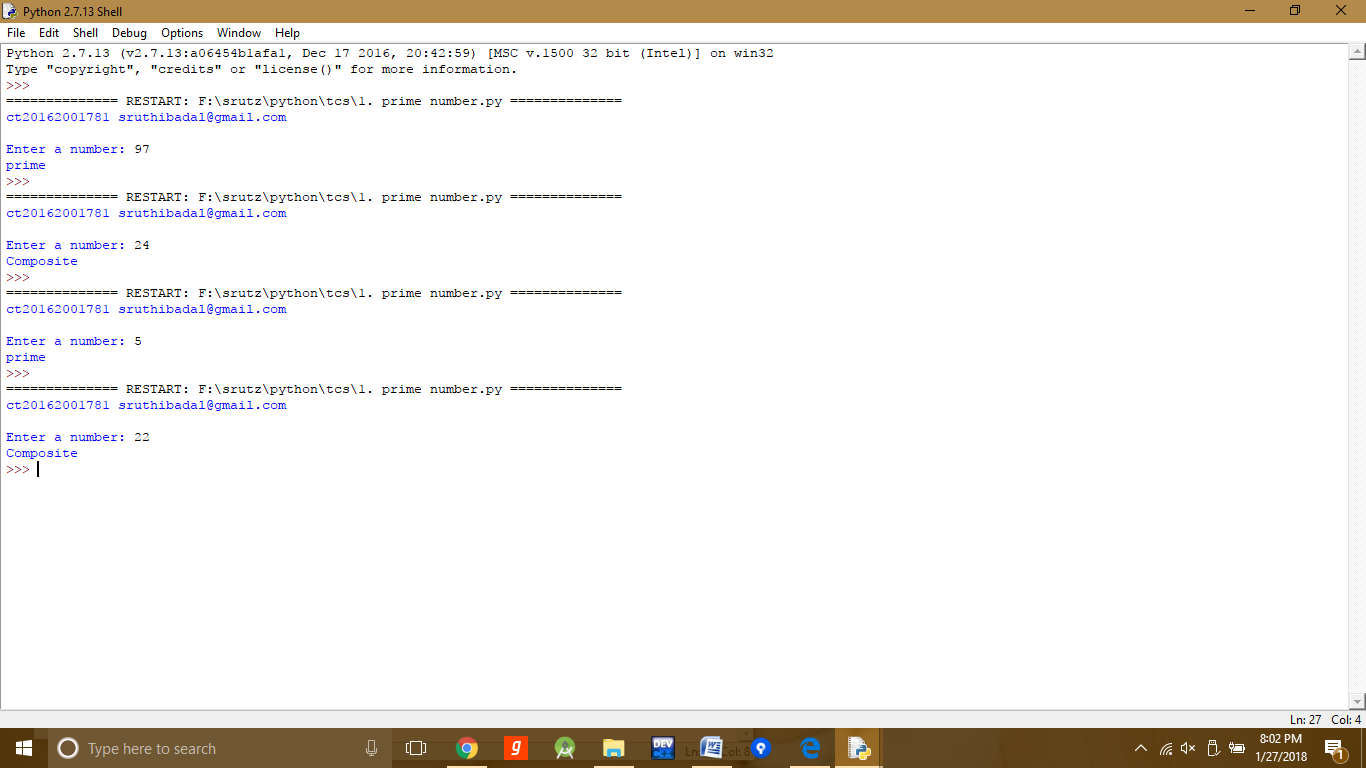
if(x%i==0):

print "Composite"

break;

else:

print "prime"

**Output:** 

2. Write a Python program which accept the radius of a circle from the user and compute the area.

**Code:**

import sys

pi=3.14

radius=float(raw\_input("Enter the radius: "))

if(radius<0.00):

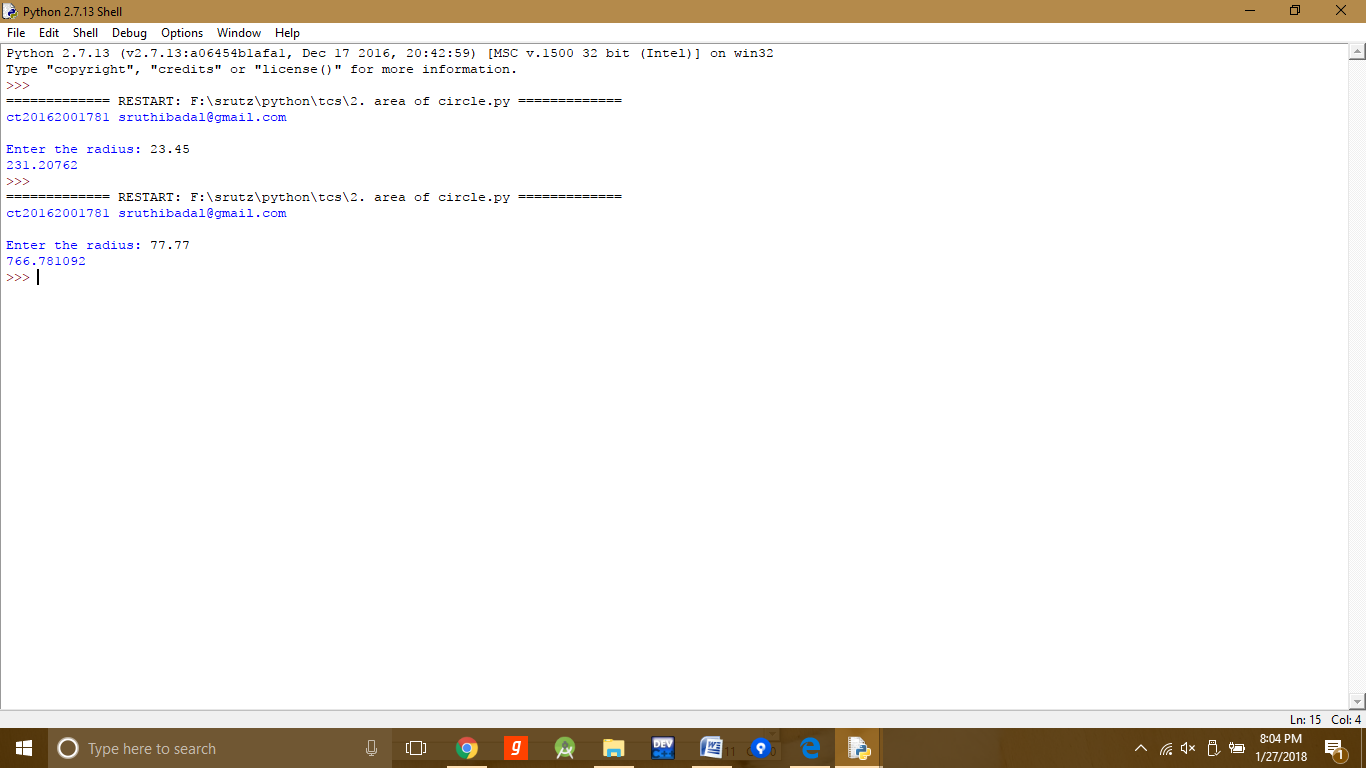
print("Please Enter a positive number")

sys.exit(0)

area=pi\*pi\*radius

print area

**Output:**

****

3. Write a Python program to print all even numbers from a given list of numbers. The list is terminated by -99999.

**Code:**

print "Enter a number, to end enter -99999 "

even\_numbers=" "

while True:

num=int(raw\_input())

if(num==-99999):

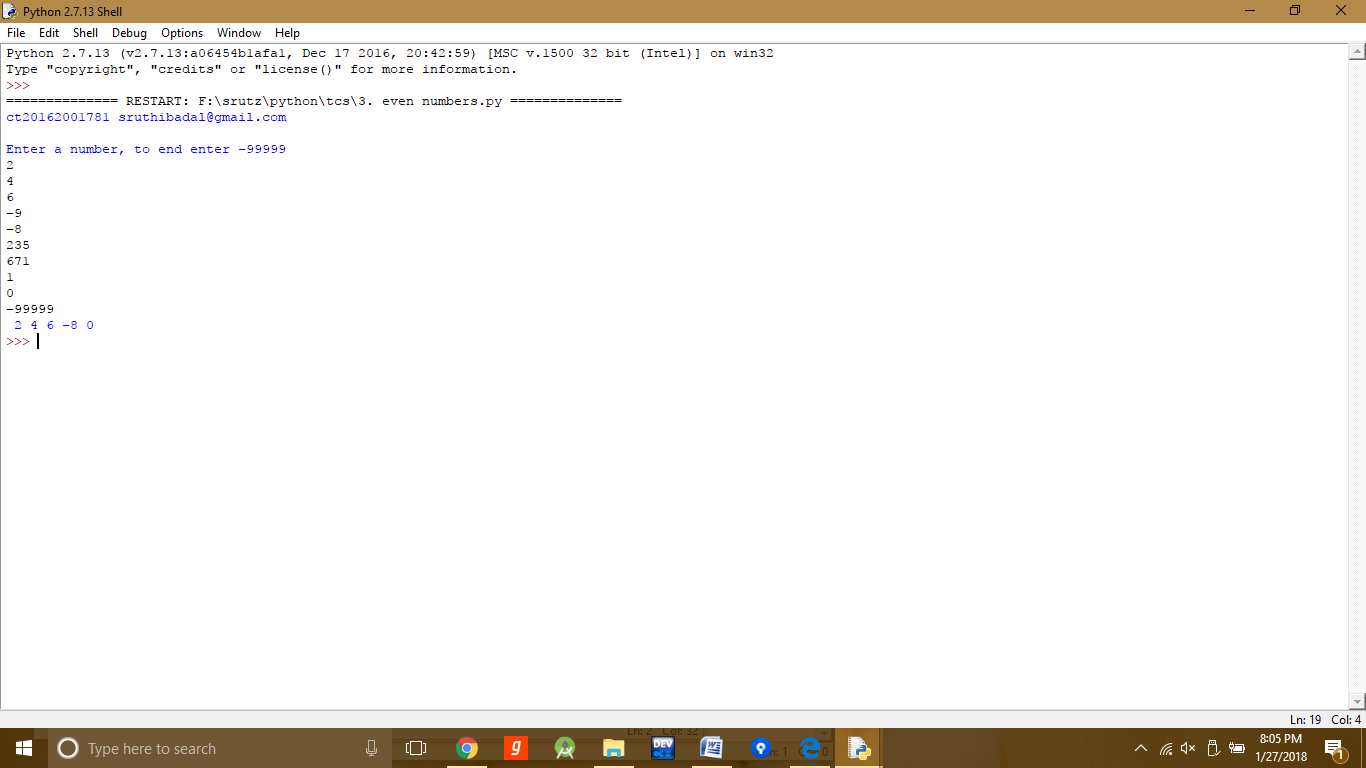
break;

if(num%2==0):

even\_numbers+=str(num)+" "

print even\_numbers

**Output:**

****

4. Write a Python program to find the occurrence of a given number in a given list of numbers.

**Code:**

def occurrence(l,num):

if num in l:

print "Yes, Present"

else:

print "No, Not Present"

def get\_list():

print "Enter a number, to end enter -99999 "

numbers=[]

while True:

num=int(raw\_input())

if(num==-99999):

break;

numbers.append(num)

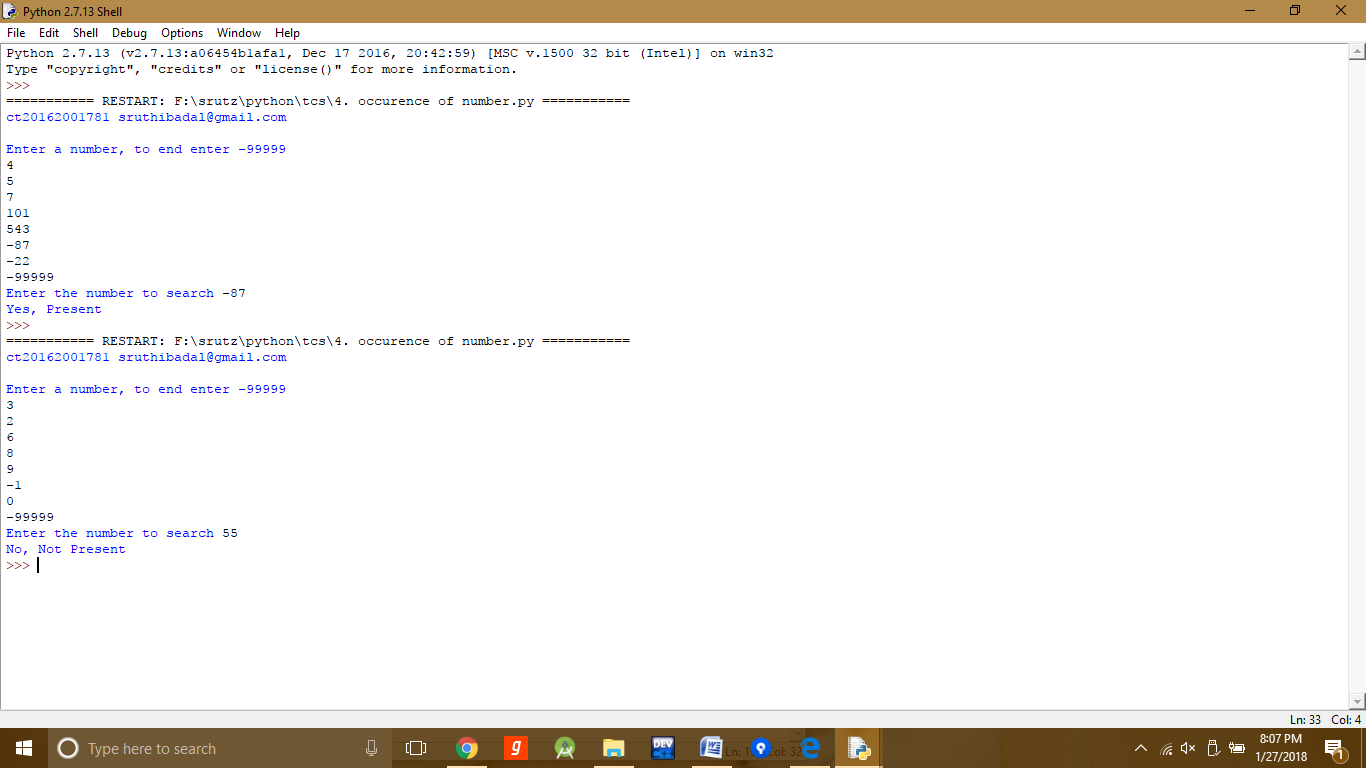
return numbers

l=get\_list()

num=int(raw\_input("Enter the number to search "))

occurrence(l,num)

**Output:**

****

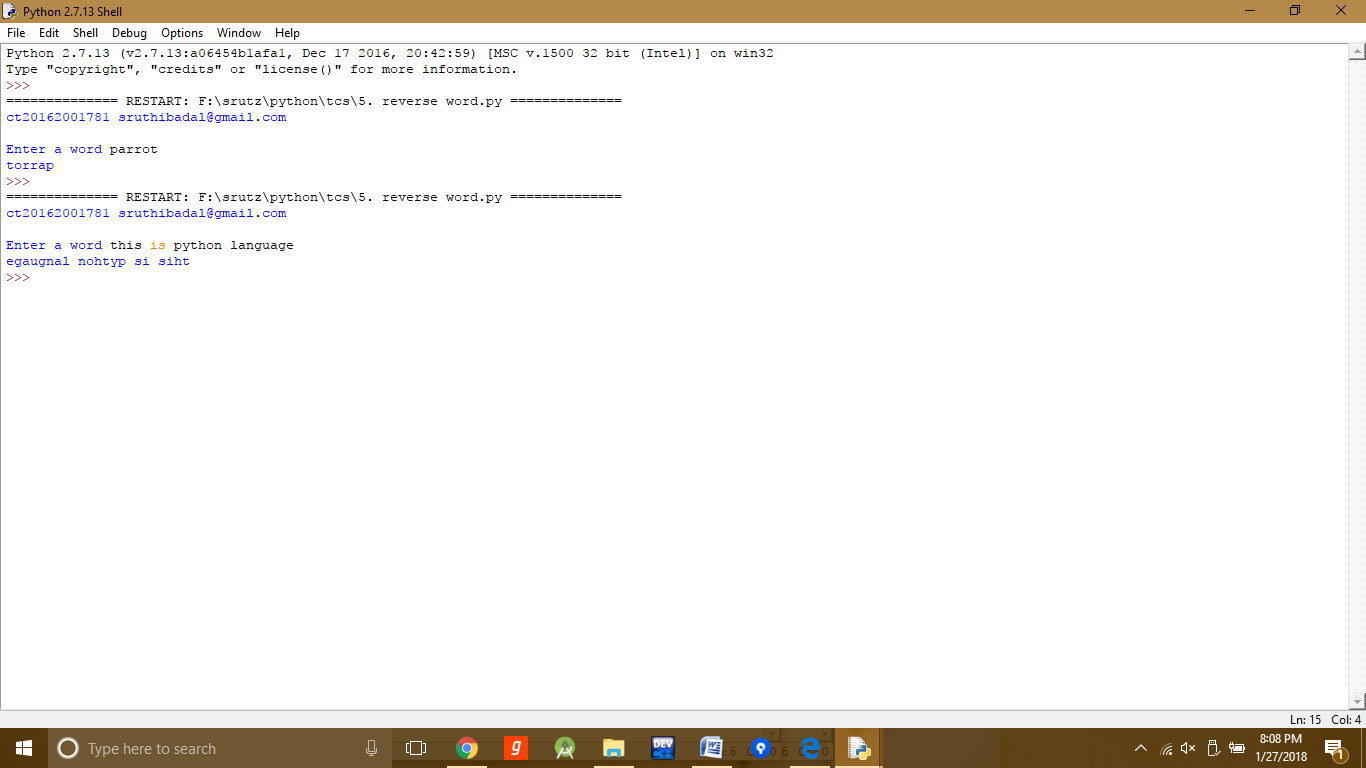
5. Write a Python program that accept a word from the user and reverse it.

**Code:**

word=raw\_input("Enter a word")

print word[::-1]

Output:



6. Write a Python program that accept a number and prints the reverse of it.

**Code:**

def reverse\_num(num):

rev=0;

while num>0:

n=num%10

rev=rev\*10+n

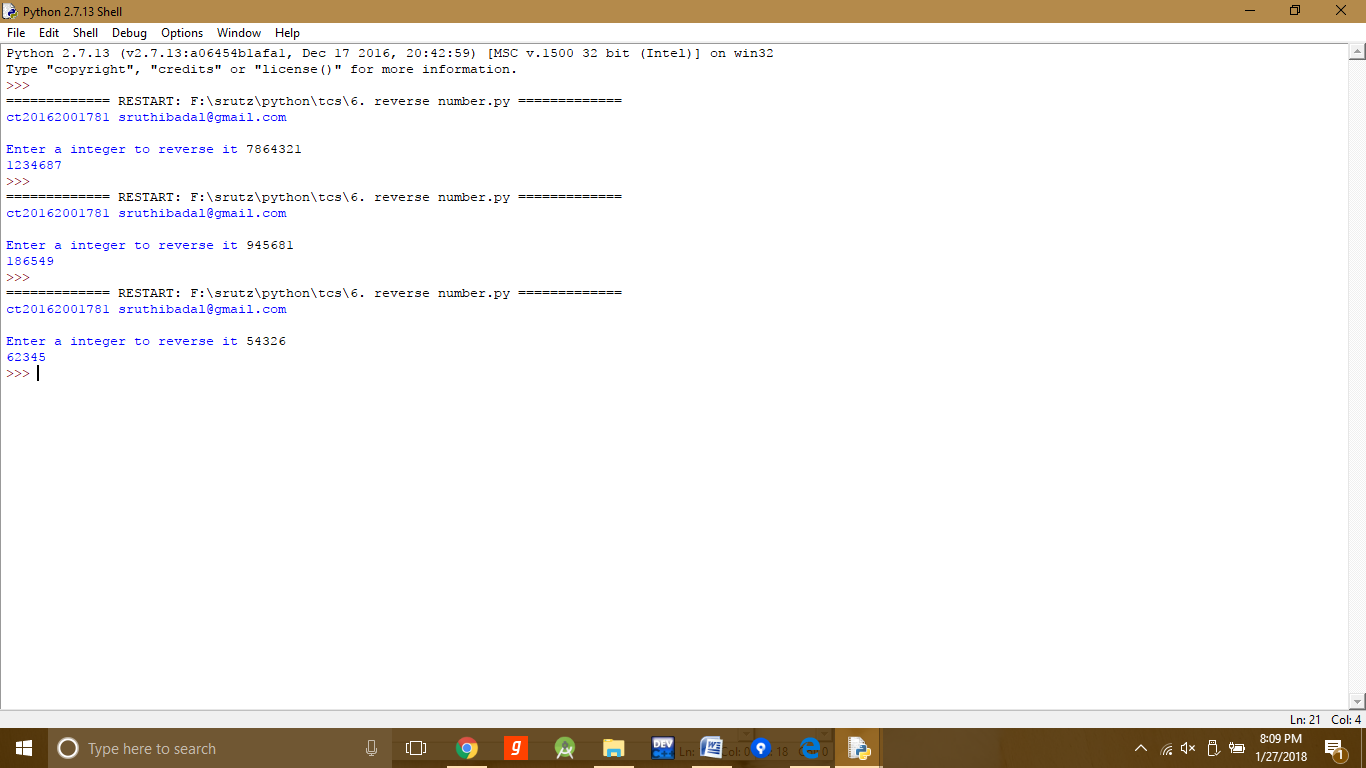
num=num/10

print rev

num=int(raw\_input("Enter a integer to reverse it "))

reverse\_num(num)

**Output:**

****

7. Write a Python program that accept a number and finds the summation of the digits in the number.

**Code:**

def summation(num):

sum=0

while num>0:

n=num%10

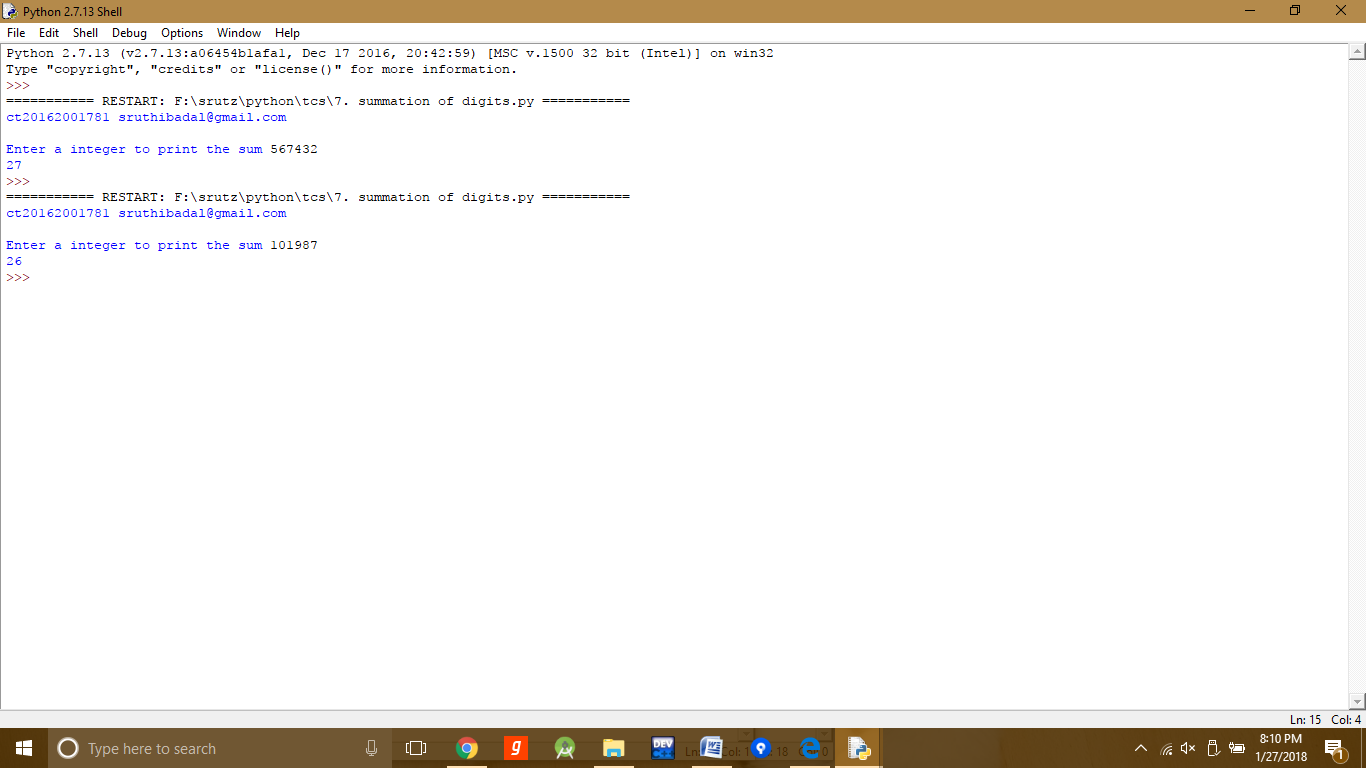
sum+=n;

num/=10

print sum

num=int(raw\_input("Enter a integer to print the sum "))

summation(num)

Output:

8. Write a Python program to copy the content of a given file to another. Take the file names as input from user.

**Code:**

def writing\_to\_file():

try:

input\_file=raw\_input("enter the inputfile name")

out\_file=raw\_input("Enter the output file")

in\_file=open(input\_file,"r+")

out\_file=open(out\_file,"w+")

i=1

for line in in\_file:

print line.rstrip()

out\_file.write(line.rstrip()+"\n")

i=i+1

in\_file.close()

out\_file.close()

print "Successfully Written"

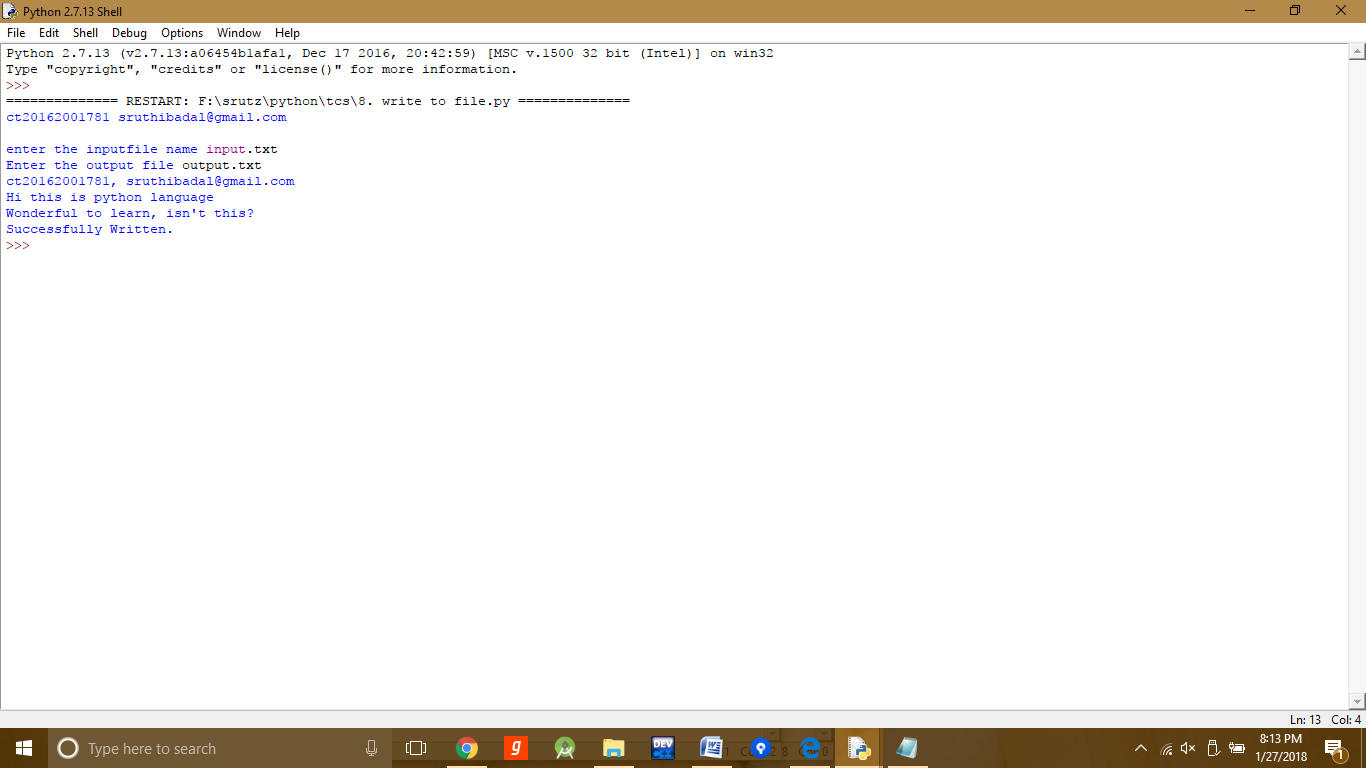
except IOError,e:

print "There was an error in running this program"

print e

writing\_to\_file()

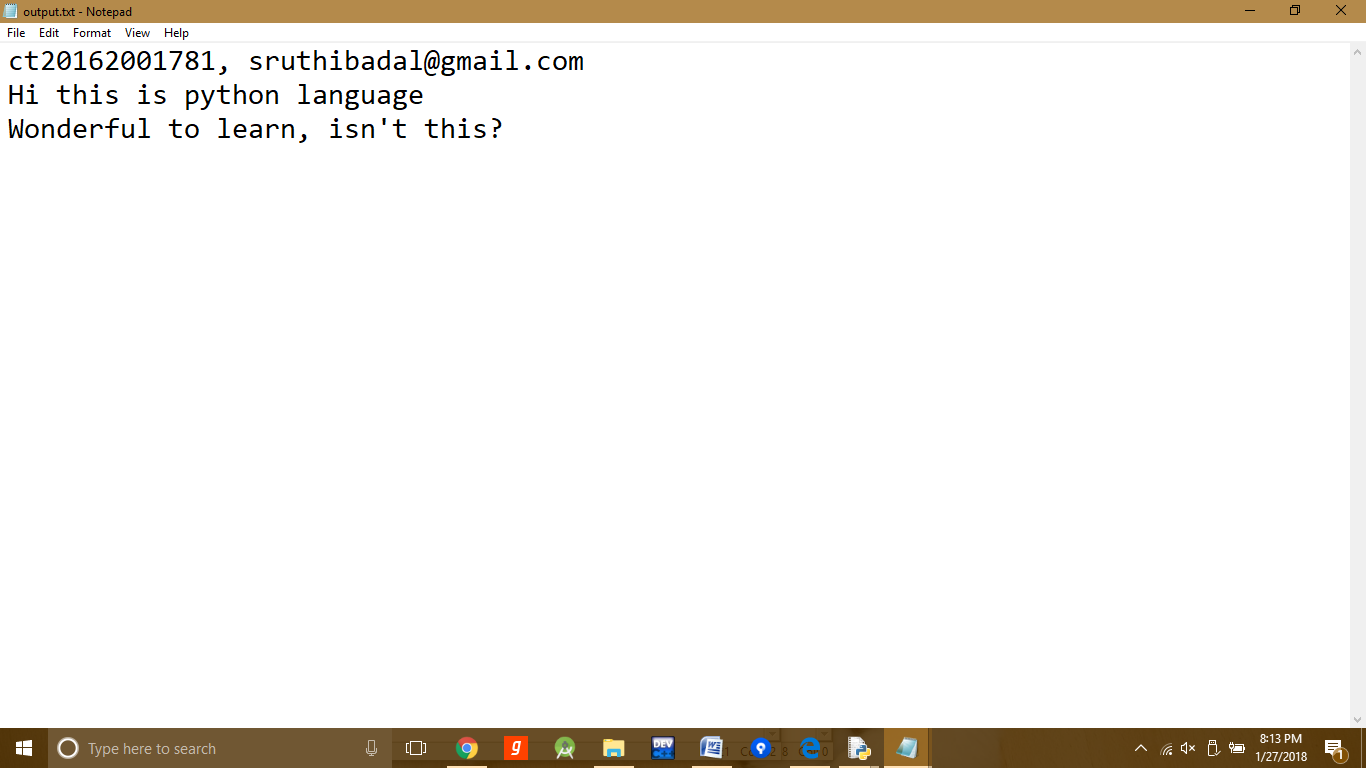
**Output:**



**Input.txt**



**Output.txt**



9. Define a class student with attributes roll number, name and score and methods to find grade. Grade is ‘A’ if score >=80, grade is ‘B’ if score >=60 and score<80, grade is ‘C’ if score >=50 and score<60 otherwise ‘F’.

**Code:**

class student:

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

self.rollnumber=roll

self.name=name

self.score=score

def calculate\_grade(self):

if(self.score>=80):

print 'Grade A'

elif(self.score>=60):

print 'Grade B'

elif(self.score>=50):

print 'Grade C'

else:

print 'Grade F'

rollnumber=int(raw\_input('Enter rollnumber '))

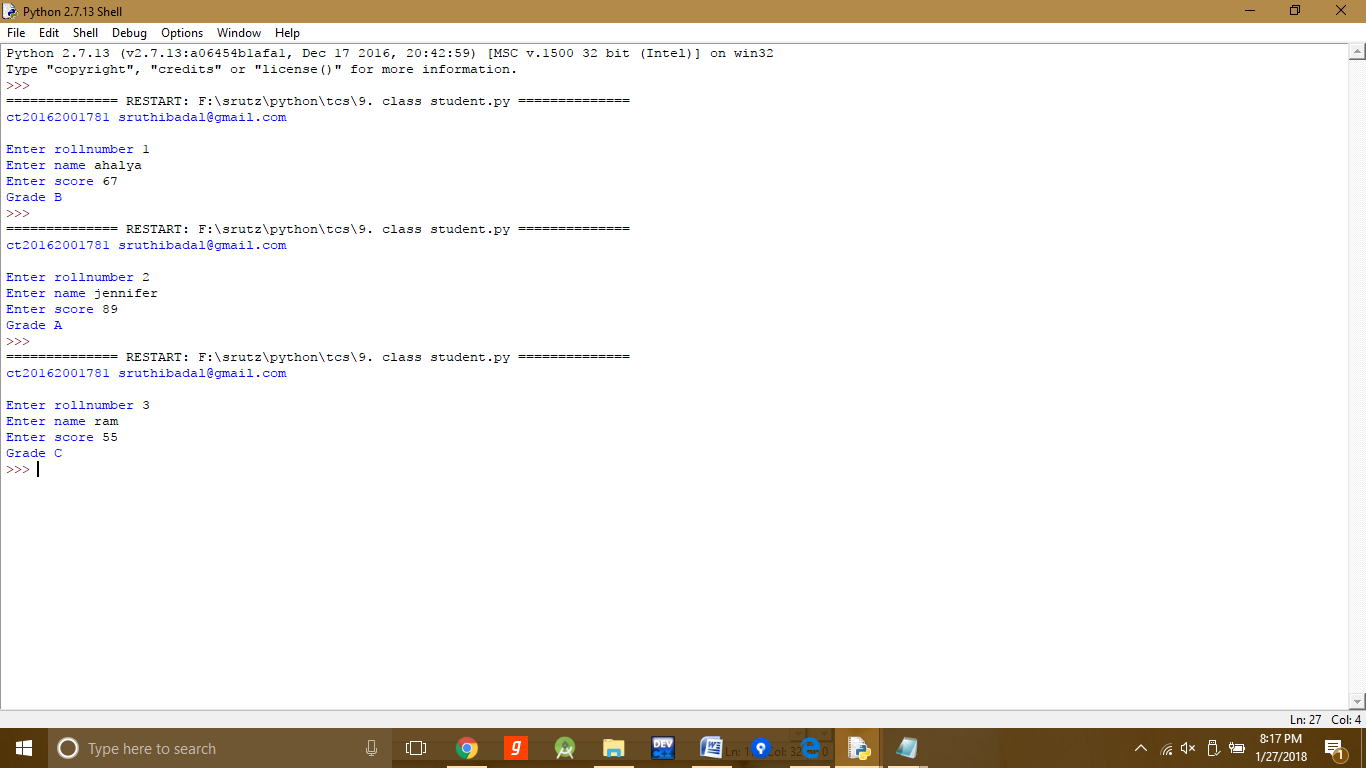
name=raw\_input('Enter name ')

score=int(raw\_input('Enter score '))

stud=student(rollnumber,name,score)

stud.calculate\_grade()

**Output:**



10. Define a class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle and one more method which will check whether the area is greater than a given value or not.

**code:**

class circle:

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

self.radius=radius

self.pi=3.14

def area(self):

self.area=self.pi\*self.radius\*self.radius

print self.area

def perimeter(self):

self.perimeter=2\*self.pi\*self.radius

print self.perimeter

def greaterarea(self,value):

if(self.area>value):

return 'true'

else:

return 'false'

radius=float(raw\_input("Enter radius "))

c=circle(radius)

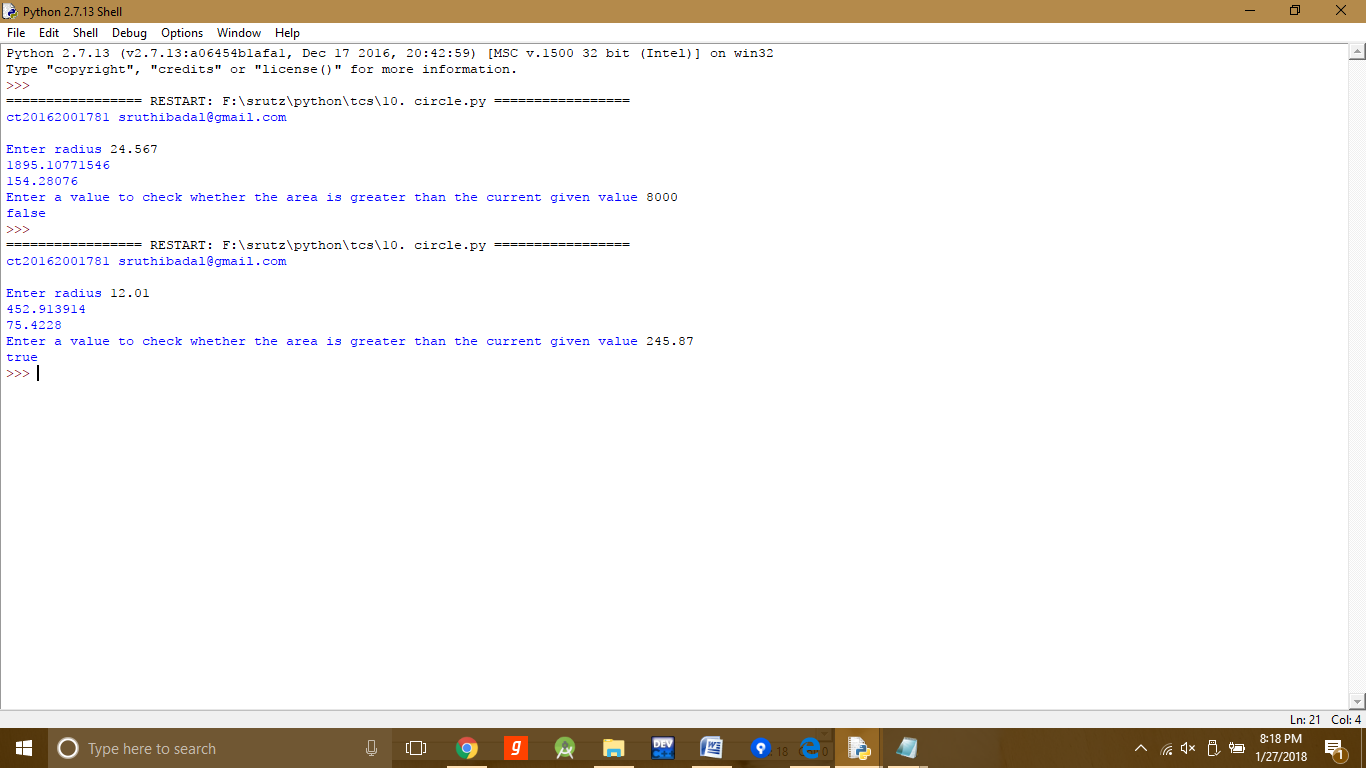
c.area()

c.perimeter()

checkvalue=float(raw\_input("Enter a value to check whether the area is greater than the current given value "))

print c.greaterarea(checkvalue)

**Output:**



11. Define a class Shape with its sub class Square . The Square has an initialize function which takes as argument a length. Both classes have a function findArea which can print the area of the shape where Shape's area is 0 by default

**Code:**

class Shape(object):

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

pass

def findArea(self):

self.area=0.00

class Square(Shape):

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

self.length=length

def findArea(self):

super(Square,self).findArea()

self.area=self.length\*self.length

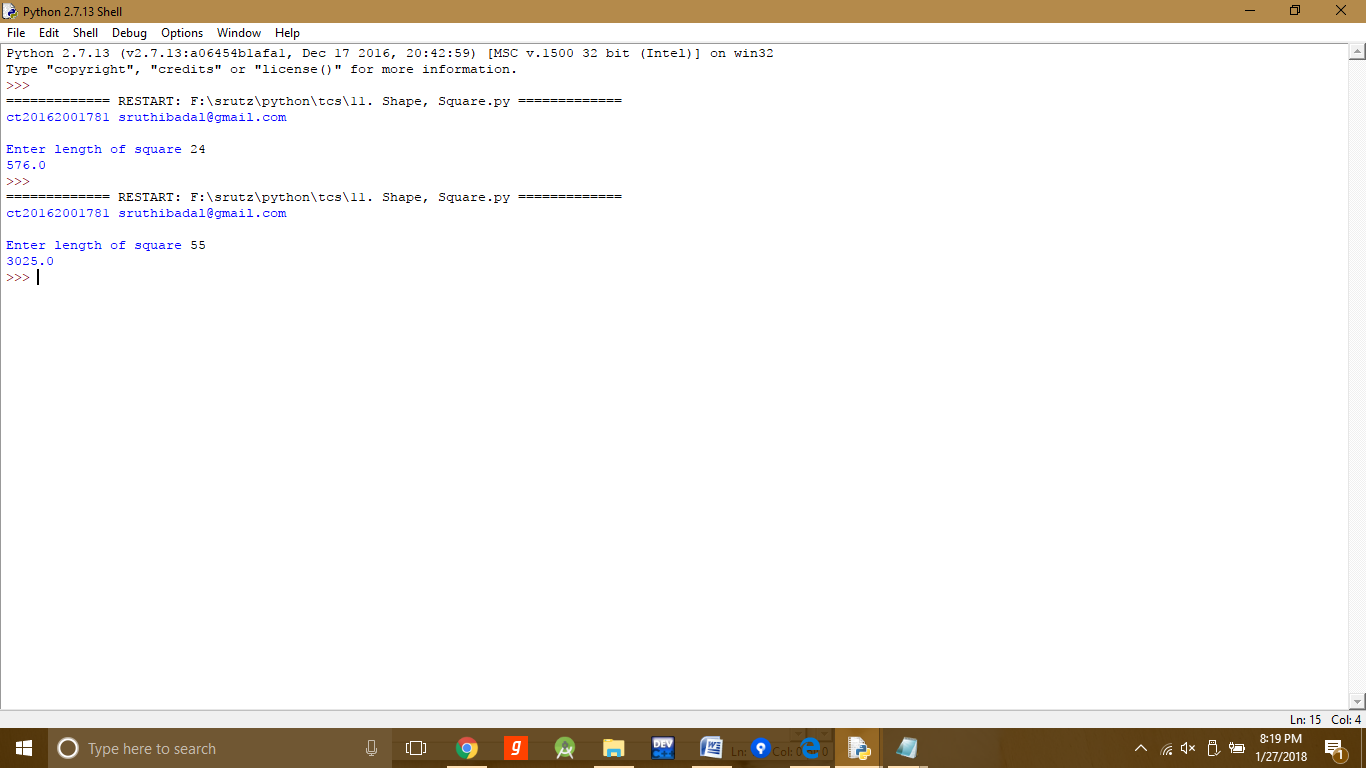
return self.area

length=float(raw\_input("Enter length of square"))

S=Square(length)

print S.findArea()

**Output:**



12. Define a class Item and its two child classes: Perishable and NonPerishable. All classes have attribute price a method "calculateSalePrice" which can print "price + (price \* .1) " for nonperishable class and "price + 500" for perishable class.

**Code:**

class Item(object):

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

pass

def calculateSalePrice(self,price):

self.price = price

print self.price

class Perishable(Item):

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

self.price=price

def calculateSalePrice(self):

self.price=self.price+500

super(Perishable,self).calculateSalePrice(self.price)

class NonPerishable(Item):

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

self.price=price

def calculateSalePrice(self):

self.price=self.price+(self.price\*0.1)

super(NonPerishable,self).calculateSalePrice(self.price)

price = float(raw\_input("Enter the price: "))

print "For perishable class the price would be: ",

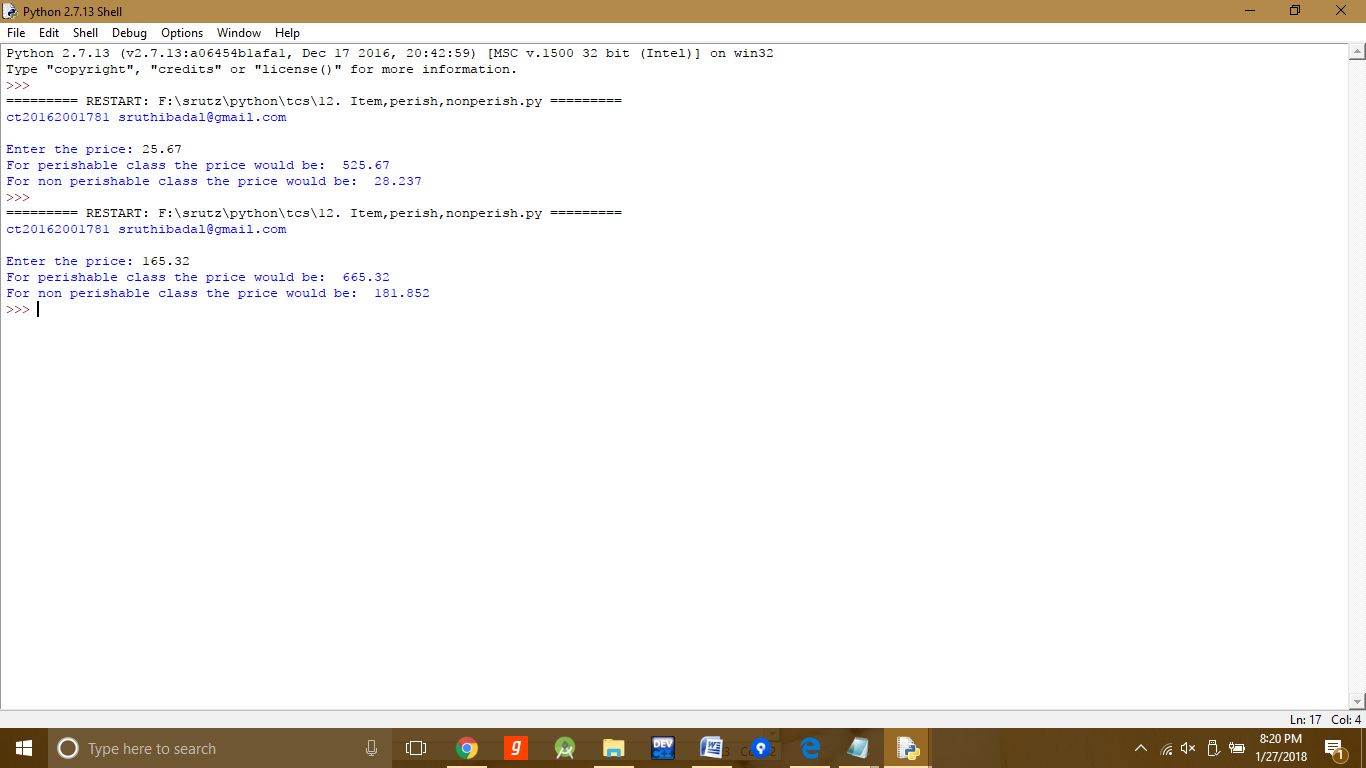
Perish = Perishable(price)

Perish.calculateSalePrice()

print "For non perishable class the price would be: ",

NonPerishable(price).calculateSalePrice()

**Output:**

****