**CO4\_1**

**Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area**

class rectangle():

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

self.breadth=breadth

self.length=length

def area(self):

return self.breadth\*self.length

def perimeter(self):

return 2\*(self.breadth+self.length)

r1=rectangle(45,24)

r2=rectangle(10,20)

print("Area of rectangle 1 :",r1.area())

print("Area of rectangle 2 :",r2.area())

print("perimeter of rectangle 1:",r1.perimeter())

print("perimeter of rectangle 2:",r2.perimeter())

if(r1.area()>r2.area()):

print("recangle 1 is of greater area")

else:

print("recangle 2 is of greater area")

**OUTPUT**

Area of rectangle 1 : 1080

Area of rectangle 2 : 200

perimeter of rectangle 1: 138

perimeter of rectangle 2: 60

recangle 1 is of greater area

**CO4\_2**

**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.**

class Bank:

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

self.bal=0

print("account is created")

def deposit(self):

amount=int(input("enter amount to deposit"))

self.bal=self.bal+amount

print("balance:",self.bal)

def withdraw(self):

amount=int(input("enter amount to withdraw"))

if(amount>self.bal):

print("Insufficient Balance!")

else:

self.bal=self.bal-amount

print("Your Remaining Balance=",self.bal)

def enquiry(self):

print("Your Balance =",self.bal)

b1= Bank()

b1.deposit()

b1.withdraw()

b1.enquiry()

**OUTPUT**

account is created

enter amount to deposit20000

balance: 20000

enter amount to withdraw200

Your Remaining Balance= 19800

Your Balance = 19800

**CO4\_3**

**Create a class Rectangle with private attributes length and width. Overload ‘<’ operator to compare the area of 2 rectangles.**

class rectangle:

\_\_area = 0

\_\_perimeter = 0

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

self.\_\_length = length

self.\_\_breadth = breadth

def calc\_area(self):

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

print("Area is :",self.\_\_area)

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

if self.\_\_area < second.\_\_area:

return True

else:

return False

length1= int(input("Enter length of the rectangle 1 : "))

breadth1 = int(input("Enter width of the rectangle 1 : "))

length2 = int(input("Enter length of the rectangle 2 : "))

breadth2 = int(input("Enter width of the rectangle 2 : "))

obj1 = rectangle(length1,breadth1)

obj2 = rectangle(length2,breadth2)

obj1.calc\_area()

obj2.calc\_area()

if obj1 < obj2:

print("Rectangle two is large")

else:

print("Rectangle one is large or these are equal")

**OUTPUT**

Enter length of the rectangle 1 : 3

Enter width of the rectangle 1 : 4

Enter length of the rectangle 2 : 46

Enter width of the rectangle 2 : 5

Area is : 12

Area is : 230

Rectangle two is large

**CO4\_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,a2):

second=self.\_\_second+a2.\_\_second

minute=self.\_\_minute+a2.\_\_minute

hour=self.\_\_hour+a2.\_\_hour

if(second>60):

second=second-60

minute=minute+1

if(minute>60):

minute=minute-60

hour=hour+1

return hour,minute,second

print("Enter time1:")

h1=int(input("hour:"))

m1=int(input("minute:"))

s1=int(input("second"))

t1=Time(h1,m1,s1)

print("Enter time2:")

h2=int(input("hour:"))

m2=int(input("minute:"))

s2=int(input("second"))

t2=Time(h2,m2,s2)

hr,min,sec=t1+t2

print(hr,end=":")

print(min,end=":")

print(sec,end=" ")

**OUTPUT**

Enter time1:

hour:2

minute:3

second3

Enter time2:

hour:4

minute:2

second43

6:5:46

**CO4\_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","AnilKumar",1000,120)

p.display()

**OUTPUT**

Title: Python Programming

Author: AnilKumar

Price: 1000

No. of Pages: 120