// Kushal Bansal----IIT2022021 // I have pasted the code with formatting so as to get the better view.

**Sol.1**: There is a compilation error if we try to inherit class “PuppyDog” from classes “Animal” and “Dog” as multiple inheritance is not supported in Java. Yes, it can be done by the concept of Interfaces in Java. The code is attached below:

interface Animal {

    void speak();

}

class Dog implements Animal {

    @Override

    public void speak() {

        System.out.println("Woof!");

    }

}

class PuppyDog extends Dog implements Animal {

    private String name;

    public PuppyDog(String name) {

        this.name = name;

    }

    @Override

    public void speak() {

        System.out.println("Yip yip!");

    }

    public String getName() {

        return name;

    }

}

public class ExecAnimal {

    public static void main(String[] args) {

        PuppyDog puppy = new PuppyDog("Bruno");

        System.out.println("Name: " + puppy.getName());

        System.out.print("Dog says: ");puppy.speak();

    }

}

**Sol.2:** This code would show the output as:

running

5

running safely with 60km

5

running safely with 60km

10

**Sol.3**: The code is attached below:

abstract class Shape {

    String color;

    public Shape(String color) {

        this.color = color;

    }

    public abstract double calculateArea();

    public abstract double calculatePerimeter();

    String getColor() {

        return color;

    }

}

class Circle extends Shape {

    private double radius;

    Circle(String color, double radius) {

        super(color);

        this.radius = radius;

    }

    @Override

    public double calculateArea() {

        return Math.PI \* radius \* radius;

    }

    @Override

    public double calculatePerimeter() {

        return 2 \* Math.PI \* radius;

    }

}

class Rectangle extends Shape {

    private double length;

    private double width;

    Rectangle(String color, double length, double width) {

        super(color);

        this.length = length;

        this.width = width;

    }

    @Override

    public double calculateArea() {

        return length \* width;

    }

    @Override

    public double calculatePerimeter() {

        return 2 \* (length + width);

    }

}

class Square extends Rectangle {

    public Square(String color, double sideLength) {

        super(color, sideLength, sideLength);

    }

}

public class Main {

    public static void main(String[] args) {

        Circle circle = new Circle("Red", 5.0);

        Rectangle rectangle = new Rectangle("Blue", 4.0, 6.0);

        Square square = new Square("Green", 5.0);

        System.out.println("Circle Color: " + circle.getColor() + ", Area: " + circle.calculateArea() + ", Perimeter: " + circle.calculatePerimeter());

        System.out.println("Rectangle Color: " + rectangle.getColor() + ", Area: " + rectangle.calculateArea() + ", Perimeter: " + rectangle.calculatePerimeter());

        System.out.println("Square Color: " + square.getColor() + ", Area: " + square.calculateArea() + ", Perimeter: " + square.calculatePerimeter());

    }

}

**Sol.4:** The code is provided below:

class Animal {

    public void announce() {

        System.out.println("I am an animal.");

    }

}

class Dog extends Animal {

    @Override

    public void announce() {

        System.out.println("I am a dog.");

    }

}

class PuppyDog extends Dog{}

public class Main {

    public static void main(String[] args) {

        PuppyDog puppy = new PuppyDog();

        puppy.announce();

    }

}

The result is “I am a Dog” as the “PuppyDog” class inherits the overridden announce method from its parent class “Dog”.

**Sol.5**: The output of the provided code should be:

111 NAME1 IIITA

222 NAME2 IIITA

333 NAME3 IIITA