**Laporan Pertemuan 2 Teknik Pemrograman**

**Praktik**

****

**Disusun oleh :**

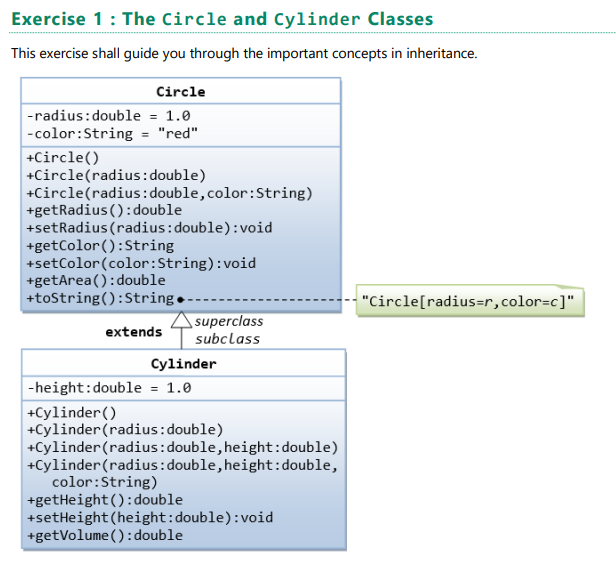
**Nesta Rizkia Saputra (231524060)**

**Kelas :**

**D4 – 1B Teknik Informatika**

**Tahun Ajaran 2023 – 2024**

# EXERCISE 1



Circle dan Cylinder merupakan sebuah class dengan properties yang ada dibawahnya. Dimana terdapat Attributes yaitu yang terdapat dibawah dari nama class. Lalu yang ada dibawahnya lagi adalah method – method yang terdapat pada class nya.

/\*\*

\* The Circle class models a circle with a radius and color.

\*/

public class Circle

{

    // Save as "Circle.java"

    // private instance variable, not accessible from outside this class

    private double radius;

    private String color;

    // Constructor 3rd task 1.1

    public Circle(double r,String c)

    {

        this.radius = r;

        this.color = c;

    }

    // Constructors (overloaded)

    /\*\* Constructs a Circle instance with default value for radius and color \*/

    public Circle()

    { // 1st (default) constructor

        radius = 1.0;

        color = "red";

    }

    /\*\* Constructs a Circle instance with the given radius and default color \*/

    public Circle(double r)

    { // 2nd constructor

        radius = r;

        color = "red";

    }

    /\*\* Returns the radius \*/

    public double getRadius()

    {

        return radius;

    }

    /\*\* Returns the area of this Circle instance \*/

    public double getArea()

    {

        return radius\*radius\*Math.PI;

    }

    /\*\* Return a self-descriptive string of this instance in the form of

    Circle[radius=?,color=?] \*/

    public String toString()

    {

        return "Circle[radius=" + radius + " color=" + color + "]";

    }

    // task 1.1

    public String getColor()

    {

        return color;

    }

    // task 1.1

    public void setColor(String color)

    {

        this.color = color;

    }

}

public class Cylinder extends Circle

{  //Save as "Cylinder.java"

    private double height; // private variable

    // 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(); // call superclass no-arg constructor Circle()

        this.height = height;

    }

    // Constructor with default color, but given radius, height

    public Cylinder(double radius, double height)

    {

        super(radius); // call superclass constructor Circle(r)

        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 super.getArea()\*height;

    }

    // task 1.2

    @Override

    public double getArea()

    {

        return 2\*Math.PI\*getRadius()\*height+2\*getVolume();

    }

    // task 1.3

    @Override

    public String toString()

    { // in Cylinder class

        return "Cylinder: subclass of " + super.toString() + " height=" + height; // use Circle's toString()

    }

}

 public class TestCylinder

    { // save as "TestCylinder.java"

    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()

    + " " + c1.toString()); // task 1.3

    // 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()

    + " " + c2.toString()); //task 1.3

    // 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()

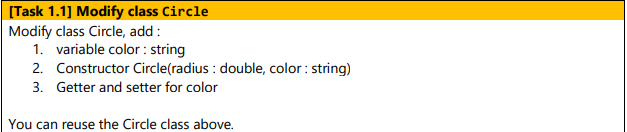
    + " volume=" + c3.getVolume()

    + " " + c3.toString()); // task 1.3

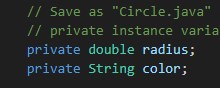
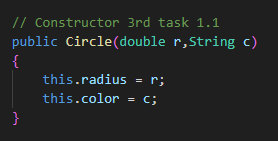
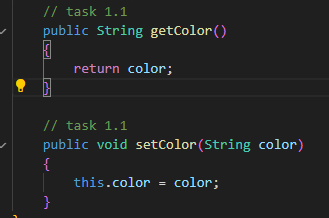
    }

}

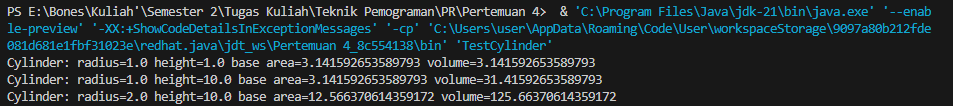
## Task 1.1



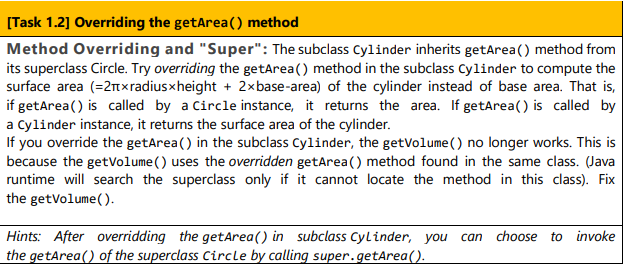
### Penyelesaian

1. 
2. 
3. 

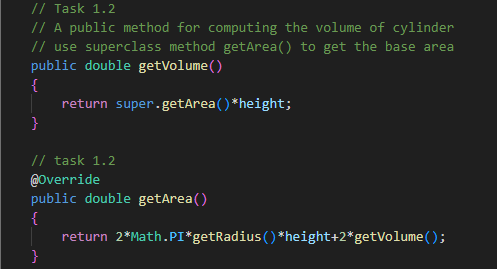
### Output



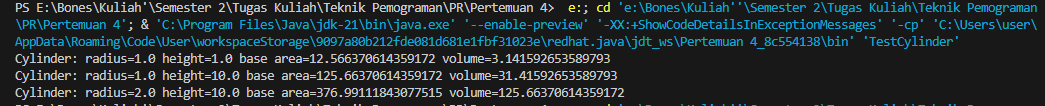
## Task 1.2



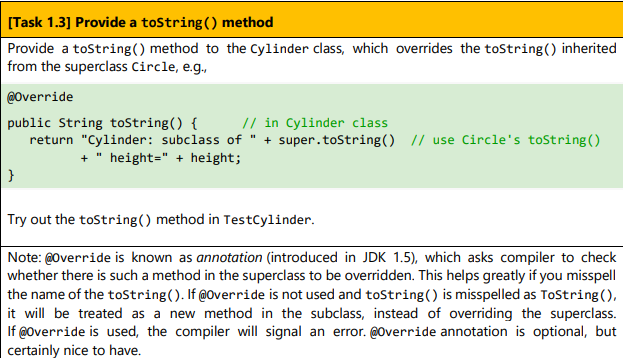
### Penyelesaian



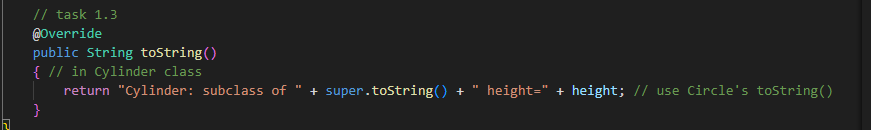
### Output



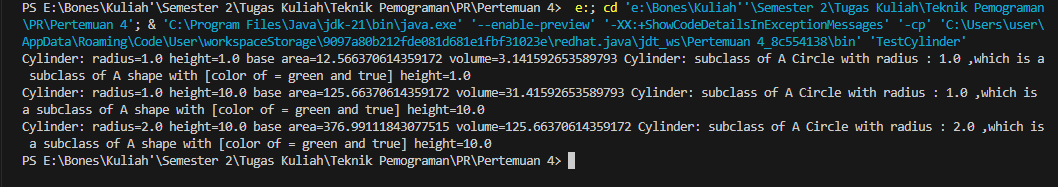
## Task 1.3



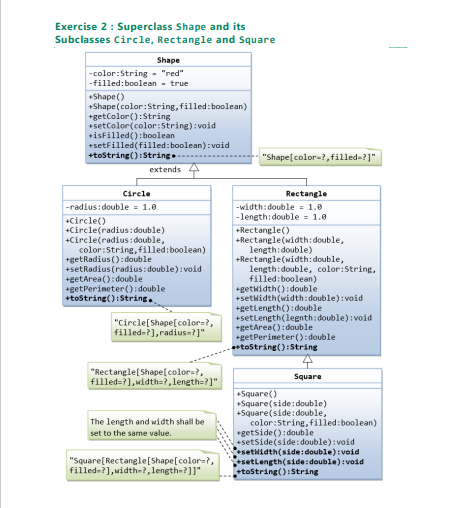
### Penyelesaian



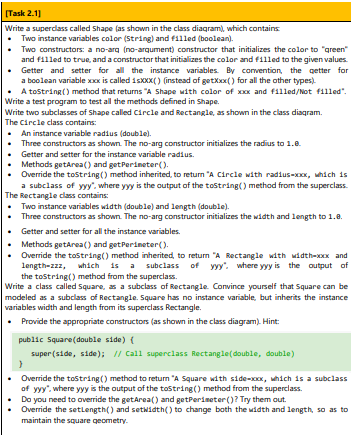
### Output



# Exercise 2



Gambar diatas merupakan request untuk class juga attributes dari masing – masing class dan method – method nya.



## Task 2.1

### SuperClass Shape

public class Shape

{

    // private instance variable

    private String color;

    private Boolean filled;

    // 1st Constructs a shape instance with default value for color and filled

    public Shape()

    {

        color = "green";

        filled = true;

    }

    // 2nd Construct a shape instance with given values for color and filled

    public Shape(String color,boolean filled)

    {

        this.color = color;

        this.filled = filled;

    }

    public String getColor()

    {

        return color;

    }

    public void setColor()

    {

        this.color = color;

    }

    public boolean isFilled()

    {

        return filled;

    }

    public void setFilled()

    {

        this.filled = filled;

    }

    public String toString()

    {

        return ("A shape with [color of = " + color + " and " + filled + "]");

    }

}

### Class Circle (subclass superclass Shape)

public class Circle extends Shape

{

    private double radius;

    public Circle ()

    {

        radius = 1.0;

    }

    public Circle (double radius)

    {

        this.radius = radius;

    }

    public Circle (double radius,String color,boolean filled)

    {

        super(color,filled);

        this.radius = radius;

    }

    public double getRadius()

    {

        return radius;

    }

    public void setRadius()

    {

        this.radius = radius;

    }

    public double getArea()

    {

        return radius\*radius\*Math.PI;

    }

    public double getPerimeter()

    {

        return 2\*Math.PI\*getRadius();

    }

    @Override

    public String toString()

    {

        return "A Circle with radius : " + getRadius() + " ,which is a subclass of " + super.toString();

    }

}

### Class Rectangle (Subclass class Shape dan SuperClass)

public class Rectangle

{

    private double width;

    private double length;

    public Rectangle()

    {

        width = 1.0;

        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();

        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 getLength()\*getWidth();

    }

    public double getPerimeter()

    {

        return 2\*getLength()+getWidth();

    }

    @Override

    public String toString() {

        return "a Rectangle with width : " + getWidth() + "and length : " + getLength() + " ,which is a subclass of " + super.toString();

    }

}

### Class Square (Subclass class Rectangle)

class Square extends Rectangle

{

    public Square()

    {

     super();

    }

    public Square(double side)

    {

        super(side,side); // call superclass Rectangle (double,double)

    }

    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.getLength();

    }

    @Override

    public void setLength(double side)

    {

        getSide();

    }

    @Override

    public void setWidth(double side)

    {

        getSide();

    }

    @Override

    public String toString()

    {

        return "A Square with side : " + getSide() + " ,which is a subclass of " + super.toString();

    }

    @Override

    public double getArea()

    {

        return getSide()\*getSide();

    }

    @Override

    public double getPerimeter()

    {

        return 4\*getSide();

    }

}

### Class TestShape (Main)

public class TestShape

{

    public static void main(String[] args)

    {

        //Declare and allocate a new instance of shape with default color and filled

        Shape s1 = new Shape();     // it's identifier for the first output

        System.out.println(s1.toString());

        // Declare and allocate a new instance of shape with given values color and filled

        Shape s2 = new Shape("white",true); // it's identifier for the second output

        System.out.println(s2.toString() + "\n" );

        //Declare and allocate a new instance of Circle with default

        Circle c1 = new Circle();     // it's identifier for the first output

        System.out.println(c1.toString());

        // Declare and allocate a new instance of Circle with given values

        Circle c2 = new Circle(10.0); // it's identifier for the second output

        System.out.println(c2.toString());

        // Declare and allocate a new instance of Circle with given values color and filled

        Circle c3 = new Circle(100.0,"White", true); // it's identifier for the Third output

        System.out.println(c3.toString() + "\n" );

        //Declare and allocate a new instance of Rectangle with default

        Rectangle R1 = new Rectangle();     // it's identifier for the first output

        System.out.println(R1.toString());

        // Declare and allocate a new instance of Rectangle with given values

        Rectangle R2 = new Rectangle(10.0,15.0); // it's identifier for the second output

        System.out.println(R2.toString());

        // Declare and allocate a new instance of Rectangle with given values color and filled

        Rectangle R3 = new Rectangle(25.0,15.0,"black", true); // it's identifier for the third output

        System.out.println(R3.toString() + "\n" );

        //Declare and allocate a new instance of Square with default

        Square Sq1 = new Square();     // it's identifier for the first output

        System.out.println(Sq1.toString());

        // Declare and allocate a new instance of Square with given values

        Square Sq2 = new Square(25.0); // it's identifier for the second output

        System.out.println(Sq2.toString());

        // Declare and allocate a new instance of Square with given values color and filled

        Square Sq3 = new Square(30.0,"White", true); // it's identifier for the third output

        System.out.println(Sq3.toString() + "\n" );

    }

}

### OUTPUT

