EX - 02 SIMPLE INHERITANCE

T SADAKOPA RAMAKRISHNAN | 3122225002109 | IT - C

Q1).Create a class Point with xcod and ycod as data members. Include member functions to read and print the coordinate values as getPoint() and showPoint() respectively. Write a main method to demonstrate the Point class

```
# -*- coding: utf-8 -*-
"""
This module provides code for Point class. This is a part
of the exercises given under the course UIT2312 (Programming
and Design Patterns).

In this source code I've executed my own logic and may contain
bugs.
The source code has followed good coding practices.

Your comments and suggestions are welcome.

Created on Fri Sept 15 2023

Revised on Wed Sept 21 2023

Original Author: T. Sadakopa Ramakrishnan
<sadakopa2210221@ssn.edu.in>
"""
```

```
class Point:
    def __init__(self, xcod = 0, ycod = 0):
        self.x = xcod
        self.y = ycod

    def getPoint(self):
        self.x = int(input("Enter x coordinate: "))
        self.y = int(input("Enter y coordinate: "))

    def showPoint(self):
        result = f"({self.x}, {self.y})"
        return result

if __name__ == "__main__":
    #Creating a Point Object
    P = Point()

    #Getting x and y coordinates
    P.getPoint()

    #Displaying x and y coordinates
    print(P.showPoint())
```

```
Enter x coordinate: 4
Enter y coordinate: 5
(4, 5)
```

Q2a) Create a class called Circle with radius as data member and getCircle(), calcRad, calcArea() as member functions to read two points, 1.centre, 2.any point in orbit and calculate the radius and area of the circle.
Q2b) Create another class called Cone which is derived from the Circle class. Have a data member apex and utilize the existing data members and the member functions of the base class by the derived class to find the volume ((1/3)*3.14*r*r*h) of a cone.

```
# -*- coding: utf-8 -*-
"""
This module provides code for two classes Circle and Cone
which is inherited from Circle. This is a part
of the exercises given under the course UIT2312 (Programming
and Design Patterns).

In this source code I've executed my own logic and may contain
bugs.
The source code has followed good coding practices.

Your comments and suggestions are welcome.

Created on Fri Sept 15 2023

Revised on Wed Sept 21 2023

Original Author: T. Sadakopa Ramakrishnan
<sadakopa2210221@ssn.edu.in>
"""
```

```
class Circle:
    def __init__(self):
        self.radius = 0.0

    def getCircle(self):
        print("Enter the coordinates of Centre")
        x1 = int(input("Enter x coordinate of center: "))
        y1 = int(input("Enter y coordinate of center: "))

        print("Enter the coordinates of any point on circumference")
        x2 = int(input("Enter x coordinate of point: "))
        y2 = int(input("Enter y coordinate of point: "))

        self.calcRad(x1,y1,x2,y2)
        self.calcArea()

    def calcRad(self, x1, y1, x2, y2):
        self.radius = (((x2 - x1) ** 2) + ((y2 - y1) ** 2)) ** 0.5

    def calcArea(self):
        area = 3.14 * (self.radius ** 2)
        print(f"The radius of the circle is {self.radius}")
        print(f"Area of circle is {area}")

class Cone(Circle):
    def __init__(self):
        super().__init__()
        self.apex = None
```

```
    def getCone(self):
        self.getCircle()
        print("Enter the coordinates of apex")
        x = int(input("Enter the x coordinates of apex: "))
        y = int(input("Enter the y coordinate of apex: "))
        self.apex = (x, y)

    def calcVolume(self):
        height = ((self.radius ** 2) + (self.apex[1]**2)) ** 0.5
        volume = (1/3) * 3.14 * (self.radius**2) * (height)
        print(f"Apex Coordinates: {self.apex}")
        print(f"Height of the cone: {height}")
        print(f"Volume of the cone: {volume}")


if __name__ == "__main__":
    print("CIRCLE!!!")
    #Creating a circle object
    circle = Circle()

    #Printing details
    circle.getCircle()

    print("_____")
    print("CONE!!!")
    #Creating a Cone object
    cone = Cone()
```

#Printing details

cone.getCone()

cone.calcVolume()

```
CIRCLE!!!
Enter the coordinates of Centre
Enter x coordinate of center: 0
Enter y coordinate of center: 0
Enter the coordinates of any point on circumference
Enter x coordinate of point: 3
Enter y coordinate of point: 4
The radius of the circle is 5.0
Area of circle is 78.5
```

```
CONE!!!
Enter the coordinates of Centre
Enter x coordinate of center: 0
Enter y coordinate of center: 0
Enter the coordinates of any point on circumference
Enter x coordinate of point: 1
Enter y coordinate of point: 2
The radius of the circle is 2.23606797749979
Area of circle is 15.700000000000003
Enter the coordinates of apex
Enter the x coordinates of apex: 3
Enter the y coordinate of apex: 4
Apex Coordinates: (3, 4)
Height of the cone: 4.58257569495584
Volume of the cone: 23.982146136935565
```

Q3a) Inherit Regular_Polygon from point. Have points_array and num_sides as a data members and have function for getting details.
Q3b) Inherit any polygon from Regular_Polygon Like square, have additional functions to calculate area, perimeter and volume.

```
# -*- coding: utf-8 -*-
"""
This module provides code for two classes Regular_Polygon
and Square which is inherited from Regular_Polygon. This is a part
of the exercises given under the course UIT2312 (Programming
and Design Patterns).

In this source code I've executed my own logic and may contain
bugs.
The source code has followed good coding practices.

Your comments and suggestions are welcome.

Created on Fri Sept 15 2023

Revised on Wed Sept 21 2023

Original Author: T. Sadakopa Ramakrishnan
<sadakopa2210221@ssn.edu.in>
"""


from ex02_1 import Point
```

```
class Regular_Polygon(Point):
    def __init__(self):
        super().__init__()
        self.points_array = []
        self.num_sides = 0

    def getPolygon(self):
        self.num_sides = int(input("Enter the number of sides: "))
        print("Enter the coordinates of polygon's vertices")
        for i in range(self.num_sides):
            point = Point()
            point.getPoint()
            self.points_array.append(point)

    def showPolygonDetails(self):
        print(f"Number of sides: {self.num_sides}")
        print("Coordinates of the polygon's vertices:")
        for i, point in enumerate(self.points_array, start=1):
            print(f"Vertex {i}: ({point.x}, {point.y})")

class Square(Regular_Polygon):
    def __init__(self):
        super().__init__()  # Call the constructor of the base class
(Regular_Polygon)
        self.side_length = 0

    def getSquareDetails(self):
        self.getPolygon()  # Reuse the getDetails method from the
Regular_Polygon class
```

```
        self.side_length = float(input("Enter the side length of the
square: "))

    def calcArea(self):
        area = self.side_length ** 2
        return area

    def calcPerimeter(self):
        perimeter = self.side_length * self.num_sides
        return perimeter


if __name__ == "__main__":
    print("Polygon!!!")
    # Create an instance of the Regular_Polygon class
    polygon = Regular_Polygon()

    # Get details of the regular polygon
    polygon.getPolygon()

    # Display the polygon details
    polygon.showPolygonDetails()

    print("_____\n")
    print("SQUARE!!!")
    # Create an instance of the Square class
    square = Square()

    # Get details of the square
```

```
square.getSquareDetails()

# Display the square details
square.showPolygonDetails()
area = square.calcArea()
perimeter = square.calcPerimeter()
print(f"Area of the square: {area:.2f}")
print(f"Perimeter of the square: {perimeter:.2f}")
```

```
Polygon!!!
Enter the number of sides: 3
Enter the coordinates of polygon's vertices
Enter x coordinate: 1
Enter y coordinate: 1
Enter x coordinate: 2
Enter y coordinate: 2
Enter x coordinate: 3
Enter y coordinate: 3
Number of sides: 3
Coordinates of the polygon's vertices:
Vertex 1: (1, 1)
Vertex 2: (2, 2)
Vertex 3: (3, 3)
```

```
SQUARE!!!
Enter the number of sides: 4
Enter the coordinates of polygon's vertices
Enter x coordinate: 1
Enter y coordinate: 1
Enter x coordinate: 2
Enter y coordinate: 2
Enter x coordinate: 3
Enter y coordinate: 3
Enter x coordinate: 4
Enter y coordinate: 4
Enter the side length of the square: 5
Number of sides: 4
Coordinates of the polygon's vertices:
Vertex 1: (1, 1)
Vertex 2: (2, 2)
Vertex 3: (3, 3)
Vertex 4: (4, 4)
Area of the square: 25.00
Perimeter of the square: 20.00
```