LAPORAN TUGAS MINGGU 6

Inheritance, Abstract Class and Interface in Java

Pemorgaman Berorientasi Objek

Resume ini disusun untuk memenuhi Tugas Mata Kuliah Pemrograman Berorientasi Objek

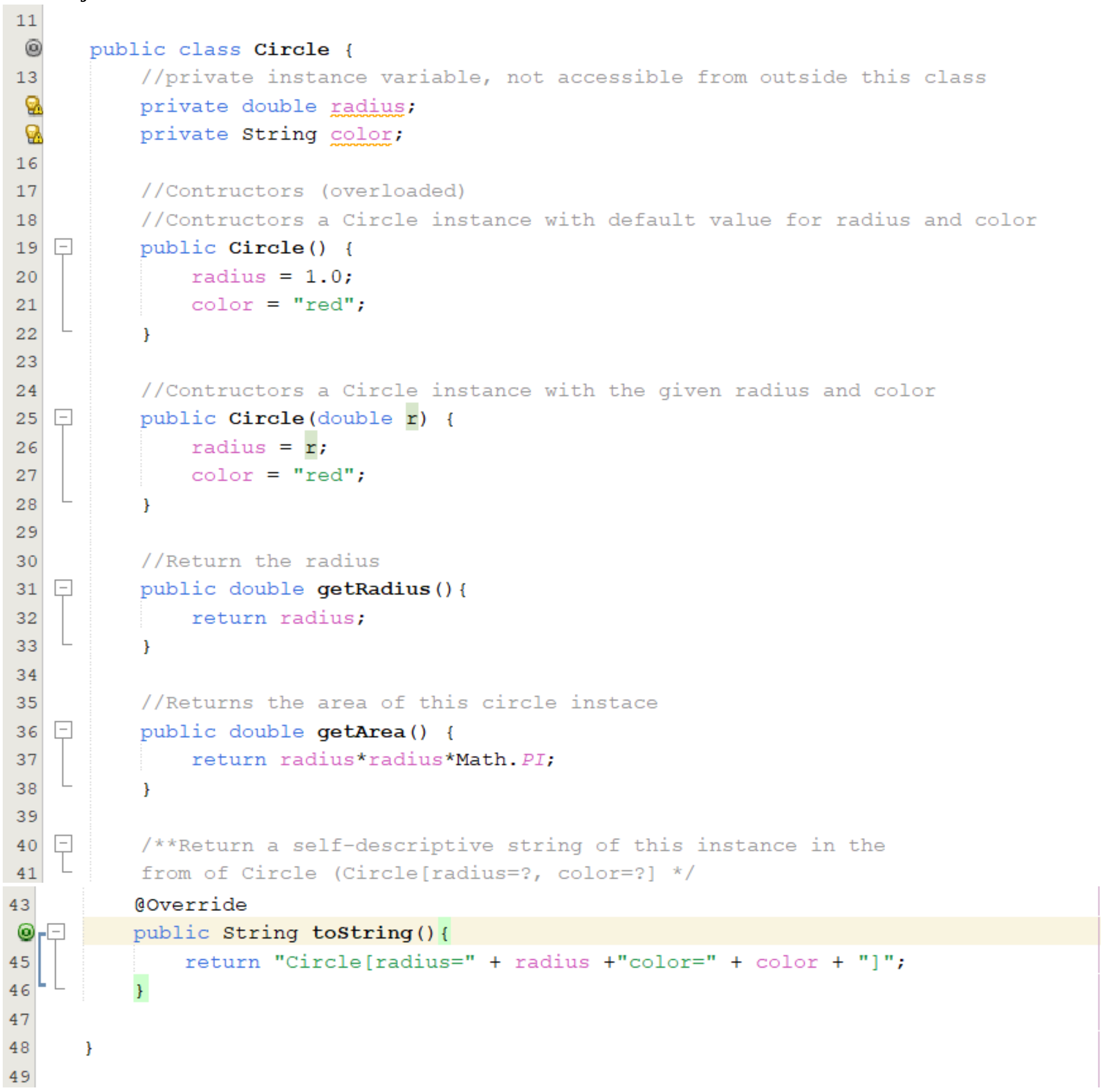
Disusun oleh:
Muhammad Rivan Rivaldi 211511048

PROGRAM STUDI D3 TEKNIK INFORMATIKA
JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA
POLITEKNIK NEGERI BANDUNG
2021

Exercise 1. The Circle and Cylinder Classes

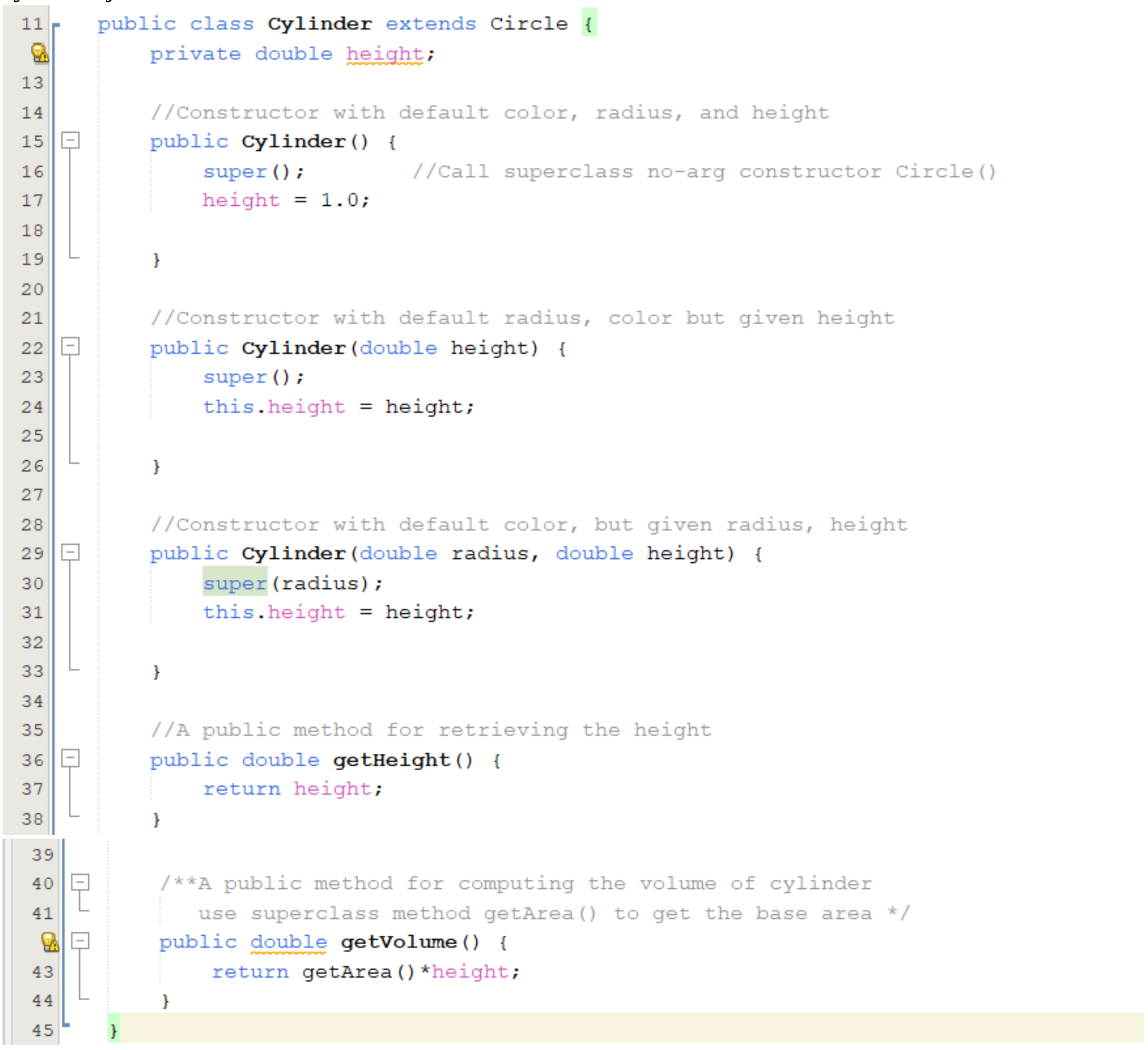
1. Source Code Asli

a) Circle.java



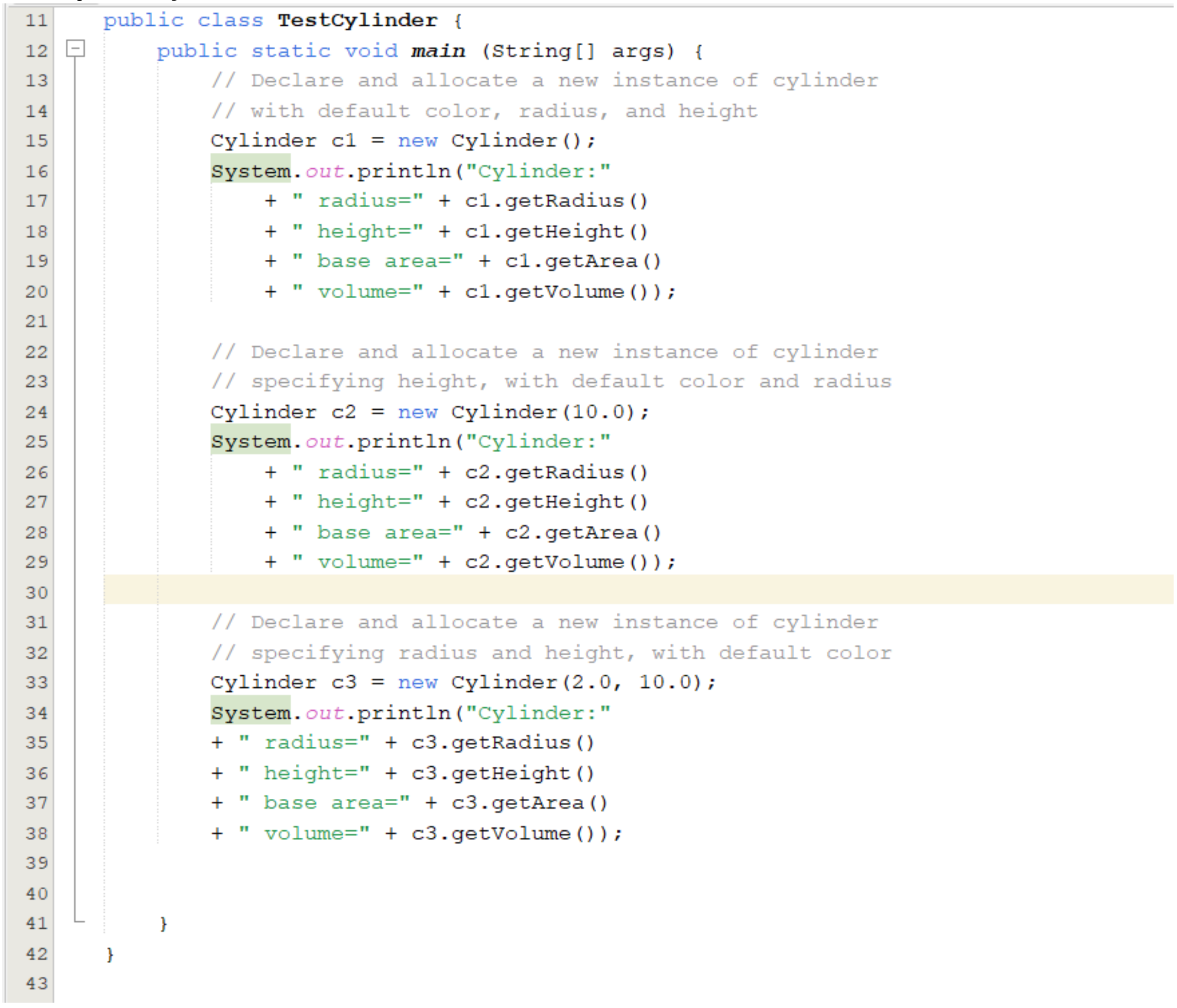
```java

public class Circle {
    //private instance variable, not accessible from outside this class
    private double radius;
    private String color;

    //Contructors (overloaded)
    //Contructors a Circle instance with default value for radius and color
    public Circle() {
        radius = 1.0;
        color = "red";
    }

    //Contructors a Circle instance with the given radius and color
    public Circle(double r) {
        radius = r;
        color = "red";
    }

    //Return the radius
    public double getRadius(){
        return radius;
    }

    //Returns the area of this circle instace
    public double getArea() {
        return radius*radius*Math.PI;
    }

    /**Return a self-descriptive string of this instance in the
    from of Circle (Circle[radius=?, color=?] */
    @Override
    public String toString(){
        return "Circle[radius=" + radius +"color=" + color + "]";
    }

}

```

b) Cylinder.java



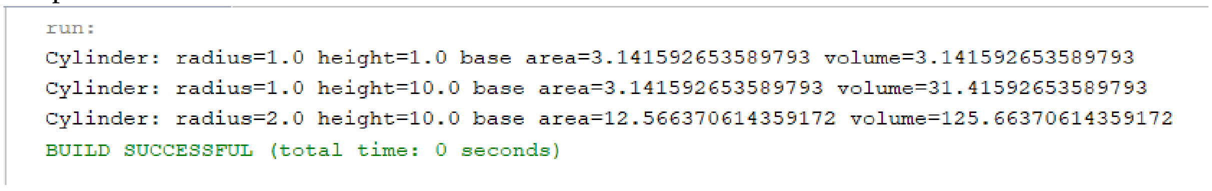
```java
public class Cylinder extends Circle {
    private double height;

    //Constructor with default color, radius, and height
    public Cylinder() {
        super();        //Call superclass no-arg constructor Circle()
        height = 1.0;

    }

    //Constructor with default radius, color but given height
    public Cylinder(double height) {
        super();
        this.height = height;

    }

    //Constructor with default color, but given radius, height
    public Cylinder(double radius, double height) {
        super(radius);
        this.height = height;

    }

    //A public method for retrieving the height
    public double getHeight() {
        return height;
    }

    /**A public method for computing the volume of cylinder
       use superclass method getArea() to get the base area */
    public double getVolume() {
        return getArea()*height;
    }
}
```

c) TestCylinder.java



```java
public class TestCylinder {
    public static void main (String[] args) {
        // Declare and allocate a new instance of cylinder
        // with default color, radius, and height
        Cylinder c1 = new Cylinder();
        System.out.println("Cylinder:"
            + " radius=" + c1.getRadius()
            + " height=" + c1.getHeight()
            + " base area=" + c1.getArea()
            + " volume=" + c1.getVolume());

        // Declare and allocate a new instance of cylinder
        // specifying height, with default color and radius
        Cylinder c2 = new Cylinder(10.0);
        System.out.println("Cylinder:"
            + " radius=" + c2.getRadius()
            + " height=" + c2.getHeight()
            + " base area=" + c2.getArea()
            + " volume=" + c2.getVolume());

        // Declare and allocate a new instance of cylinder
        // specifying radius and height, with default color
        Cylinder c3 = new Cylinder(2.0, 10.0);
        System.out.println("Cylinder:"
        + " radius=" + c3.getRadius()
        + " height=" + c3.getHeight()
        + " base area=" + c3.getArea()
        + " volume=" + c3.getVolume());


    }
}
```

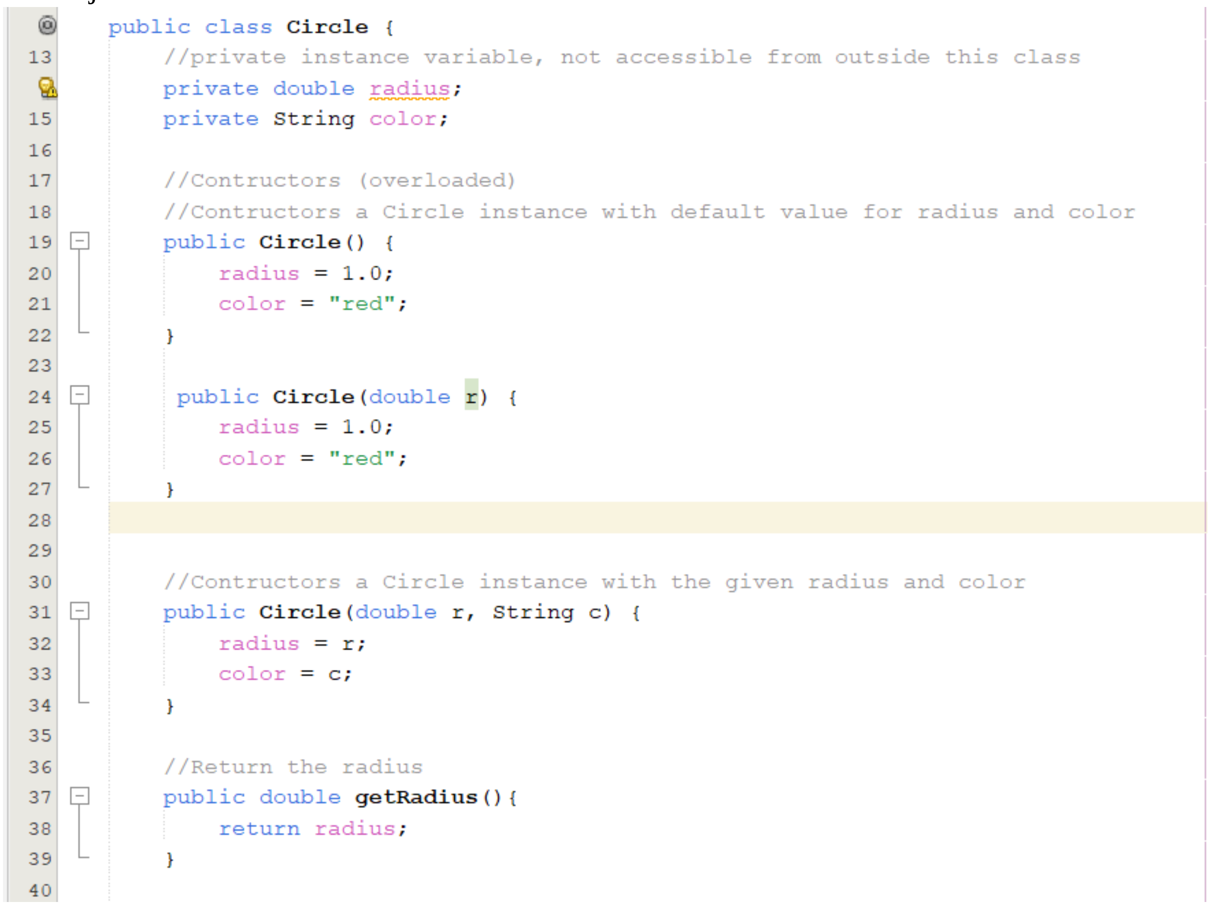
Output



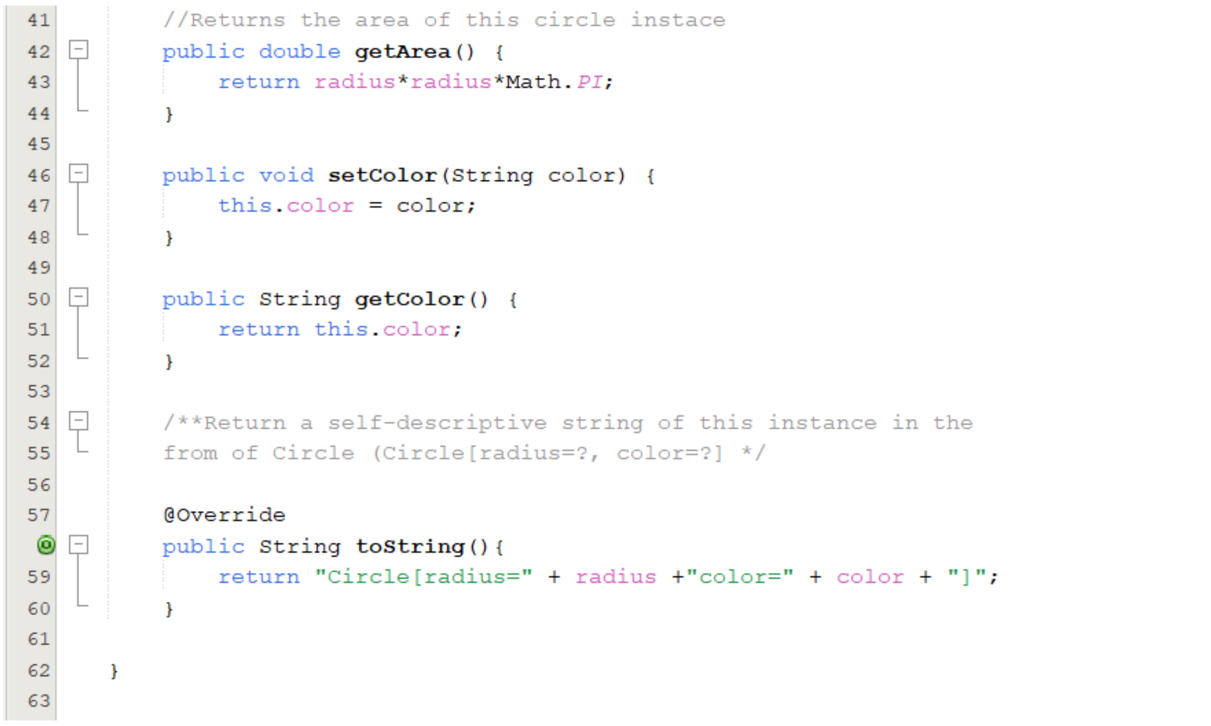
```
run:
Cylinder: radius=1.0 height=1.0 base area=3.141592653589793 volume=3.141592653589793
Cylinder: radius=1.0 height=10.0 base area=3.141592653589793 volume=31.41592653589793
Cylinder: radius=2.0 height=10.0 base area=12.566370614359172 volume=125.66370614359172
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Source Code yang Dimanipulasi

a) Circle.java

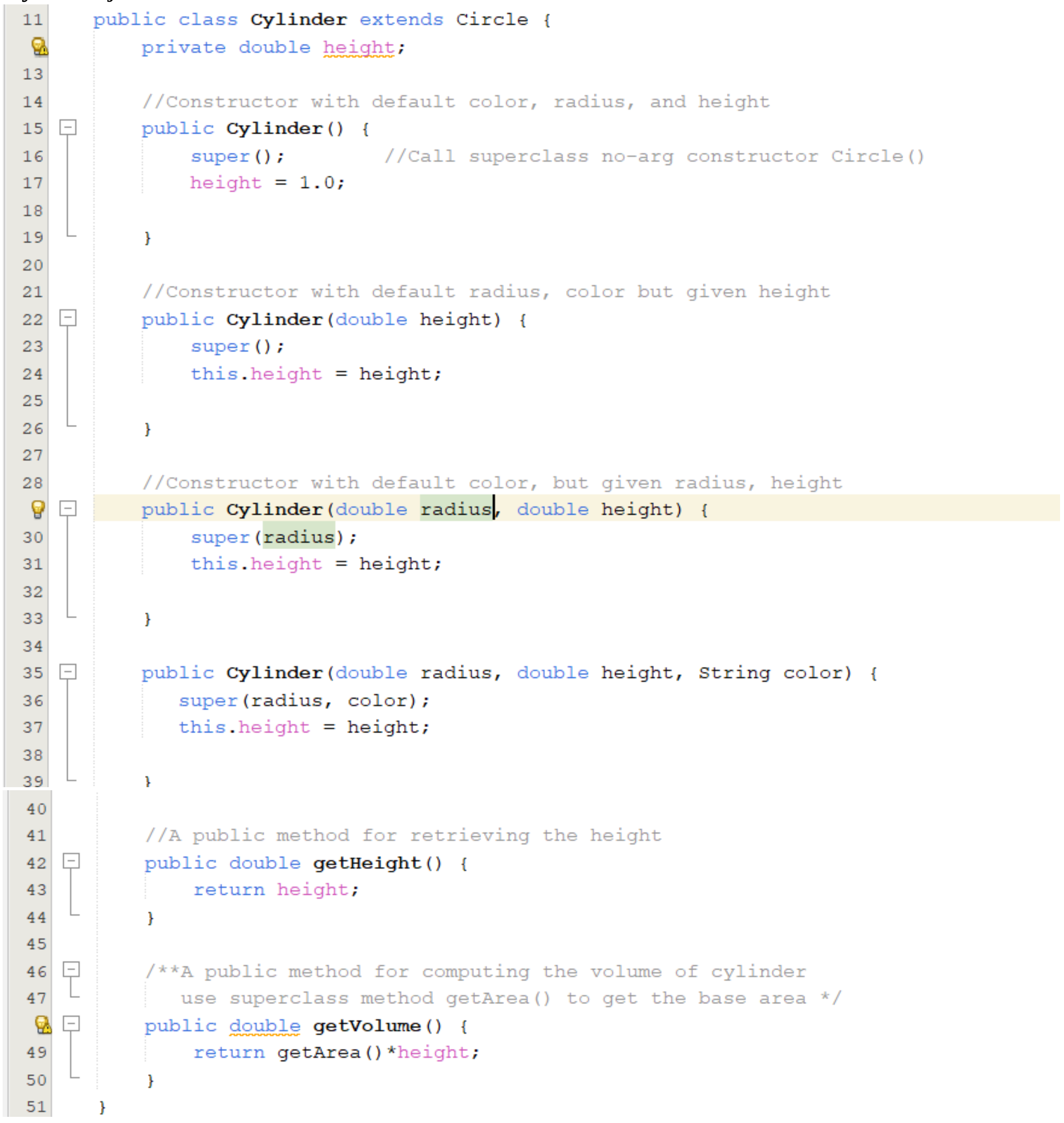


```java
public class Circle {
    //private instance variable, not accessible from outside this class
    private double radius;
    private String color;

    //Contructors (overloaded)
    //Contructors a Circle instance with default value for radius and color
    public Circle() {
        radius = 1.0;
        color = "red";
    }

     public Circle(double r) {
        radius = 1.0;
        color = "red";
    }



    //Contructors a Circle instance with the given radius and color
    public Circle(double r, String c) {
        radius = r;
        color = c;
    }

    //Return the radius
    public double getRadius(){
        return radius;
    }

```



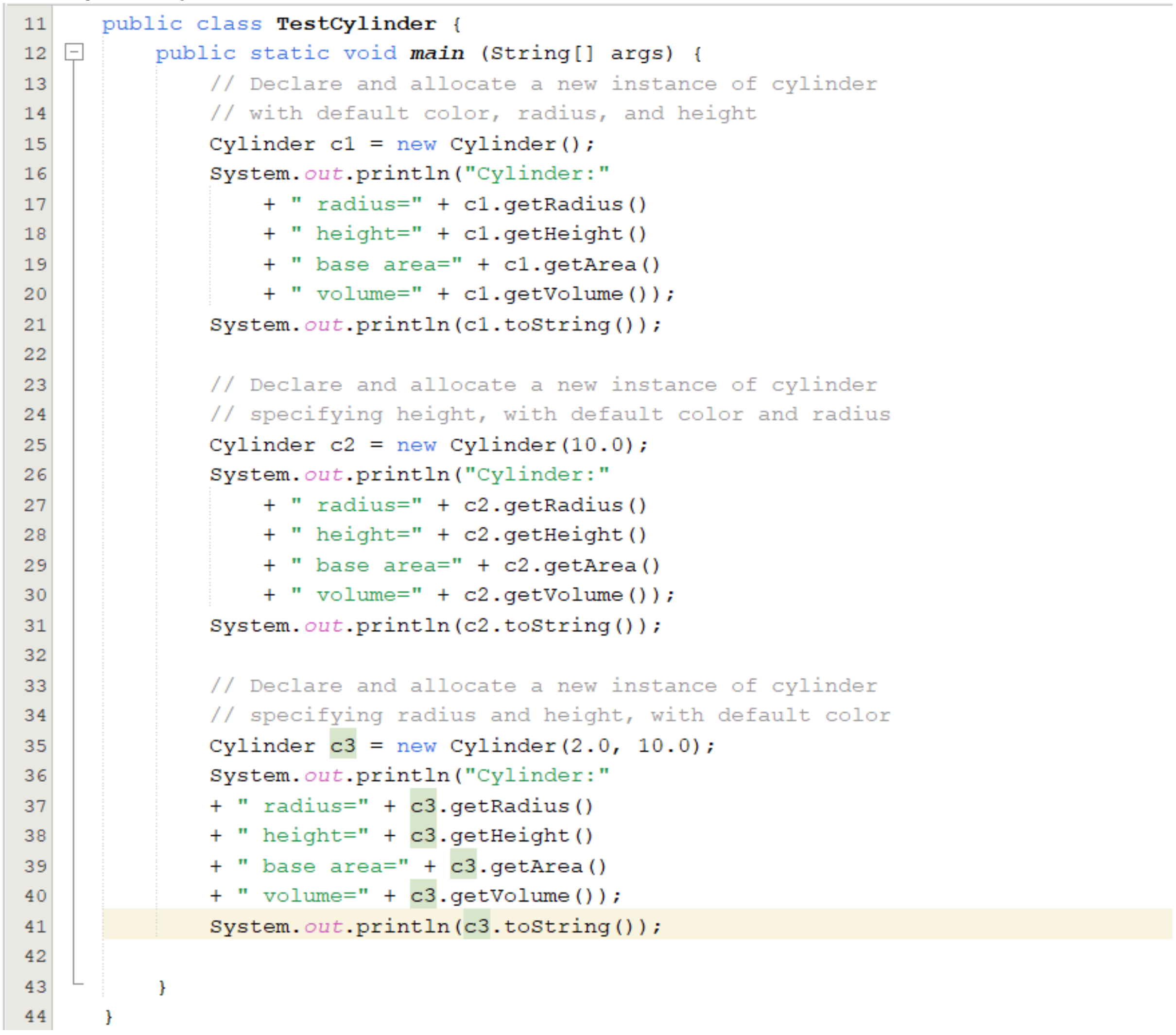
```java
    //Returns the area of this circle instace
    public double getArea() {
        return radius*radius*Math.PI;
    }

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

    public String getColor() {
        return this.color;
    }

    /**Return a self-descriptive string of this instance in the
    from of Circle (Circle[radius=?, color=?] */

    @Override
    public String toString(){
        return "Circle[radius=" + radius +"color=" + color + "]";
    }

}

```

b) Cylinder.java



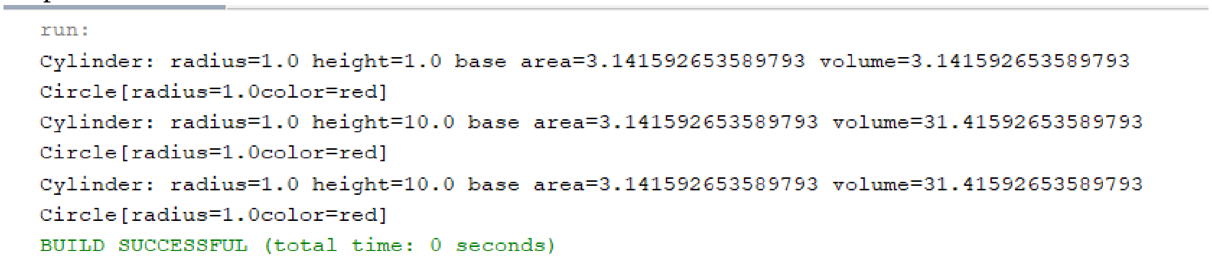
```
public class Cylinder extends Circle {
    private double height;

    //Constructor with default color, radius, and height
    public Cylinder() {
        super();        //Call superclass no-arg constructor Circle()
        height = 1.0;

    }

    //Constructor with default radius, color but given height
    public Cylinder(double height) {
        super();
        this.height = height;

    }

    //Constructor with default color, but given radius, height
    public Cylinder(double radius, double height) {
        super(radius);
        this.height = height;

    }

    public Cylinder(double radius, double height, String color) {
       super(radius, color);
       this.height = height;

    }

    //A public method for retrieving the height
    public double getHeight() {
        return height;
    }

    /**A public method for computing the volume of cylinder
       use superclass method getArea() to get the base area */
    public double getVolume() {
        return getArea()*height;
    }
}
```

c) TestCylinder.java



```java
public class TestCylinder {
    public static void main (String[] args) {
        // Declare and allocate a new instance of cylinder
        // with default color, radius, and height
        Cylinder c1 = new Cylinder();
        System.out.println("Cylinder:"
            + " radius=" + c1.getRadius()
            + " height=" + c1.getHeight()
            + " base area=" + c1.getArea()
            + " volume=" + c1.getVolume());
        System.out.println(c1.toString());

        // Declare and allocate a new instance of cylinder
        // specifying height, with default color and radius
        Cylinder c2 = new Cylinder(10.0);
        System.out.println("Cylinder:"
            + " radius=" + c2.getRadius()
            + " height=" + c2.getHeight()
            + " base area=" + c2.getArea()
            + " volume=" + c2.getVolume());
        System.out.println(c2.toString());

        // Declare and allocate a new instance of cylinder
        // specifying radius and height, with default color
        Cylinder c3 = new Cylinder(2.0, 10.0);
        System.out.println("Cylinder:"
        + " radius=" + c3.getRadius()
        + " height=" + c3.getHeight()
        + " base area=" + c3.getArea()
        + " volume=" + c3.getVolume());
        System.out.println(c3.toString());

    }
}
```

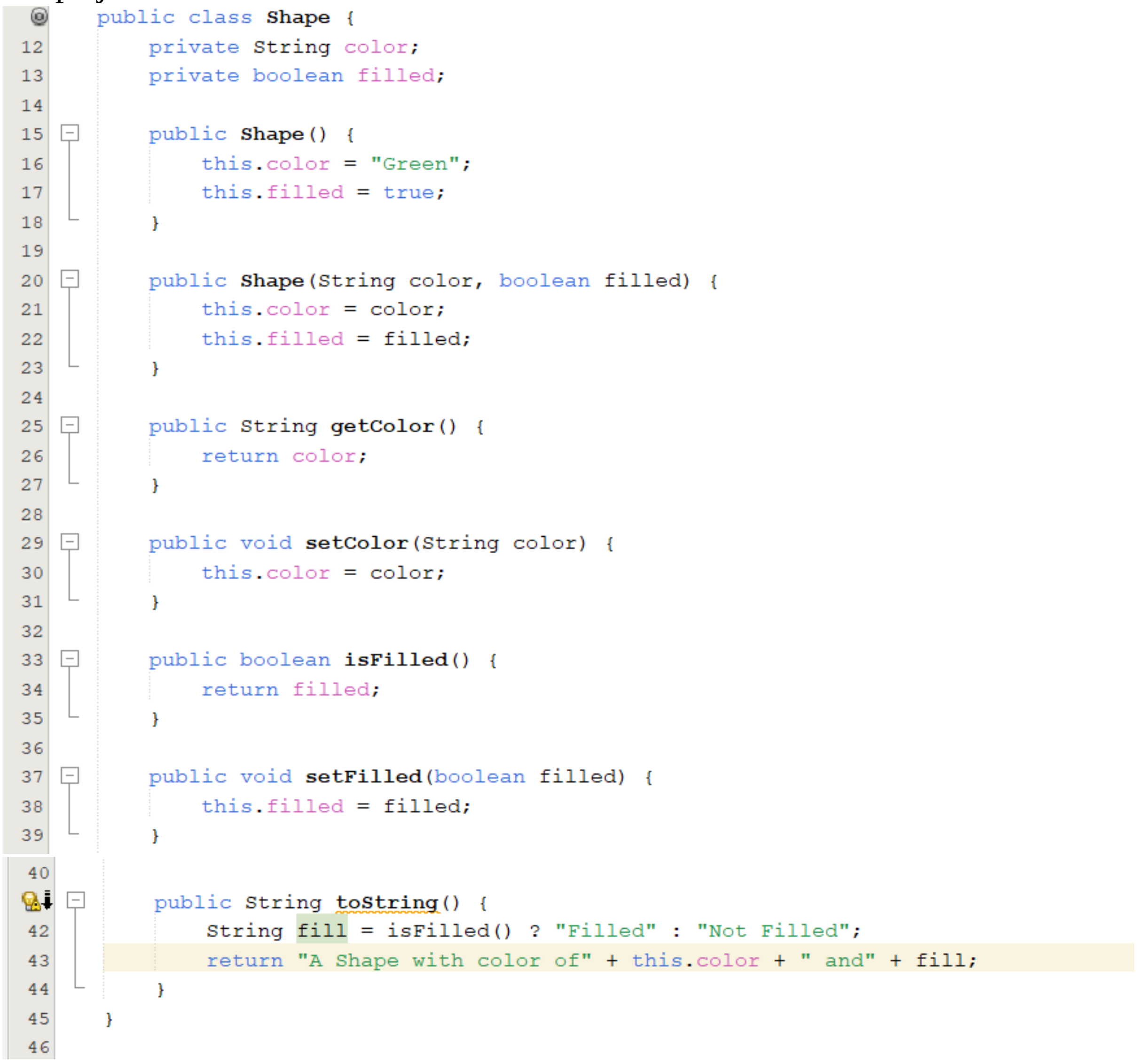
Ouput



```
run:
Cylinder: radius=1.0 height=1.0 base area=3.141592653589793 volume=3.141592653589793
Circle[radius=1.0color=red]
Cylinder: radius=1.0 height=10.0 base area=3.141592653589793 volume=31.41592653589793
Circle[radius=1.0color=red]
Cylinder: radius=1.0 height=10.0 base area=3.141592653589793 volume=31.41592653589793
Circle[radius=1.0color=red]
BUILD SUCCESSFUL (total time: 0 seconds)
```

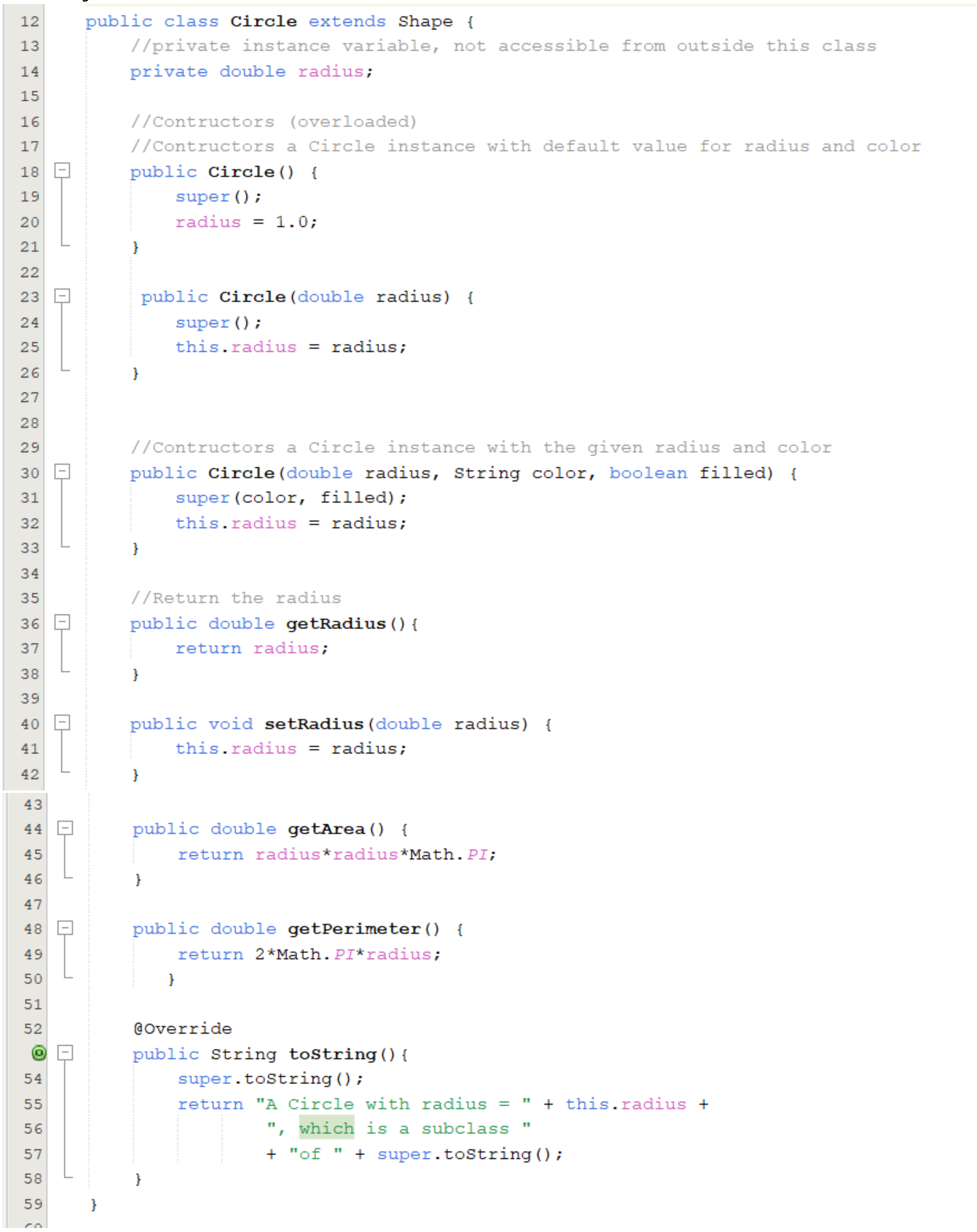
Exercise 2. Superclass Shape and its Subclasses Circle, Rectangle and Square

1. Shape.java



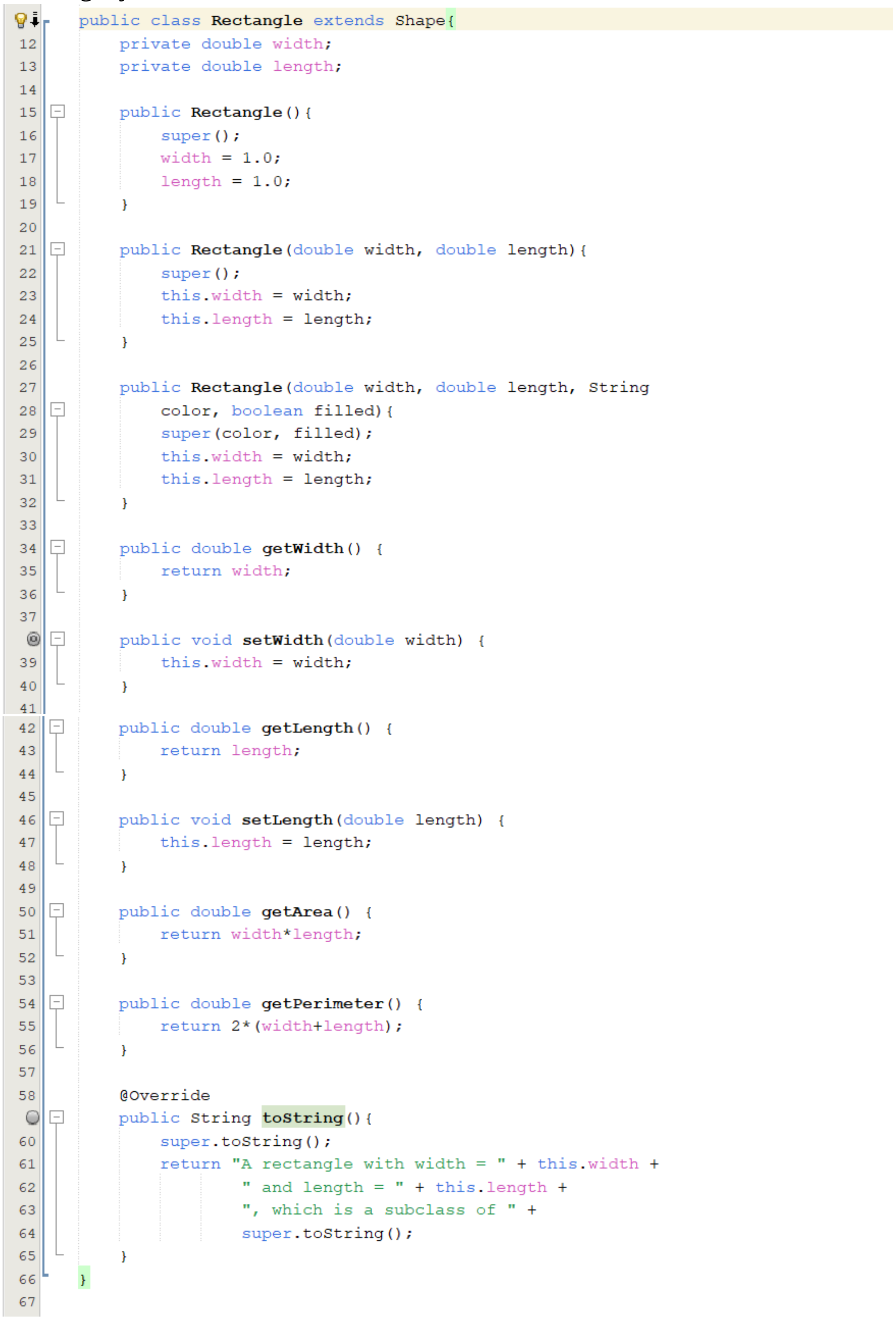
```java
public class Shape {
    private String color;
    private boolean filled;

    public Shape() {
        this.color = "Green";
        this.filled = true;
    }

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

    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 String toString() {
        String fill = isFilled() ? "Filled" : "Not Filled";
        return "A Shape with color of" + this.color + " and" + fill;
    }
}
```

2. Circle.java



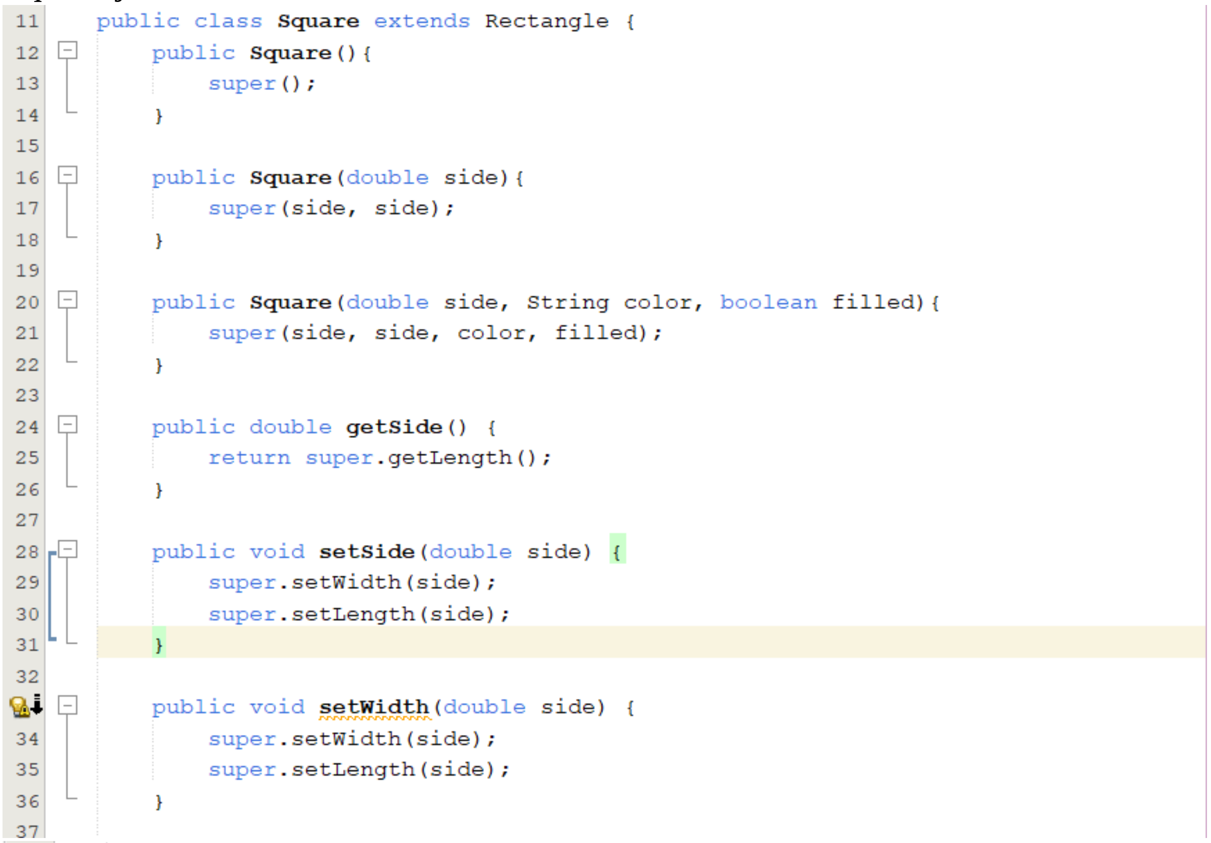
```java
public class Circle extends Shape {
    //private instance variable, not accessible from outside this class
    private double radius;

    //Contructors (overloaded)
    //Contructors a Circle instance with default value for radius and color
    public Circle() {
        super();
        radius = 1.0;
    }

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


    //Contructors a Circle instance with the given radius and color
    public Circle(double radius, String color, boolean filled) {
        super(color, filled);
        this.radius = radius;
    }

    //Return the radius
    public double getRadius(){
        return radius;
    }

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

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

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

    @Override
    public String toString(){
        super.toString();
        return "A Circle with radius = " + this.radius +
                ", which is a subclass "
                + "of " + super.toString();
    }
}
```

3. Rectangle.java

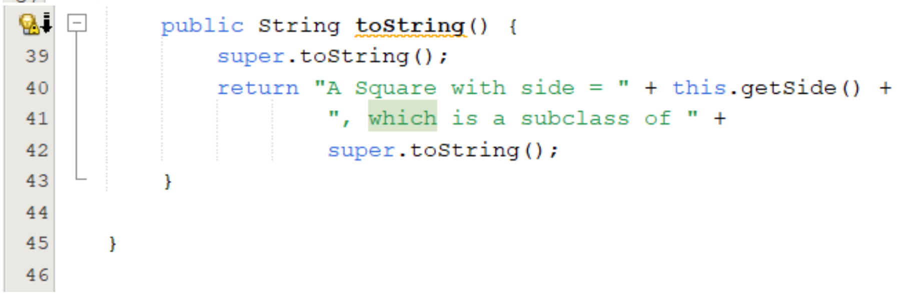


```java
public class Rectangle extends Shape{
    private double width;
    private double length;

    public Rectangle(){
        super();
        width = 1.0;
        length = 1.0;
    }

    public Rectangle(double width, double length){
        super();
        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;
    }

    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;
    }

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

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

    @Override
    public String toString(){
        super.toString();
        return "A rectangle with width = " + this.width +
                " and length = " + this.length +
                ", which is a subclass of " +
                super.toString();
    }
}
```

4. Square.java

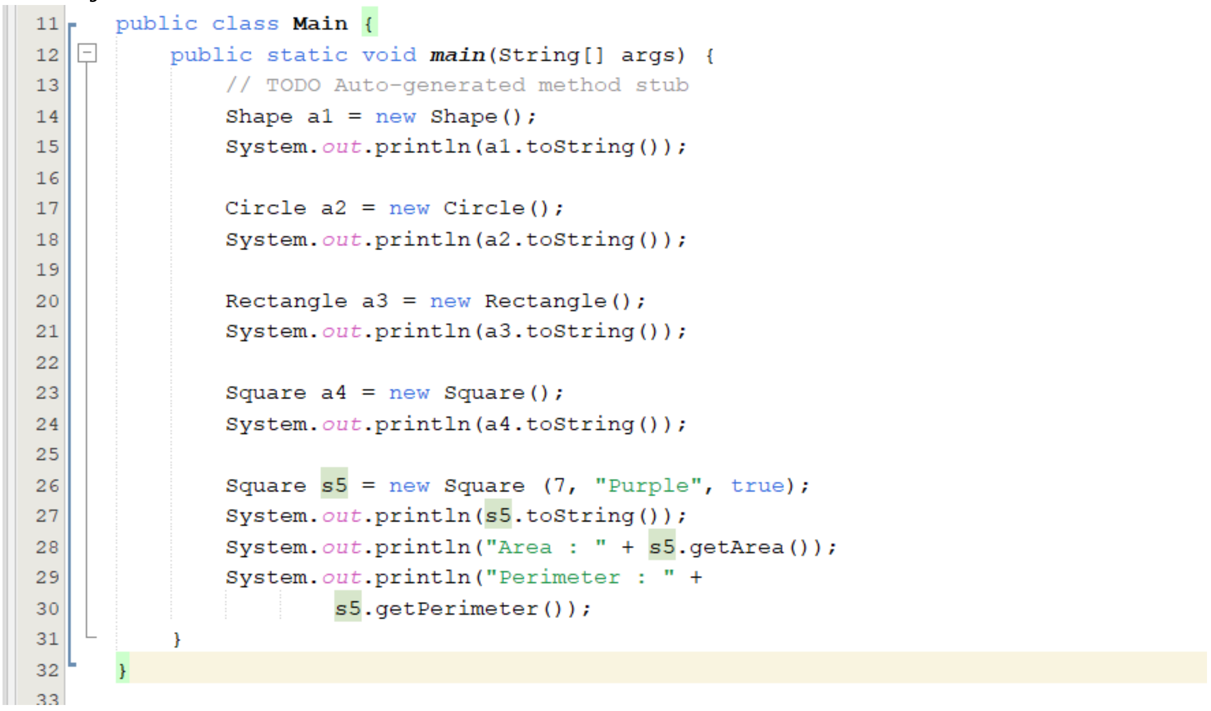


```java
public class Square extends Rectangle {
    public Square(){
        super();
    }

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

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

    public double getSide() {
        return super.getLength();
    }

    public void setSide(double side) {
        super.setWidth(side);
        super.setLength(side);
    }

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

```



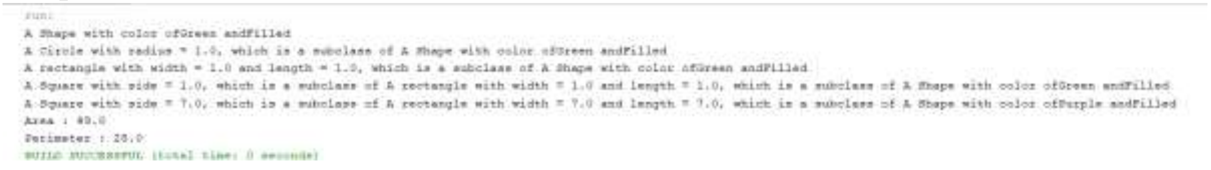
```java
    public String toString() {
        super.toString();
        return "A Square with side = " + this.getSide() +
                ", which is a subclass of " +
                super.toString();
    }

}

```

5. Main.java



```java
public class Main {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Shape a1 = new Shape();
        System.out.println(a1.toString());

        Circle a2 = new Circle();
        System.out.println(a2.toString());

        Rectangle a3 = new Rectangle();
        System.out.println(a3.toString());

        Square a4 = new Square();
        System.out.println(a4.toString());

        Square s5 = new Square (7, "Purple", true);
        System.out.println(s5.toString());
        System.out.println("Area : " + s5.getArea());
        System.out.println("Perimeter : " +
                s5.getPerimeter());
    }
}
```

Output



```
run:
A Shape with color ofGreen andFilled
A Circle with radius = 1.0, which is a subclass of A Shape with color ofGreen andFilled
A rectangle with width = 1.0 and length = 1.0, which is a subclass of A Shape with color ofGreen andFilled
A Square with side = 1.0, which is a subclass of A rectangle with width = 1.0 and length = 1.0, which is a subclass of A Shape with color ofGreen andFilled
A Square with side = 7.0, which is a subclass of A rectangle with width = 7.0 and length = 7.0, which is a subclass of A Shape with color ofPurple andFilled
Area : 49.0
Perimeter : 28.0
BUILD SUCCESSFUL (total time: 0 seconds)
```