**Co4**

**Program 1-create a rectangle class with attributes length and breadth and methods to find area and perimeter .compare 2 rectangle objects by their area.**

**class rectangle:**

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

**self.length=length**

**self.breadth=breadth**

**def area(self):**

**a=self.length\*self.breadth**

**print("area=",a)**

**return(a)**

**def perimeter(self):**

**p=2\*(self.length+self.breadth)**

**print("perimeter=",p)**

**return(p)**

**p1=rectangle(4,2)**

**b=p1.area()**

**p1.perimeter()**

**p2=rectangle(5,2)**

**c=p2.area()**

**p2.perimeter()**

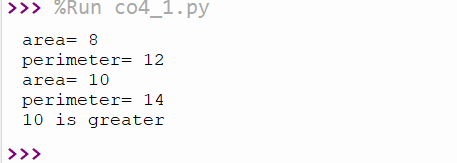
**if(b>c):**

**print(b,"is greater")**

**else:**

**print(c,"is greater")**

**output:**

****

**program 2:create a bank account with members account numbers,names,type of account ad balance.Write constructor and methods to deposit at the bank and withdraw an amount from the bank .**

**class bank:**

**bal=0**

**def \_\_init\_\_(self,accno,name,ac\_type,bal):**

**self.accno=accno**

**self.name=name**

**self.ac\_type=ac\_type**

**self.bal=bal**

**def display(self):**

**print("\nAccount Info:")**

**print("Account Number:",self.accno)**

**print("Account Name:",self.name)**

**print("Account Type:",self.ac\_type)**

**print("Account Balance:",self.bal)**

**def deposit(self):**

**dep=int(input("Enter the amount to deposit:"))**

**self.bal=self.bal+dep**

**def withdraw(self):**

**w=int(input("Enter the amount to withdraw:"))**

**if w > self.bal:**

**print("Insufficient Balance")**

**else:**

**self.bal=self.bal-w**

**print("RS-",w,"Withdrawn successfully")**

**acc\_no=int(input("Enter the Account Number:"))**

**acc\_name=input("Enter the name:")**

**acc\_type=input("Enter the account type-(savings/current):")**

**balance=int(input("Enter the initial balance:"))**

**b1=bank(acc\_no,acc\_name,acc\_type,balance)**

**while(1):**

**print("\n1.Account Info\n2.Deposit\n3.Withdraw\n4.Exit")**

**opt=int(input("Select your option:"))**

**if opt == 1:**

**b1.display()**

**elif opt == 2:**

**b1.deposit()**

**elif opt == 3:**

**b1.withdraw()**

**elif opt == 4:**

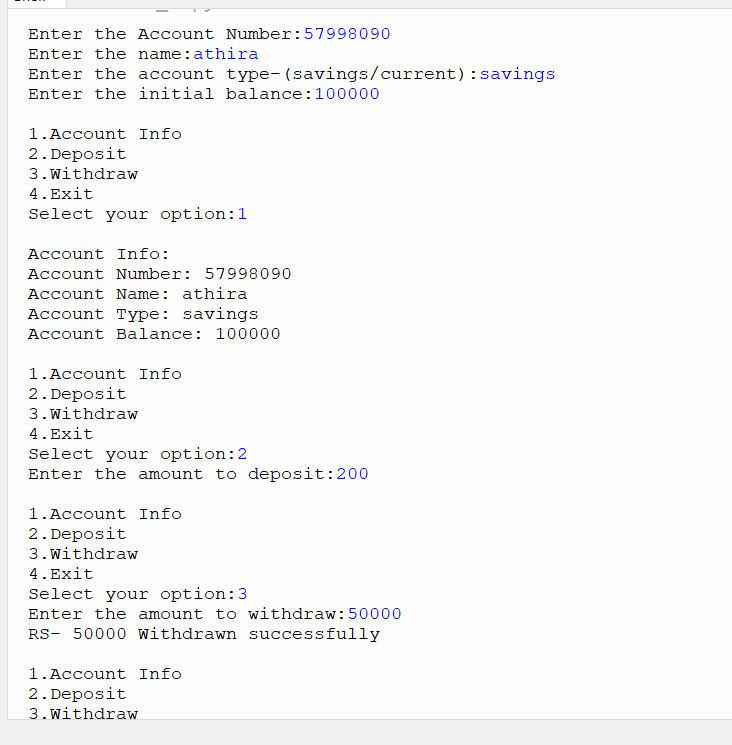
**print("Exited")**

**break**

**else:**

**print("Invalid Option")**

**output:**

****

**Program 3:create a class rectangle with private attributes length and width .Overload’<’ operator to compare the area of 2 rectangles.**

**class rectangle:**

**def \_\_init\_\_(self,l,b):**

**self.\_\_length=l**

**self.\_\_breadth=b**

**def area(self):**

**self.area=self.\_\_length\*self.\_\_breadth**

**print("Area=",self.area)**

**def \_\_lt\_\_(self,second):**

**if self.area < second.area:**

**return True**

**else:**

**return False**

**print("first Rectangle:")**

**len1=int(input("Enter the length:"))**

**bread1=int(input("Enter the breadth:"))**

**obj1=rectangle(len1,bread1)**

**obj1.area()**

**print("\nSecond Rectangle:")**

**len2=int(input("Enter the length:"))**

**bread2=int(input("Enter the breadth:"))**

**obj2=rectangle(len2,bread2)**

**obj2.area()**

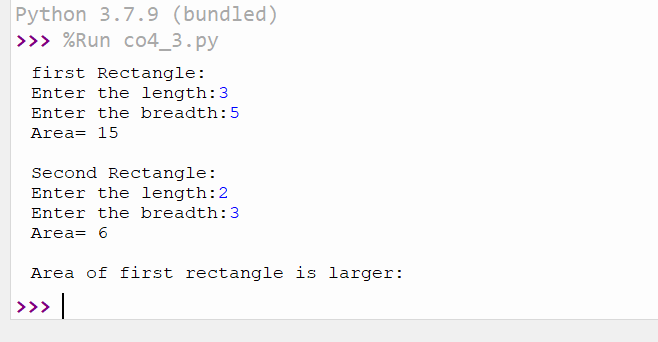
**if obj1 < obj2 :**

**print("\nArea of second rectangle is larger:")**

**else:**

**print("\nArea of first rectangle is larger:")**

**output:**

****

**Program 4:** **Create a class Time with private attributes hour, minute and second. Overload ‘+’ operator to find sum of 2 time.**

**class time:**

**def \_\_init\_\_(self,hour,minute,second):**

**self.\_\_hour=hour**

**self.\_\_minute=minute**

**self.\_\_second=second**

**def \_\_add\_\_(self,second):**

**print("\nHour:",self.\_\_hour + second.\_\_hour)**

**print("Minutes:",self.\_\_minute + second.\_\_minute)**

**print("Seconds:",self.\_\_second + second.\_\_second)**

**hour1=int(input("Enter the hour:"))**

**minute1=int(input("Enter the minutes:"))**

**sec1=int(input("Enter the second:"))**

**obj1=time(hour1,minute1,sec1)**

**hour2=int(input("\nEnter the hour:"))**

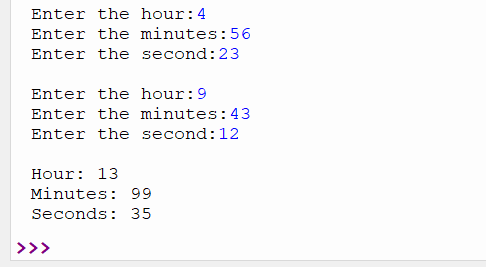
**minute2=int(input("Enter the minutes:"))**

**sec2=int(input("Enter the second:"))**

**obj2=time(hour2,minute2,sec2)**

**obj1 + obj2**

**output:**

****

**Program 5:** **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.**

**class publisher:**

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

**self.title=title**

**self.author=author**

**def display(self):**

**print("Title:",self.title)**

**print("Author:",self.author)**

**class book(publisher):**

**def \_\_init\_\_(self,price,no\_of\_page):**

**self.price=price**

**self.no\_of\_page=no\_of\_page**

**def display(self):**

**print("Price:",self.price)**

**print("No. of Pages:",self.no\_of\_page)**

**class python(book):**

**def \_\_init\_\_(self,title,author,price,no\_of\_page):**

**publisher.\_\_init\_\_(self,title,author)**

**book.\_\_init\_\_(self,price,no\_of\_page)**

**def display(self):**

**print("Title:",self.title)**

**print("Author:",self.author)**

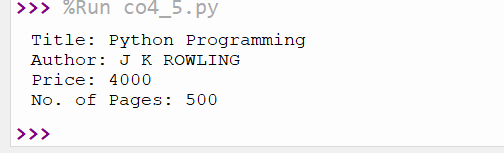
**print("Price:",self.price)**

**print("No. of Pages:",self.no\_of\_page)**

**p=python("Python Programming","J K ROWLING",4000,500)**

**p.display()**

**output:**

****