Selected files

5 printable files

```
2B\Circle.java
2B\Rectangle.java
2B\Shape.java
2B\Square.java
2B\TestShape.java
```

2B\Circle.java

```java
public class Circle extends Shape {
    // Instance variable
    protected double radius;

    // Constructors
    public Circle() {
        this.radius = 1.0;
    }

    public Circle(double radius) {
        this.radius = radius;
    }

    public Circle(double radius, String color, boolean filled) {
        super(color, filled);
        this.radius = radius;
    }

    // Getter and setter for radius
    public double getRadius() {
        return radius;
    }

    public void setRadius(double radius) {
        this.radius = radius;
    }

    // Methods
    public double getArea() {
        return Math.PI * radius * radius;
    }

    public double getPerimeter() {
        return 2 * Math.PI * radius;
    }

    // Override toString method
    @Override
    public String toString() {
        return "Circle[" + super.toString() + ", radius=" + radius + "]";
    }
}
```

2B\Rectangle.java

```java
public class Rectangle extends Shape {
    // Instance variables
    protected double width;
    protected double length;

    // Constructors
    public Rectangle() {
        this.width = 1.0;
        this.length = 1.0;
    }

    public Rectangle(double width, double length) {
        this.width = width;
        this.length = length;
    }

    public Rectangle(double width, double length, String color, boolean filled) {
        super(color, filled);
        this.width = width;
        this.length = length;
    }

    // Getter and setter for width and length
    public double getWidth() {
        return width;
    }

    public void setWidth(double width) {
        this.width = width;
    }

    public double getLength() {
        return length;
    }

    public void setLength(double length) {
        this.length = length;
    }

    // Methods
    public double getArea() {
        return width * length;
    }

    public double getPerimeter() {
        return 2 * (width + length);
    }

    // Override toString method
    @Override
    public String toString() {
        return "Rectangle[" + super.toString() + ", width=" + width + ", length=" + length + "]";
    }
}
```

2B\Shape.java

```java
public abstract class Shape {
    // Instance variables
    protected String color;
    protected boolean filled;

    // Constructors
    public Shape() {
        this.color = "green";
        this.filled = true;
    }

    public Shape(String color, boolean filled) {
        this.color = color;
        this.filled = filled;
    }

    // Getters and setters
    public String getColor() {
        return color;
    }

    public void setColor(String color) {
        this.color = color;
    }

    public boolean isFilled() {
        return filled;
    }

    public void setFilled(boolean filled) {
        this.filled = filled;
    }

    public abstract double getArea();

    public abstract double getPerimeter();

    // toString method
    @Override
    public String toString() {
        return "Shape[color=" + color + ", filled=" + filled + "]";
    }
}
```

2B\Square.java

```java
public class Square extends Rectangle {
    // Constructors
    public Square() {
        super();
    }

    public Square(double side) {
        super(side, side);
    }

    public Square(double side, String color, boolean filled) {
        super(side, side, color, filled);
    }

    // Override toString method
    @Override
    public String toString() {
        return "Square[" + super.toString() + "]";
    }

    // Override setWidth and setLength to maintain square geometry
    @Override
    public void setWidth(double side) {
        super.setWidth(side);
        super.setLength(side);
    }

    @Override
    public void setLength(double side) {
        super.setWidth(side);
        super.setLength(side);
    }
}
```

2B\TestShape.java

```java
public class TestShape {
    public static void main(String[] args) {
        // Upcasting Circle to Shape
        Shape s1 = new Circle(5.5, "red", false);
        System.out.println(s1); // calls Circle's toString()
        System.out.println("Area: " + s1.getArea()); // calls Circle's getArea()
        System.out.println("Perimeter: " + s1.getPerimeter()); // calls Circle's getPerimeter()
        System.out.println("Color: " + s1.getColor()); // calls Shape's getColor()
        System.out.println("Filled: " + s1.isFilled()); // calls Shape's isFilled()

        // Downcasting back to Circle
        Circle c1 = (Circle) s1;
        System.out.println(c1); // calls Circle's toString()
        System.out.println("Area: " + c1.getArea()); // calls Circle's getArea()
        System.out.println("Perimeter: " + c1.getPerimeter()); // calls Circle's getPerimeter()
        System.out.println("Color: " + c1.getColor()); // calls Shape's getColor()
        System.out.println("Filled: " + c1.isFilled()); // calls Shape's isFilled()
        System.out.println("Radius: " + c1.getRadius()); // calls Circle's getRadius()

        // Upcasting Rectangle to Shape
        Shape s2 = new Rectangle(1.0, 2.0, "red", false);
        System.out.println(s2); // calls Rectangle's toString()
        System.out.println("Area: " + s2.getArea()); // calls Rectangle's getArea()
        System.out.println("Perimeter: " + s2.getPerimeter()); // calls Rectangle's getPerimeter()
        System.out.println("Color: " + s2.getColor()); // calls Shape's getColor()

        // Downcasting back to Rectangle
        Rectangle r1 = (Rectangle) s2;
        System.out.println(r1); // calls Rectangle's toString()
        System.out.println("Area: " + r1.getArea()); // calls Rectangle's getArea()
        System.out.println("Color: " + r1.getColor()); // calls Shape's getColor()
        System.out.println("Length: " + r1.getLength()); // calls Rectangle's getLength()

        // Upcasting Square to Shape
        Shape s3 = new Square(6.6);
        System.out.println(s3); // calls Square's toString()
        System.out.println("Area: " + s3.getArea()); // calls Square's getArea()
        System.out.println("Color: " + s3.getColor()); // calls Shape's getColor()
        System.out.println("Side: " + ((Square) s3).getWidth()); // Downcasting to access getWidth()
    }
}
```