

TUGAS PRATIKUM 7

(Sebagai pemenuhan salah satu tugas mata kuliah

Pemograman Berorientasi Objek(Pratikum) Program Studi D3 Teknik Informatika)



Disusun oleh :

Rani Indrianna Sembiring (201511025)

Kelas : 2A

POLITEKNIK NEGERI BANDUNG

JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA

PRODI D3 TEKNIK INFORMATIKA

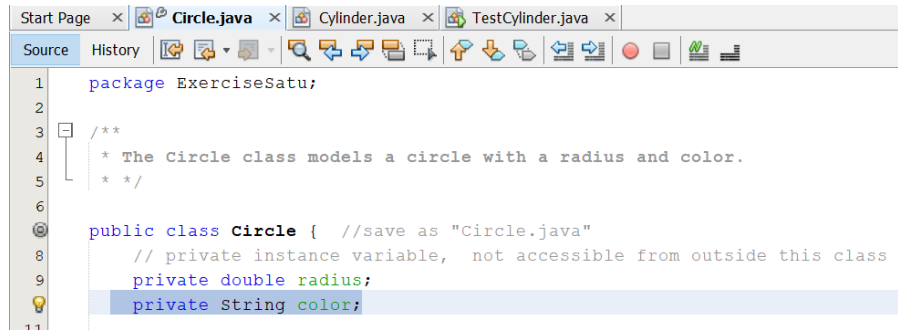
2021 – 2022

Exercise 1 : The Circle and Cylinder Classes

- [Task 1.1] Modify class Circle

1. Add variable color : string

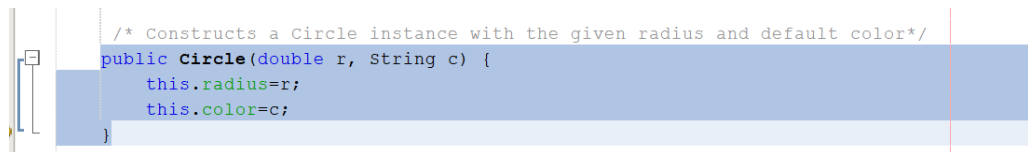
Menambahkan atribut color bertipe String (private)



```
1 package ExerciseSatu;
2
3 /**
4  * The Circle class models a circle with a radius and color.
5  */
6
7 public class Circle { //save as "Circle.java"
8     // private instance variable, not accessible from outside this class
9     private double radius;
10    private String color;
11 }
```

2. Add Constructor Circle (radius : double, color : string)

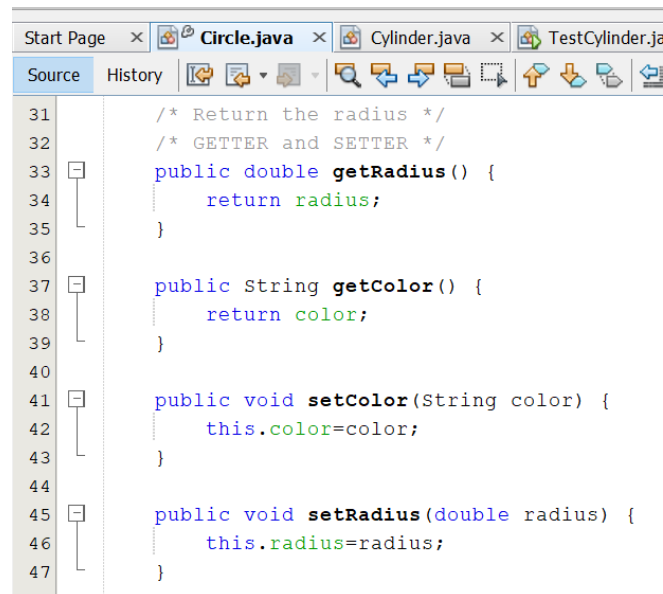
Menambahkan Constructor Circle dengan r bertipe double dan c bertipe String



```
/* Constructs a Circle instance with the given radius and default color*/
public Circle(double r, String c) {
    this.radius=r;
    this.color=c;
}
```

3. Add Getter dan Setter for color

Menambahkan getter dan setter pada color dan juga pada radius



```
31 /* Return the radius */
32 /* GETTER and SETTER */
33 public double getRadius() {
34     return radius;
35 }
36
37 public String getColor() {
38     return color;
39 }
40
41 public void setColor(String color) {
42     this.color=color;
43 }
44
45 public void setRadius(double radius) {
46     this.radius=radius;
47 }
```

- [Task 1.2] Overriding the `getArea()` method

- Menambahkan override dari metode `getArea()` di subclass `Cylinder` untuk menghitung luas permukaan `Cylinder`

```
@Override
public double getArea() {
    return 2*Math.PI*getRadius()*getHeight()+2*super.getArea();
}
```

- Memperbaiki syntax dari method `getVolume()` karena `getArea()` di subclass `Sylinder` telah di buat menjadi override Maka `getVolume()` sudah tidak berfungsi lagi Maka perlu memperbaiki syntax dengan cara memanggil `super.getArea() * height`.

```
//use superclass method getArea() to get the base area
public double getVolume() {
    return super.getArea()*height; //Use circle's get area
}
```

- [Task 1.3] Provide a `toString()` method

- Memberikan method `toString()` pada class `Cylinder` dan memberikan override pada method tersebut karena superclass nya sendiri sudah ada method `toString()`

```
@Override
public String toString() { //in Cylinder class
    return "Cylinder: subclass of " + super.toString() //use circle's toString()
        + " height=" + height;
}
```

- Kemudian melakukan testing pada method `toString()` yang ada pada class `Cylinder.java` dengan memanggil method tersebut pada class `TestCylinder.java` kemudian menghasilkan compilenya.

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

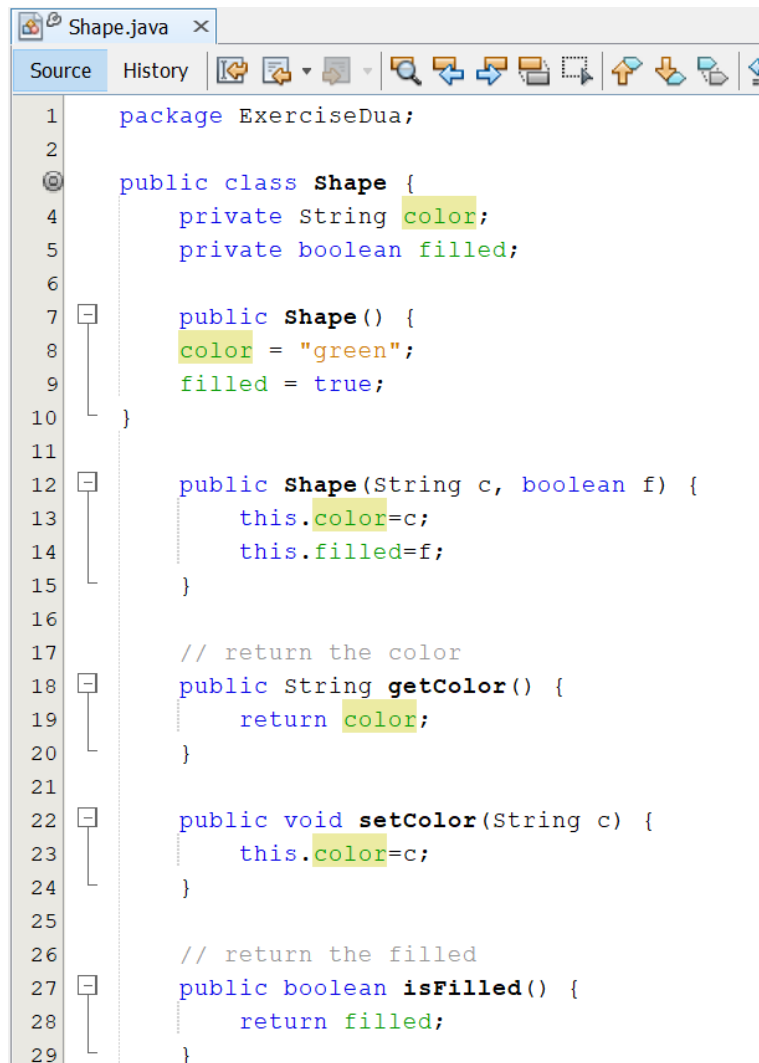
        //Declare and allocate a new instance of cylinder
        // specifying height, with default color and radius
        Cylinder c2 = new Cylinder(10.0);
        System.out.println("Cylinder:")
            + " radius=" + c2.getRadius()
            + " height=" + c2.getHeight()
            + " base area="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

```
Output - W7_P8O_PR (run-single) x
compile-single:
run-single:
Cylinder: subclass of Circle[radius=1.0 color=red] height=1.0
Cylinder: radius=1.0 height=10.0 base area=69.11503837897544 volume=31.41592653589793
Cylinder: radius=2.0 height=10.0 base area=150.79644737231007 volume=125.66370614359172
BUILD SUCCESSFUL (total time: 1 second)
```

Exercise 2 : Superclass Shape dan it's Subclass Circle, Rectangle and Square

- [Task 2.1]

- Membuat superclass untuk memanggil **Shape** yang berisi tentang :
 1. Dua variabel color bertipe String and Filled bertipe Boolean
 2. Dua constructor : a no-arg (no-argument) yang menginisialisasi color = "green" dan diisi ke true, dan constructor yang menginisialisasi warna dan diisi ke nilai yang di berikan
 3. Getter and setter untuk semua variabel instan. Dari konvensi, getter and setter untuk variabel Boolean xxx di panggil dengan isXXX() (bukan menjadi getXxx()) untuk semua type)
 4. Method toString() yang mengembalikan " A Shape with color of xxx and Filled/Not Filled"



```
1 package ExerciseDua;
2
3 public class Shape {
4     private String color;
5     private boolean filled;
6
7     public Shape() {
8         color = "green";
9         filled = true;
10    }
11
12    public Shape(String c, boolean f) {
13        this.color=c;
14        this.filled=f;
15    }
16
17    // return the color
18    public String getColor() {
19        return color;
20    }
21
22    public void setColor(String c) {
23        this.color=c;
24    }
25
26    // return the filled
27    public boolean isFilled() {
28        return filled;
29    }
```

```

30
31     public void setFilled(boolean f) {
32         this.filled=f;
33     }
34
35     public String toString() {
36         if(isFilled()) {
37             return "A shape with color of " + color + " is Filled";
38         }else{
39             return "A shape with color of " + color + " isn't Filled";
40         }
41     }
42 }

```

- Membuat test program untuk mengetest method yang di definisikan pada Shape

The screenshot shows an IDE with two tabs: Shape.java and Main.java. The Main.java tab is active, showing a test program. The output window at the bottom shows the results of running the program.

```

Main.java
package ExerciseDua;

public class Main {
    public static void main(String args[]) {
        Shape s1 = new Shape();
        System.out.println(s1.toString());
        System.out.println("*****");
        Shape s2 = new Shape ("green", true);
        System.out.println(s2.toString());
        System.out.println("*****");
    }
}

Output - W7_PBO_PR (run-single)
Compiling 1 source file to C:\Users\Asus\Documents\TUGAS PBO\W7_PBO_PR\build\classes
compile-single:
run-single:
A shape with color of red is Filled
*****
A shape with color of green is Filled
*****
BUILD SUCCESSFUL (total time: 1 second)

```

- Membuat subclass ke dua dari Shape yang memanggil Circle dan Rectangle
 - Kelas Circle yang berisi tentang :**
 1. Sebuah variabel radius bertipe double
 2. Tiga konstruktor seperti yang di tunjukkan. Konstruktor no-arg menginisialisasi radius ke 1.0
 3. Getter dan setter untuk variabel radius
 4. Method getArea() dan getPerimeter()
 5. Mengganti method toString() yang di warisi untuk mengembalikan “Circle with radius=xxx, which is a subclass of yyy” , dimana yyy merupakan output dari metode toString() dari superclass.

```
Shape.java x Circle.java x
Source History
1 package ExerciseDua;
2
3 public class Circle extends Shape {
4     private double radius;
5
6     public Circle() {
7         super();
8         radius = 1.0;
9     }
10
11     public Circle(double r) {
12         super();
13         this.radius=r;
14     }
15
16     public Circle(double r, String c, boolean f) {
17         this.radius=r;
18         super.setColor(c);
19         super.setFilled(f);
20     }
21
22     //Getter and Setter
23     public double getRadius() {
24         return radius;
25     }
26
27     public void setRadius(double r) {
28         this.radius=r;
29     }
30
31     public double getArea() {
32         return Math.PI*radius*radius;
33     }
34
35     public double getPerimeter() {
36         return 2*Math.PI*radius;
37     }
38
39     public String toString() {
40         return "A circle with radius = " + getRadius() + ", which is subclass of\n"
41             + super.toString();
42     }
43 }
```

- **Class Rectangle berisi :**

1. Variabel width dengan tipe data double dan variabel length tipe data double
2. Tiga constructor seperti yang di tunjukkan. Constructor no-arg menginisialisasi width dan length menjadi 1.0
3. Getter dan setter untuk semua variabel instan
4. Metode getArea() dan getPerimeter()
5. Mengganti method toString() yang di warisi, untuk mengembalikan “ A Rectangle with width=xxx and length=zzz, which is a subclass of yyy”, dimana yyy = output dari method toString() dari superclass.

```
Shape.java x Circle.java x Rectangle.java x Square.java x
Source History
1 package ExerciseDua;
2
3
4 public class Rectangle extends Shape {
5     private double width;
6     private double length;
7
8     public Rectangle() {
9         super();
10        this.length=1.0;
11        this.width=1.0;
12    }
13
14    public Rectangle(double width, double length) {
15        super();
16        this.width=width;
17        this.length=length;
18    }
19
20    public Rectangle(double width, double length, String c, boolean f) {
21        this.length=length;
22        this.width=width;
23        super.setColor(c);
24        super.setFilled(f);
25    }
26
27    //getter and setter
28    public double getWidth() {
29        return width;
30    }
31
32    public void setWidth(double w) {
33        this.width=w;
34    }
35
36    public double getLength() {
37        return length;
38    }
39
40    public void setLength(double l) {
41        this.length=l;
42    }
43
44    public double getArea() {
45        return width*length;
46    }
47
48    public double getPerimeter() {
49        return 2*(width+length);
50    }
51
52    public String toString() {
53        return "A Rectangle with width = " + width + " and length = " + length
54        + ", which is a subclass of \n " + super.toString();
55    }
56 }
```

- **Membuat class Square sebagai subclass dari Rectangle.** Square tidak memiliki variabel instan, tetapi mewarisi width dan length dari superclass Rectangle

1. Berikan constructor yang sesuai:

```
public Square(double side) {
    super(side, side); // Call superclass Rectangle(double, double)
}
```

2. Mengganti method toString() untuk mengembalikan “ A Square with side=xxx, which is a subclass of yyy” dimana yyy = output dari metode toString() dari superclass
3. Mengganti setLength() dan setWidth() untuk mengubah width dan length untuk mempertahankan geometri persegi

```

Shape.java x Circle.java x Rectangle.java x Main.java x Square.java x
Source History
1 package ExerciseDua;
2
3 public class Square extends Rectangle {
4
5     public Square() {
6         super();
7     }
8
9     public Square(double side) {
10        super(side,side); //call superclass rectangle (double, double)
11    }
12
13    public Square(double side, String c, boolean f) {
14        super(side,side, c,f);
15    }
16
17    public double getSide() {
18        return super.getLength();
19    }
20
21    public void setside(double side) {
22        setWidth(side);
23        setLength(side);
24    }
25
26    @Override
27    public String toString() {
28        return "A Square width side= " + getSide() +
29            " which is a subclass of \n" + super.toString();
30    }
31 }

```

• TEST PROGRAM

```

Main.java x
Source History
1 package ExerciseDua;
2
3 public class Main {
4
5     public static void main(String args[]) {
6
7         System.out.println("---SHAPE---");
8         Shape s1 = new Shape();
9         Shape s2 = new Shape("blue", false);
10        System.out.println(s1.toString());
11        System.out.println(s2.toString());
12
13        System.out.println("---CIRCLE---");
14        Circle c1 = new Circle();
15        Circle c2 = new Circle(10.0, "white", false);
16        System.out.println(c1.toString());
17        System.out.println("Luas = " + c1.getArea());
18        System.out.println(c2.toString());
19
20        System.out.println("---RECTANGLE---");
21        Rectangle r1 = new Rectangle();
22        Rectangle r2 = new Rectangle(12.0, 6.0, "black", true);
23        System.out.println(r1.toString());
24        System.out.println(r2.toString());
25        System.out.println("Luas = " + r2.getArea() + " Keliling = " + r2.getPerimeter());
26
27        System.out.println("---SQUARE---");
28        Square sq1 = new Square();
29        Square sq2 = new Square(4.5, "yellow", false);
30        System.out.println(sq1.toString());
31        System.out.println(sq2.toString());
32        System.out.println("Luas = " + sq1.getArea() + " Keliling = " + sq1.getPerimeter());
33    }
34 }
35

```

```

Output - W7_PBO_PR (run-single) x
compile-single:
run-single:
---SHAPE---
A shape with color of green is Filled
A shape with color of blue isn't Filled
---CIRCLE---
A circle [A shape with color of green is Filled], radius= 1.0]
Luas = 3.141592653589793
A circle [A shape with color of white isn't Filled], radius= 10.0]
---RECTANGLE---
Square[Rectangle[A shape with color of green is Filled, width= 1.0, length= 1.0]]
Square[Rectangle[A shape with color of black is Filled, width= 12.0, length= 6.0]]
Luas = 72.0 Keliling = 36.0
---SQUARE---
Square[Square[Rectangle[A shape with color of green is Filled, width= 1.0, length= 1.0]]
Square[Square[Rectangle[A shape with color of yellow isn't Filled, width= 4.5, length= 4.5]]
Luas = 1.0 Keliling = 4.0
BUILD SUCCESSFUL (total time: 0 seconds)

```