Selected files

5 printable files

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

2B\Circle.java

```
1
    public class Circle extends Shape {
 2
        // Instance variable
 3
        protected double radius;
 4
 5
        // Constructors
 6
        public Circle() {
 7
            this.radius = 1.0;
 8
 9
10
        public Circle(double radius) {
            this.radius = radius;
11
12
13
        public Circle(double radius, String color, boolean filled) {
14
15
             super(color, filled);
            this.radius = radius;
16
17
        }
18
19
        // Getter and setter for radius
20
        public double getRadius() {
21
            return radius;
22
        }
23
24
        public void setRadius(double radius) {
25
            this.radius = radius;
26
27
28
        // Methods
29
        public double getArea() {
             return Math.PI * radius * radius;
30
31
32
33
        public double getPerimeter() {
34
            return 2 * Math.PI * radius;
35
36
37
        // Override toString method
38
        @Override
39
        public String toString() {
            return "Circle[" + super.toString() + ", radius=" + radius + "]";
40
41
42 }
```

2B\Rectangle.java

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

```
protected double length;
 5
 6
         // Constructors
 7
         public Rectangle() {
 8
             this.width = 1.0;
 9
             this.length = 1.0;
 10
 11
         public Rectangle(double width, double length) {
12
13
             this.width = width;
 14
             this.length = length;
15
         }
 16
17
         public Rectangle(double width, double length, String color, boolean filled) {
 18
             super(color, filled);
 19
             this.width = width;
 20
             this.length = length;
 21
         }
22
 23
         // Getter and setter for width and length
         public double getWidth() {
 24
 25
             return width;
 26
27
 28
         public void setWidth(double width) {
 29
             this.width = width;
 30
 31
         public double getLength() {
 32
 33
             return length;
 34
 35
 36
         public void setLength(double length) {
 37
             this.length = length;
 38
         }
 39
40
         // Methods
         public double getArea() {
41
42
             return width * length;
43
         }
44
45
         public double getPerimeter() {
46
             return 2 * (width + length);
47
48
49
         // Override toString method
         @Override
 50
 51
         public String toString() {
52
             return "Rectangle[" + super.toString() + ", width=" + width + ", length=" + length +
53
54 }
2B\Shape.java
```

```
public abstract class Shape {
1
2
       // Instance variables
3
       protected String color;
4
       protected boolean filled;
```

```
5
 6
        // Constructors
 7
        public Shape() {
            this.color = "green";
 8
 9
            this.filled = true;
10
        }
11
12
        public Shape(String color, boolean filled) {
13
            this.color = color;
            this.filled = filled;
14
15
        }
16
        // Getters and setters
17
18
        public String getColor() {
19
            return color;
20
        }
21
22
        public void setColor(String color) {
23
            this.color = color;
24
25
        public boolean isFilled() {
26
            return filled;
27
28
29
30
        public void setFilled(boolean filled) {
            this.filled = filled;
31
32
        }
33
34
        public abstract double getArea();
35
36
        public abstract double getPerimeter();
37
        // toString method
38
39
        @Override
        public String toString() {
40
41
            return "Shape[color=" + color + ", filled=" + filled + "]";
42
43
   }
```

2B\Square.java

```
public class Square extends Rectangle {
 1
 2
        // Constructors
 3
        public Square() {
 4
            super();
 5
 6
 7
        public Square(double side) {
 8
            super(side, side);
9
10
        public Square(double side, String color, boolean filled) {
11
            super(side, side, color, filled);
12
13
14
15
        // Override toString method
16
        @Override
        public String toString() {
17
```

```
return "Square[" + super.toString() + "]";
19
20
21
        // Override setWidth and setLength to maintain square geometry
22
        @Override
23
        public void setWidth(double side) {
24
            super.setWidth(side);
25
            super.setLength(side);
26
27
28
        @Override
29
        public void setLength(double side) {
            super.setWidth(side);
30
31
            super.setLength(side);
32
33 }
```

2B\TestShape.java

```
1
    public class TestShape {
        public static void main(String[] args) {
 2
 3
            // Upcasting Circle to Shape
            Shape s1 = new Circle(5.5, "red", false);
 4
 5
            System.out.println(s1); // calls Circle's toString()
            System.out.println("Area: " + s1.getArea()); // calls Circle's getArea()
6
            System.out.println("Perimeter: " + s1.getPerimeter()); // calls Circle's
 7
    getPerimeter()
8
            System.out.println("Color: " + s1.getColor()); // calls Shape's getColor()
            System.out.println("Filled: " + s1.isFilled()); // calls Shape's isFilled()
9
10
            // Downcasting back to Circle
11
            Circle c1 = (Circle) s1;
12
13
            System.out.println(c1); // calls Circle's toString()
            System.out.println("Area: " + c1.getArea()); // calls Circle's getArea()
14
            System.out.println("Perimeter: " + c1.getPerimeter()); // calls Circle's
15
    getPerimeter()
            System.out.println("Color: " + c1.getColor()); // calls Shape's getColor()
16
            System.out.println("Filled: " + c1.isFilled()); // calls Shape's isFilled()
17
            System.out.println("Radius: " + c1.getRadius()); // calls Circle's getRadius()
18
19
20
            // Upcasting Rectangle to Shape
            Shape s2 = new Rectangle(1.0, 2.0, "red", false);
21
            System.out.println(s2); // calls Rectangle's toString()
22
            System.out.println("Area: " + s2.getArea()); // calls Rectangle's getArea()
23
            System.out.println("Perimeter: " + s2.getPerimeter()); // calls Rectangle's
24
    getPerimeter()
            System.out.println("Color: " + s2.getColor()); // calls Shape's getColor()
25
26
27
            // Downcasting back to Rectangle
28
            Rectangle r1 = (Rectangle) s2;
29
            System.out.println(r1); // calls Rectangle's toString()
            System.out.println("Area: " + r1.getArea()); // calls Rectangle's getArea()
30
            System.out.println("Color: " + r1.getColor()); // calls Shape's getColor()
31
32
            System.out.println("Length: " + r1.getLength()); // calls Rectangle's getLength()
33
            // Upcasting Square to Shape
34
35
            Shape s3 = new Square(6.6);
            System.out.println(s3); // calls Square's toString()
36
            System.out.println("Area: " + s3.getArea()); // calls Square's getArea()
37
            System.out.println("Color: " + s3.getColor()); // calls Shape's getColor()
38
```

```
System.out.println("Side: " + ((Square) s3).getWidth()); // Downcasting to access getWidth()
40
41 }
```