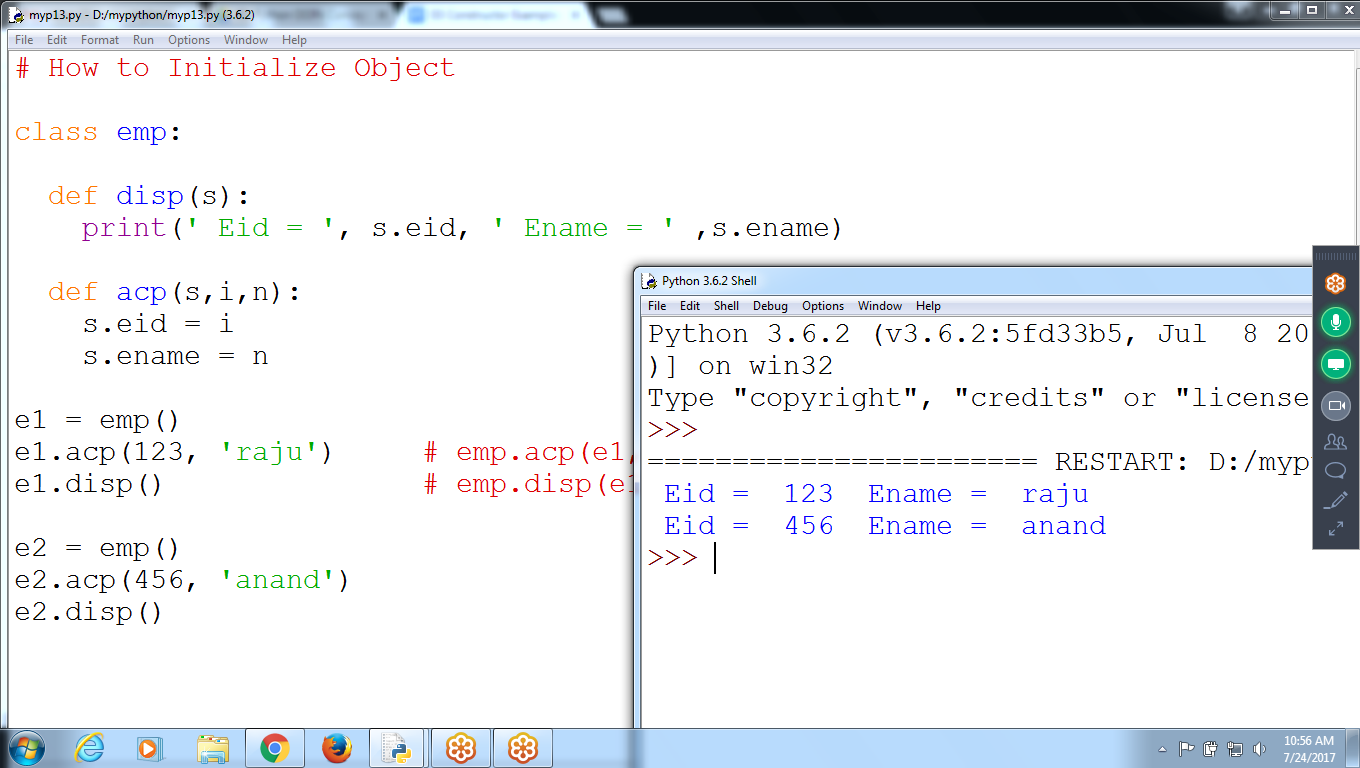
**P1# Passing parameters to acp() and displaying values using disp()**

**To INitialize e1 and e2 objects, calling acp() explicitly**



# How to Initialize Object

class emp:

def disp(s):

print(' Eid = ', s.eid, ' Ename = ' ,s.ename)

def acp(s,i,n):

s.eid = i

s.ename = n

e1 = emp()

e1.acp(123, 'raju') # emp.acp(e1,eid,ename)

e1.disp() # emp.disp(e1)

e2 = emp()

e2.acp(456, 'anand')

e2.disp()

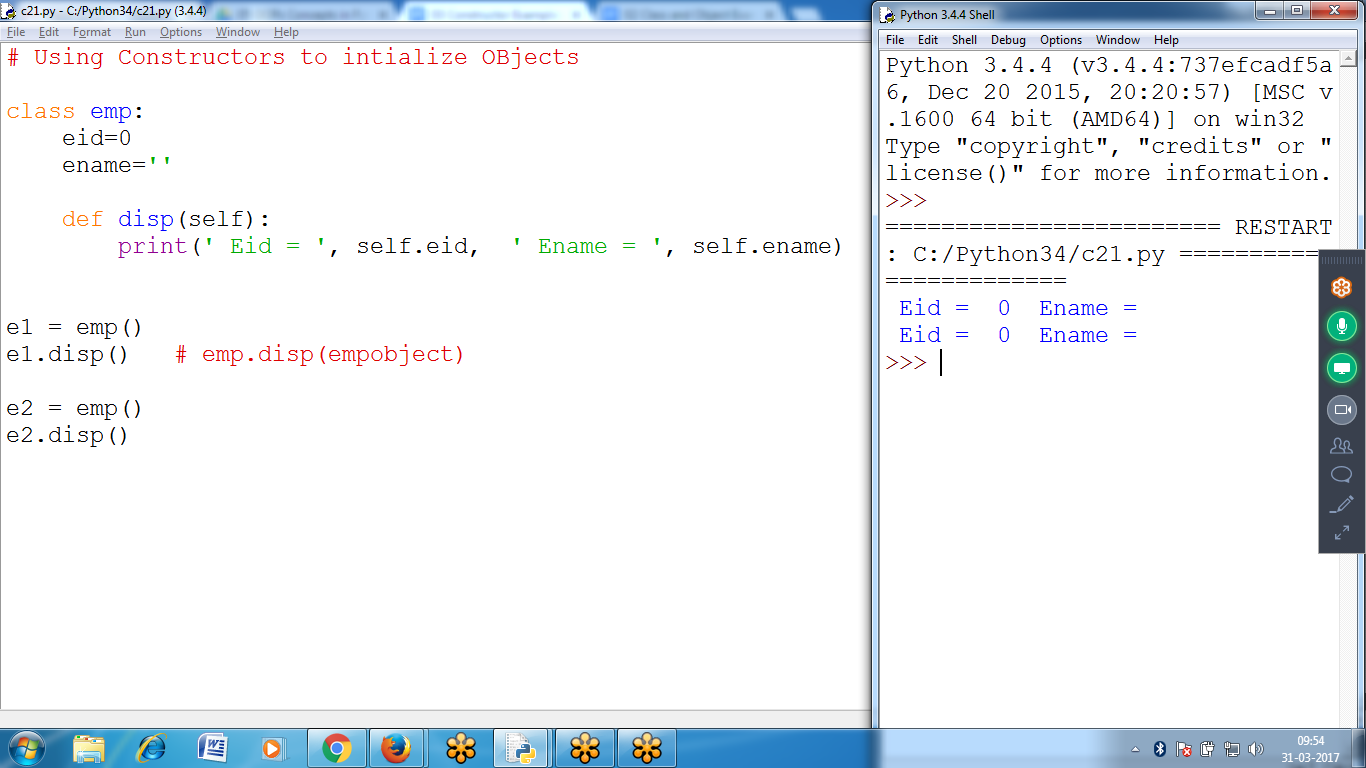
To Initialize OBJECts e1 and e2

calling acp() explicitly

To Effect multiple objects very difficulty to call acp()

**# P2 : To Initialize objects Automatically use CONSTRUCTORS**

**Without Constructor**



# Using Constructors to initialize OBjects

class emp:

eid=0

ename=''

def disp(self):

print(' Eid = ', self.eid, ' Ename = ', self.ename)

e1 = emp()

e1.disp() # emp.disp(empobject)

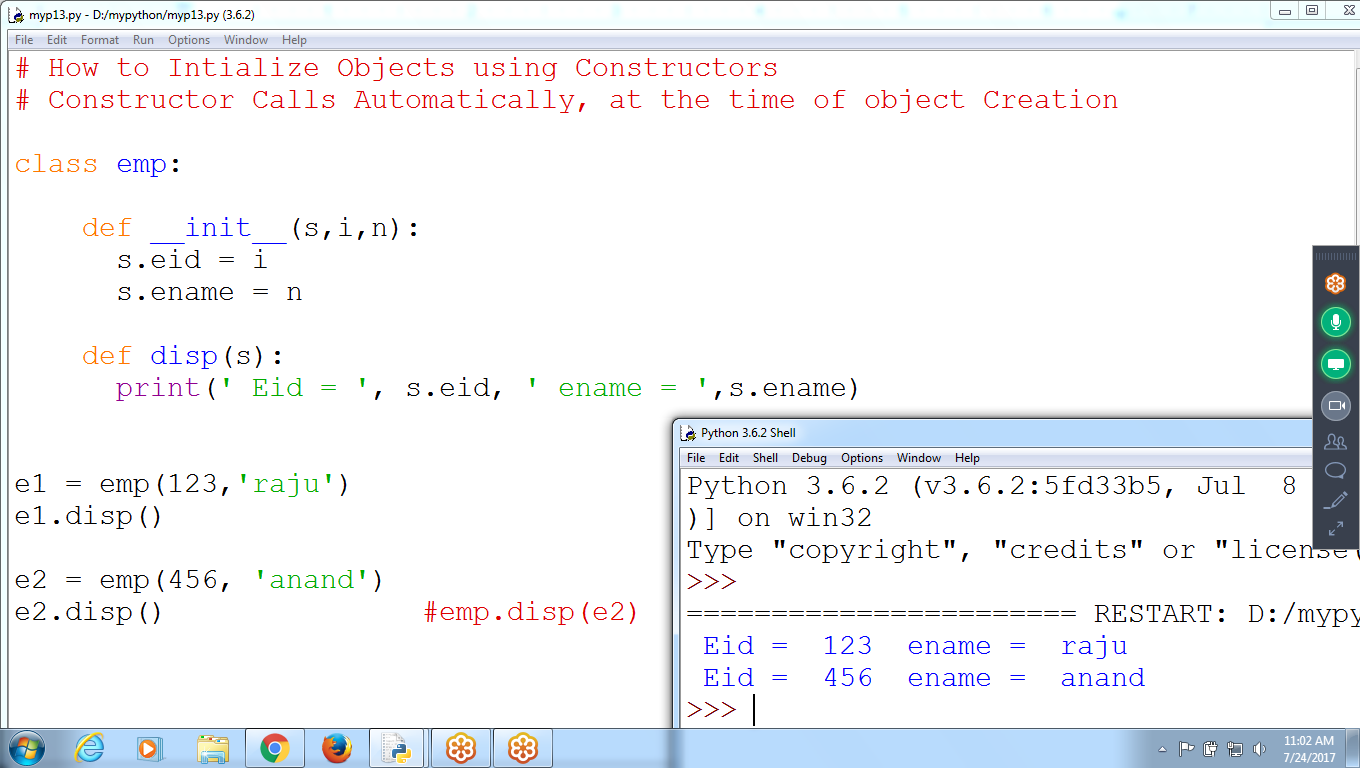
e2 = emp()

e2.disp()

**Purpose of Constructor : TO Initialize Objects at the time of Object Creation**

**IF Their is NO Constructor, have to call method explicitly**

**# P3:: Using Parameter Constructor**

****

**# How to Initialize Objects using Constructors**

**# Constructor Calls Automatically, at the time of object Creation**

**class emp:**

**def \_\_init\_\_(s,i,n):**

**s.eid = i**

**s.ename = n**

**def disp(s):**

**print(' Eid = ', s.eid, ' ename = ',s.ename)**

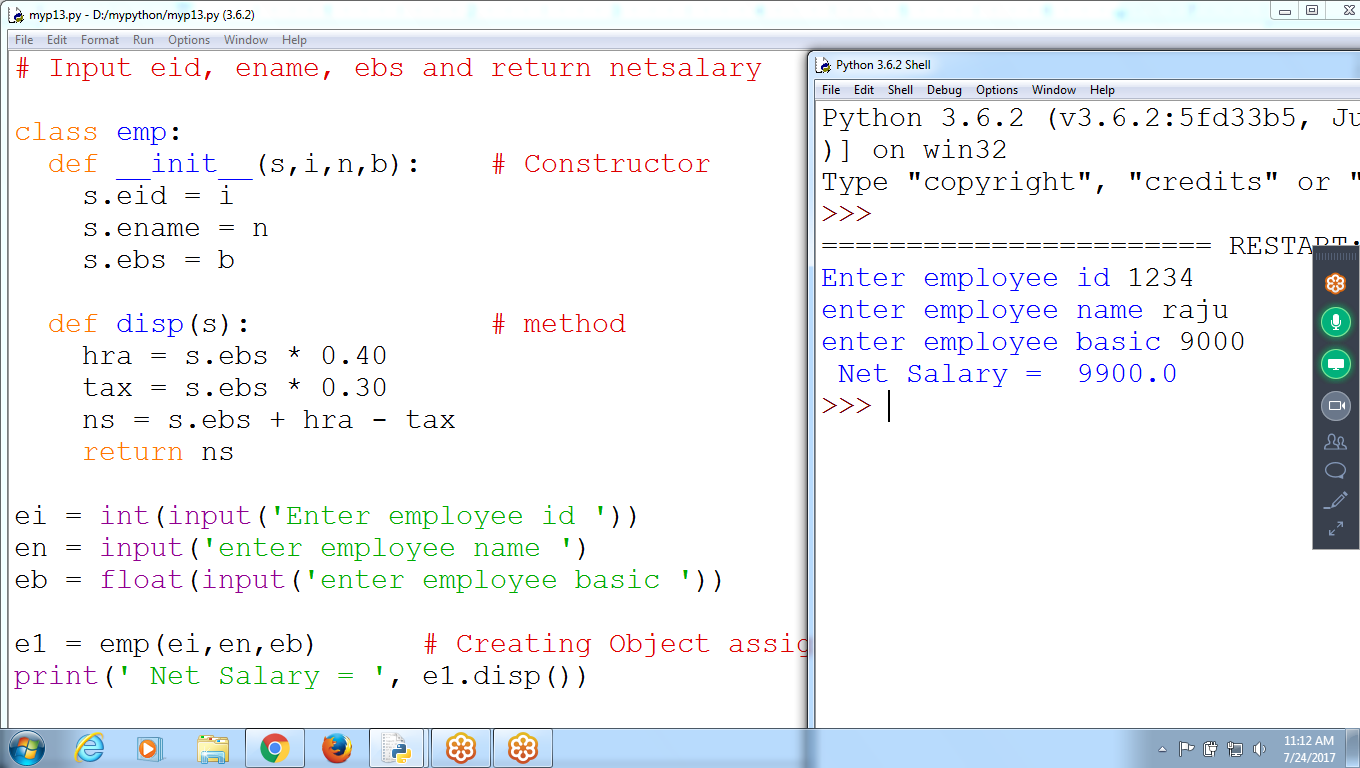
**e1 = emp(123,'raju')**

**e1.disp()**

**e2 = emp(456, 'anand')**

**e2.disp() #emp.disp(e2)**

**Accepting values eid, ename, ebs print Net Salary**

****

# Input eid, ename, ebs and return netsalary

class emp:

def \_\_init\_\_(s,i,n,b): # Constructor

s.eid = i

s.ename = n

s.ebs = b

def disp(s): # method

hra = s.ebs \* 0.40

tax = s.ebs \* 0.30

ns = s.ebs + hra - tax

return ns

ei = int(input('Enter employee id '))

en = input('enter employee name ')

eb = float(input('enter employee basic '))

e1 = emp(ei,en,eb) # Creating Object assigning thru constructor

print(' Net Salary = ', e1.disp())

If No Constructors, need to call Method (acp()) Explicitly

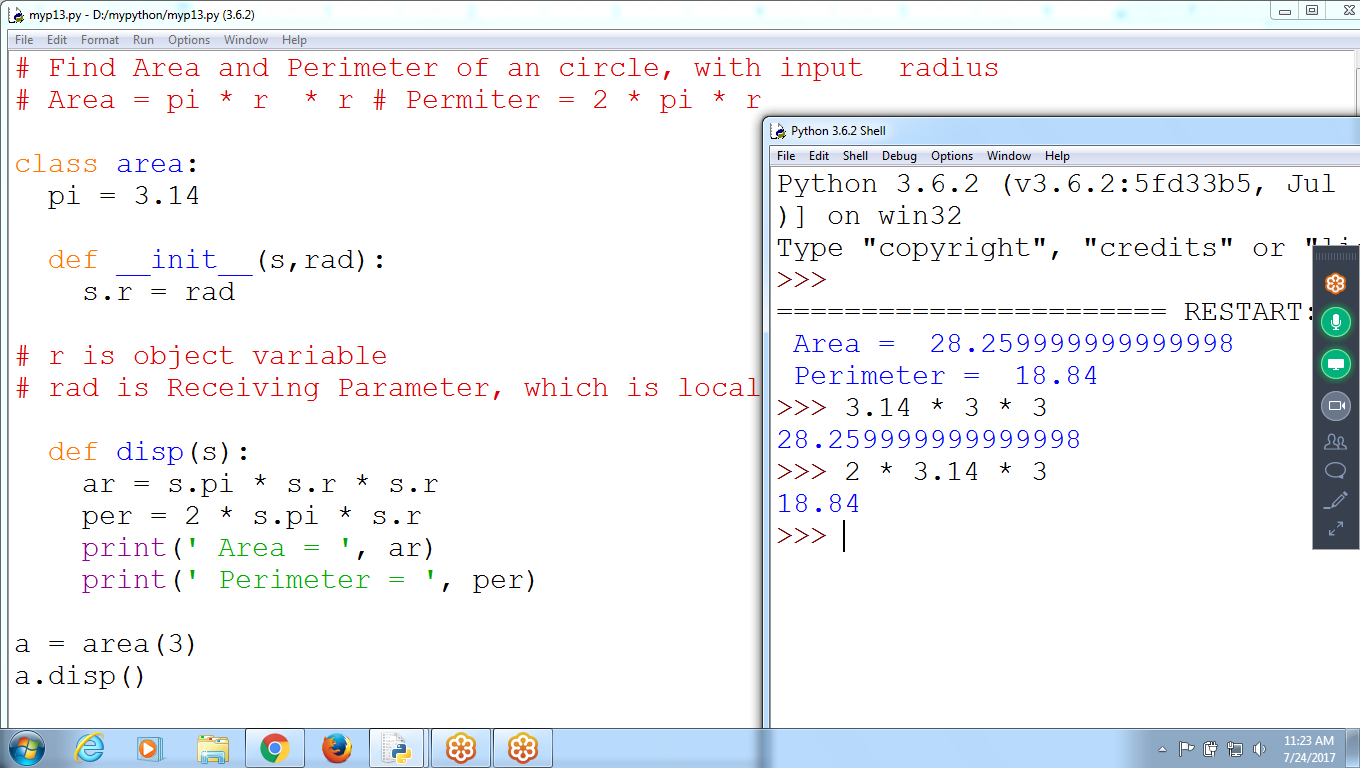
Constructors are used to Initialize object at the time of object creation

ex: **e1 = emp(eid,ename,ebs)**

**# CIRCLE Using Constructor Example**

**# Find ARea of Circle : pi \* r \* r**

**# find perimeter :: 2 \* pi \* r**

****

**# Find Area and Perimeter of a circle, with input radius**

**# Area = pi \* r \* r # Perimeter = 2 \* pi \* r**

**class area:**

**pi = 3.14**

**def \_\_init\_\_(s,rad):**

**s.r = rad**

**# r is object variable**

**# rad is Receiving Parameter, which is local**

**def disp(s):**

**ar = s.pi \* s.r \* s.r**

**per = 2 \* s.pi \* s.r**

**print(' Area = ', ar)**

**print(' Perimeter = ', per)**

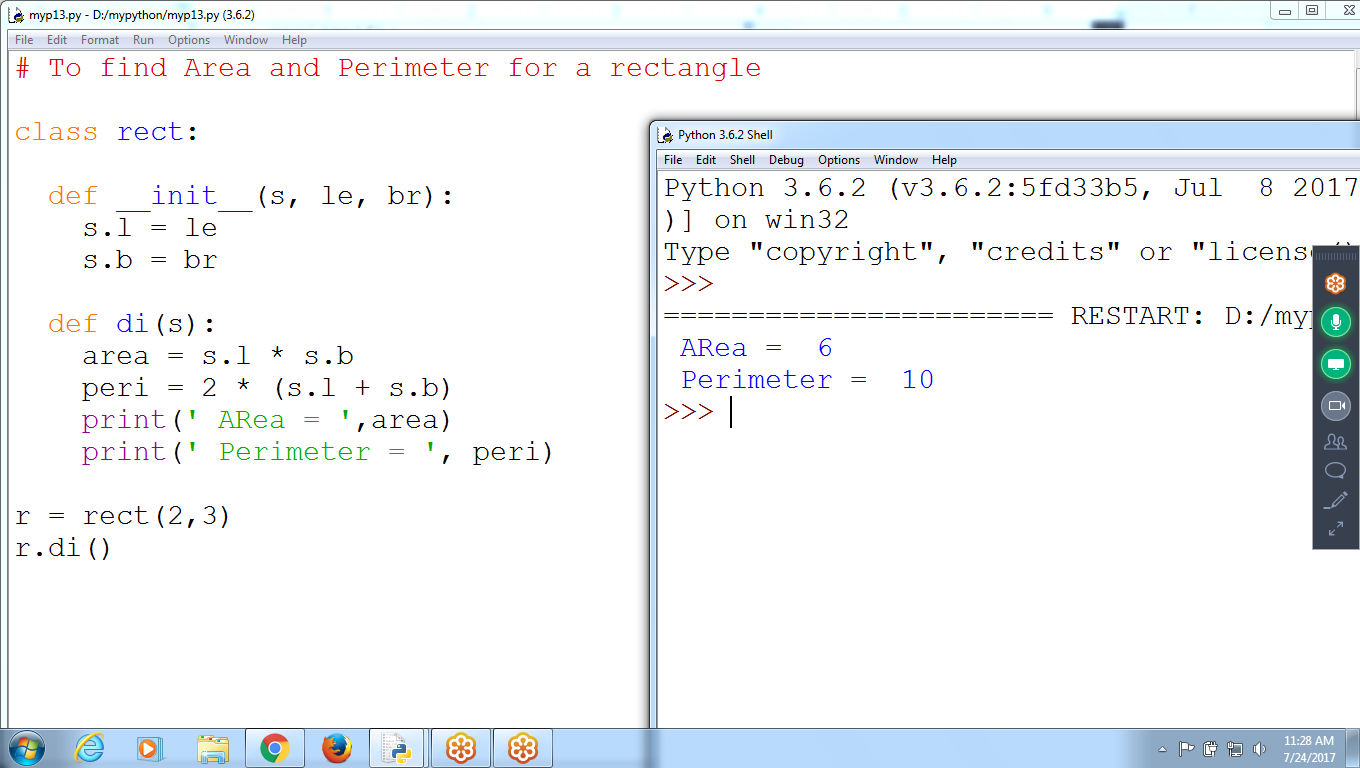
**a = area(3)**

**a.disp()**

**# Using Constructors**

**# TO Find area of REctangle :: l \*B**

**# To Perimeter of rectangle : 2\* (l+b)**

****

**# To find Area and Perimeter for a rectangle**

**class rect:**

**def \_\_init\_\_(s, le, br):**

**s.l = le**

**s.b = br**

**def di(s):**

**area = s.l \* s.b**

**peri = 2 \* (s.l + s.b)**

**print(' ARea = ',area)**

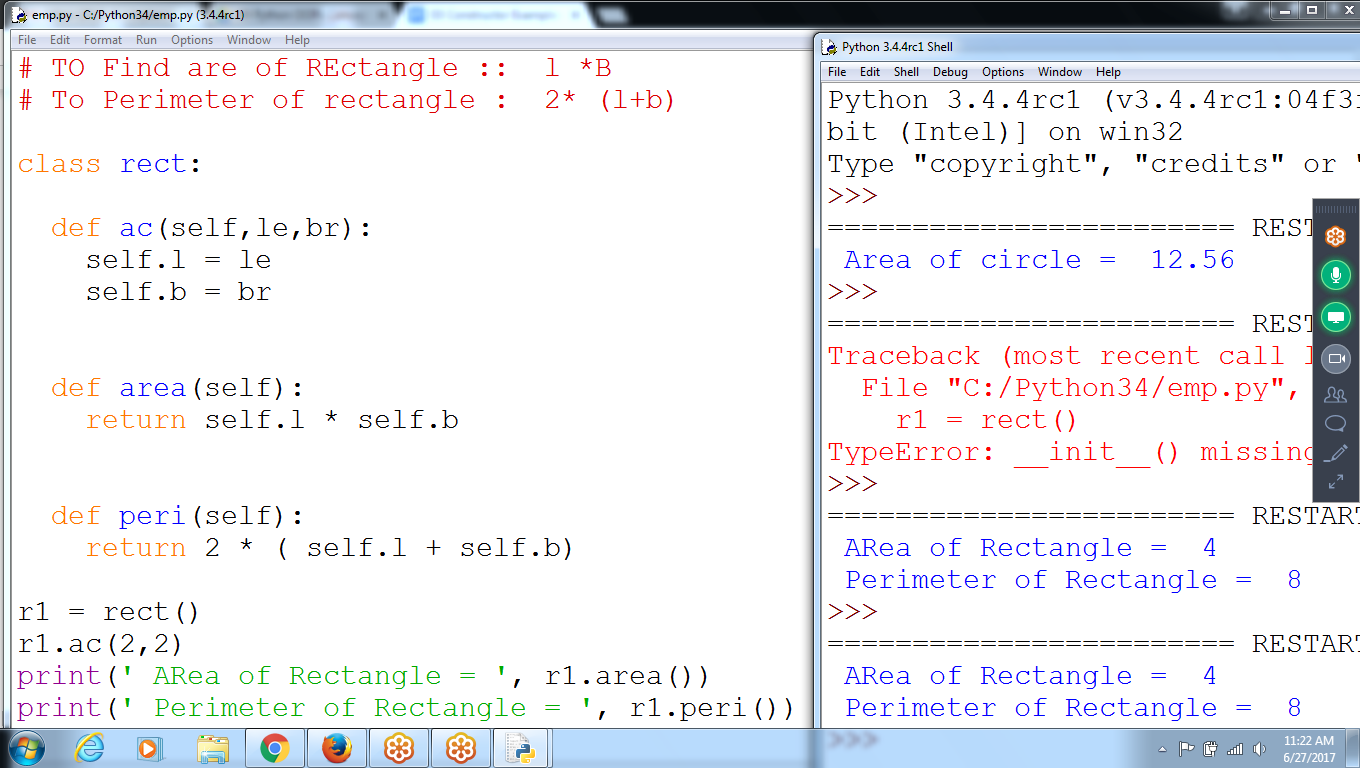
**print(' Perimeter = ', peri)**

**r = rect(2,3)**

**r.di()**

**Without Constructor**

**Calling ac() Method Explicitly**

****

**# TO Find are of REctangle :: l \*B**

**# To Perimeter of rectangle : 2\* (l+b)**

**class rect:**

**def ac(self,le,br):**

**self.l = le**

**self.b = br**

**def area(self):**

**return self.l \* self.b**

**def peri(self):**

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

**r1 = rect()**

**r1.ac(2,2)**

**print(' ARea of Rectangle = ', r1.area())**

**print(' Perimeter of Rectangle = ', r1.peri())**