**DEPARTMENT OF INFORMATION TECHNOLOGY**

**COURSE CODE:** DJ19ITL406 **DATE: 10/05/2022**

**COURSE NAME:** Programing Laboratory 2 (Python) **CLASS:** SYBTECH

**EXPERIMENT NO. 6**

**CO/LO: CO1, CO2.**

**AIM / OBJECTIVE:**

Write python programs to implement Inheritance &Polymorphism.[Classes & Objects, Constructors]

**DESCRIPTION OF EXPERIMENT:**

1. Object oriented features
2. Self-variables
3. Constructors
4. Type of variables
5. Methods
6. Inner classes

**QUESTIONS:**

1. Define a method in the inner class and access the same by code outside the outerclass
2. Create a class Rectangle. The class has 2 attributes, length and width, each of which defaults to 0. It has methods to calculate the perimeter and area of the rectangle. It has set and get methods for both length and width. The set method should verify that length and width are floating point numberslargerthan0.0andlessthan20.0
3. WAP to arrange the names of students in descending order of their total marks, input data consists of student’s details such as names,ID. no, marks of math,physics,and chemistry.
4. WAP to display area of square and rectangle using constructor

**Code:**

**1.**

class A:  
def met(self):  
print("Hello sir .... welcome to class A")  
  
class B(A):  
def met(self):  
print("Hello sir .... welcome to class B")  
  
class C(A):  
def met(self):  
print("Hello sir .... welcome to class C")  
  
class D(B,C):  
def met(self):  
pass  
print("Hello sir .... welcome to class D")  
  
  
  
d=D()  
d.met()

**2.**

class Rectangle:  
  
def Perimeter(self):  
self.length=0  
self.width=0  
  
def getPerimeter(self,length,width):  
self.length = length  
self.width = width  
return 2\*(self.length+self.width)  
  
  
def setPerimeter(self):  
  
if (((self.lengthand self.width) >0.0) and ((self.lengthand self.width)<20.0)):  
print("Data in range")  
  
else:  
print("Data out of range")  
  
def Area(self):  
self.length = 0  
self.width = 0  
  
def getArea(self, length, width):  
self.length = length  
self.width = width  
return self.length \* self.width  
  
def setArea(self):  
  
if (self.length>0.0 and self.width<20.0):  
length= self.length  
print("Data in range")  
  
else:  
print("Data out of range")  
  
  
  
s=Rectangle()  
s.Perimeter()  
print(s.getPerimeter(4,1))  
s.setPerimeter()  
  
s.Area()  
print(s.getArea(2,10))  
s.setArea()

**3.**

n = int(input('How many student are in the class: '))  
  
  
ls = []  
  
for iin range(0, n):  
  
 x = input("Enter the student name :")  
 y=input(f"Enter Percentage of {x}")  
  
  
  
ls.append((y, x))  
  
  
ls = sorted(ls, reverse=True)  
  
print('Sorted list in descending Order---')  
  
for iin ls:  
  
print(i[1], i[0])

**4.**class Quadrilateral:  
  
def \_\_init\_\_(self, a, b, c, d):  
self.a = a  
self.b = b  
self.c = c  
self.d = d  
  
def Perimeter(self):  
return self.a + self.b + self.c + self.d  
  
  
class Rectangle(Quadrilateral):  
def \_\_init\_\_(self, a, b):  
super().\_\_init\_\_(a, b, a, b)  
  
def area(self):  
return self.a \* self.b  
  
  
class Square(Rectangle):  
def \_\_init\_\_(self, a):  
super().\_\_init\_\_(a, a)  
  
def area(self):  
return self.a \* self.a  
  
  
a = Quadrilateral(2, 3, 4, 5)  
print(a.Perimeter())  
  
b = Rectangle(8, 9)  
print(b.area())  
print(b.Perimeter())  
  
c = Square(5)  
print(c.area())  
print(c.Perimeter())

**OBSERVATIONS / DISCUSSION OF RESULT:**

To understand the meaning of classes we have to understand the built-in \_\_init\_\_() function.

All classes have a function called \_\_init\_\_(), which is always executed when the class is being initiated.

The \_\_init\_\_() function is called automatically every time the class is being used to create a new object.

Inheritance allows us to define a class that inherits all the methods and properties from another class.

**Parent class** is the class being inherited from, also called base class.

**Child class** is the class that inherits from another class, also called derived class.

There are 3 types of Inheritance :

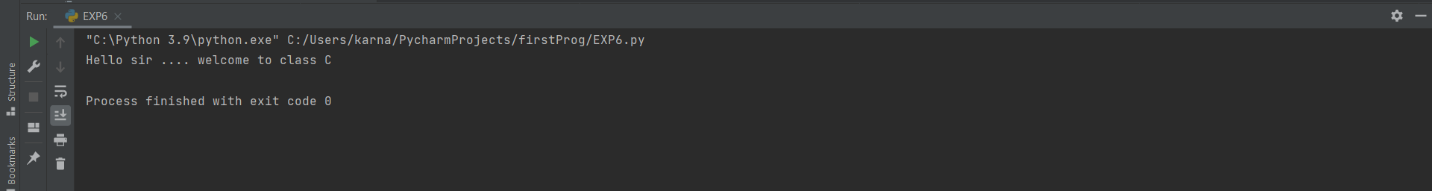
1.Single Inheritance

2.Multipleinheritance

3.Multivalued Inheritance

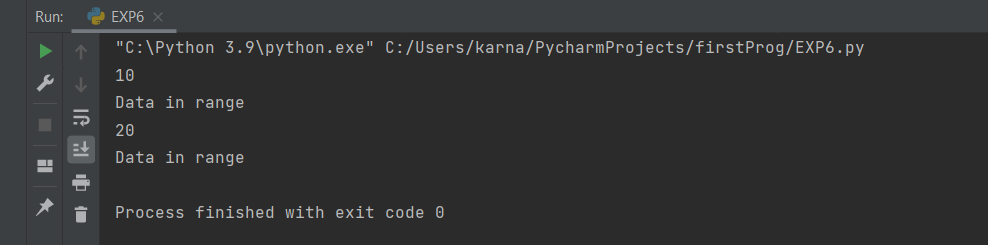
Python also has a super() function that will make the child class inherit all the methods and properties from its parent class.

1.

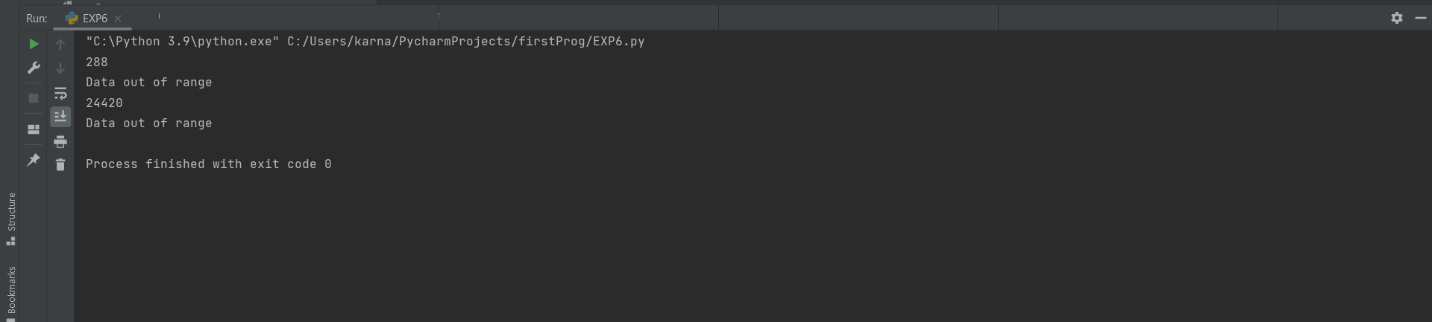


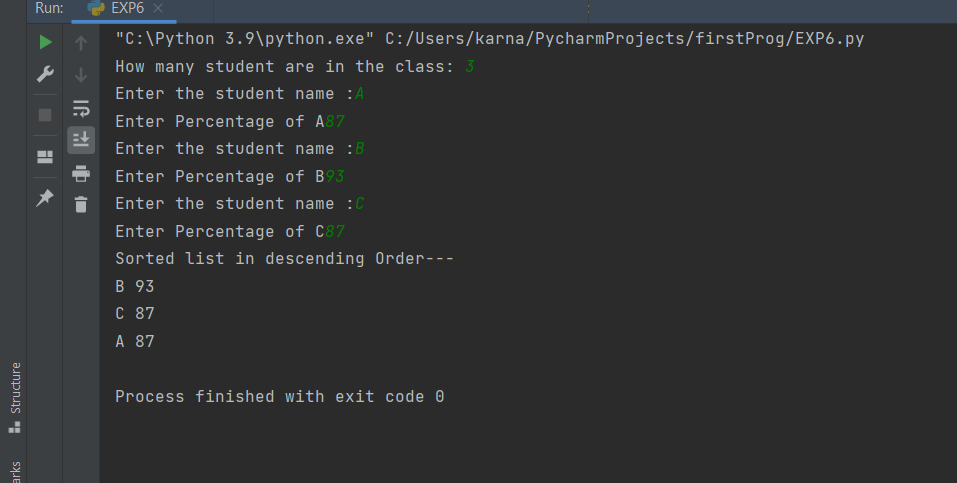
2.

Case1:

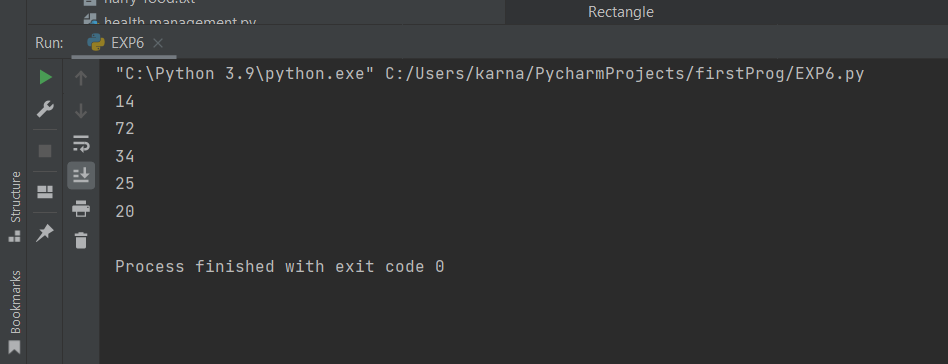


Case2:



3.

4.



**CONCLUSION:**

Hence we have understood and successfully implemented Concepts of OOPs.

**REFERENCES:**

**Website References:​**

[1] https://www.w3schools.com/python