LAPORAN PRAKTIKUM 6

Pemrograman Berbasis Objek

Disusun Oleh :

Muhammad Wildan Gumilang (231511087)

Jurusan Teknik Komputer dan Informatika

Politeknik Negeri Bandung

DAFTAR ISI

Exercise 1

[Task 1.1] Modify class Circle

Modify class Circle, add :

1. variable color : string

```
public class Circle {
    // Save as "Circle.java"
    // private instance variable, not
    private double radius;
    private String color;
```

Membuat variable color menjadi tipe data String

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

```
public Circle(double radius, String color) {
    this.radius = radius;
    this.color = color;
}
```

Membuat constructor dari class Circle

3. Getter and setter for color

```
public String getColor(){
    return color;
}

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

Membuat getter dan setter untuk color pada class Circle

You can reuse the Circle class above.

Output :

```
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
```

Komentar :

Pada program ini saya memodifikasi class circle, yaitu mengubah variable colo menjadi String, membuat constructor, dan juga membuat setter dan juga getter.

[Task 1.2] Overriding the getArea() method

```
public double getVolume() {
    return super.getArea()*height;
}

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

Output :

```
Cylinder: radius=1.0 height=1.0 base area=12.566370614359172 volume=12.566370614359172
Cylinder: radius=1.0 height=10.0 base area=69.11503837897544 volume=691.1503837897544
Cylinder: radius=2.0 height=10.0 base area=150.79644737231007 volume=1507.9644737231006
```

Komentar :

Pada program ini saya membuat fungsi getArea pada class cylinder untuk menghitung luas alas dari sebuah cylinder. Dilakukan overriding getArea() yang ada di cirlce dengan getArea() yang ada di cylinder. Dan untuk menghitung volume dari cylinder, dilakukan juga overriding pada getArea dari class circle.

[Task 1.3] Provide a toString() method

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

```
    + " volume=" + c1.getVolume()
    + " to string " + c1.toString());
```

Output :

```
Cylinder: radius=1.0 height=1.0 base area=12.566370614359172 volume=3.141592653589793 to string Cylinder: subclass of Circl
e[radius=1.0 color=red] height=1.0
Cylinder: radius=1.0 height=10.0 base area=69.11503837897544 volume=31.41592653589793 to string Cylinder: subclass of Circl
e[radius=1.0 color=red] height=1.0
Cylinder: radius=2.0 height=10.0 base area=150.79644737231007 volume=125.66370614359172 to string Cylinder: subclass of Cir
cle[radius=1.0 color=red] height=1.0
```

Komentar :

Pada program ini, saya membuat method toString dan juga memodifikasinya dengan menambahkan super (overriding)

[Task 2.1]

Class shape :

```java
package com.shape;

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

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

    public String getColor() {
        return color;
    }

    public boolean isFilled() {
        return filled;
    }

    public String toString() { // in Cylinder class
        return "A Shape with color of " + color + " and " + filled;
    }
```

Class Circle :

```java
package com.shape;

public class Circle extends Shape {

    private double radius;

    public Circle() {
        super();
        radius = 1.0;
    }

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

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

Class Rectangle :

```java
public class Rectangle extends Shape{
    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(String color, boolean filled, double width, double length) {
        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);
    }

    public String toString() {
        return "A Rectangle with width " + width + " and length " + length + " which is a subclass of " +super.toString();
    }
```

Class Square :

```java
package com.shape;

public class Square extends Rectangle {

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


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

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

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

Output :

```
com.shape.TestShape'
Shape1: A Shape with color of green and true
circle1: A Circle with radius 1.0, which is a subclass of A Shape with color of green and true
rectangle1: A Rectangle with width 1.0 and length 1.0 which is a subclass of A Shape with color of green and true
square1: A Square with side=3.0, which is a subclass of A Rectangle with width 3.0 and length 3.0 which is a subclass of A
Shape with color of green and true
Shape2: A Shape with color of blue and false
Shape2 (updated): A Shape with color of red and true
Shape2's color: red
Shape2 is filled: true
```

Komentar :

Pada program ini, fungsi getArea() pada kelas Cylinder diimplementasikan untuk menghitung luas alas dari sebuah silinder. Metode ini melakukan overriding terhadap getArea() yang ada di kelas Circle. Dengan demikian, ketika getArea() dipanggil dari objek Cylinder, metode ini akan menghitung luas permukaan silinder, bukan hanya luas alas. Selain itu, untuk menghitung volume silinder, metode getVolume() dalam kelas Cylinder juga diubah agar memanfaatkan metode getArea() dari kelas Circle melalui penggunaan super.getArea(). Hal ini memastikan

bahwa volume silinder dihitung dengan benar, menggunakan luas alas yang didefinisikan dalam kelas induk Circle.

[Task 3.1] Extending the Sortable abstract class

```
abstract class Sortable {
    public abstract int compare(Sortable b);

    public static void shell_sort(Sortable[] a) {
        int n = a.length;
        for (int gap = n / 2; gap > 0; gap /= 2) {
            for (int i = gap; i < n; i++) {
                Sortable temp = a[i];
                int j;
                for (j = i; j >= gap && temp.compare(a[j - gap]) < 0; j -= gap) {
                    a[j] = a[j - gap];
                }
                a[j] = temp;
            }
        }
    }
}
```

```
public class EmployeeTest {
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {

        Employee[] staff = new Employee[3];
        staff[0] = new Employee("Antonio Rossi", 2000000, 1, 10, 1989);
        staff[1] = new Manager("Maria Bianchi", 2500000, 1, 12, 1991);
        staff[2] = new Employee("Isabel Vidal", 3000000, 1, 11, 1993);

        for (Employee emp : staff) {
            emp.raiseSalary(5);
        }

        Sortable.shell_sort(staff);

        for (Employee emp : staff) {
            emp.print();
        }
    }
}
```

```
public class ManagerTest{
    Run | Debug | Run main | Debug main
    public static void main (String[] args){
    Employee[] staff = new Employee[3];
    staff[0] = new Employee("Antonio Rossi", 2000000, 1, 10, 1989);
    staff[1] = new Manager("Maria Bianchi", 2500000, 1, 12, 1991);
    staff[2] = new Employee("Isabel Vidal", 3000000, 1, 11, 1993);
    int i;
    for (i = 0; i < 3; i++) staff[i].raiseSalary(5);
    for (i = 0; i < 3; i++) staff[i].print();
    }
}
```

Output :

```
est.java } ; if ($?) { java Emplo
Antonio Rossi 2100000.0 1989
Maria Bianchi 3037500.0 1991
Isabel Vidal 3150000.0 1993
PS D:\wg\Kuliah\semester 3\PBO\pr
```

Case 1

Dalam menyelesaikan case ini, kelas Employee mewarisi kelas abstrak Sortable dan mengimplementasikan metode compare, yang membandingkan gaji antar objek Employee. Dengan ini, array objek Employee dapat diurutkan berdasarkan gaji menggunakan metode shell_sort dari Sortable.

Case 2

Untuk Case 2, kelas Manager mewarisi Employee, jadi secara otomatis Manager juga dapat diurutkan berdasarkan gaji, karena metode compare sudah diimplementasikan di Employee. Hal ini memungkinkan objek Manager dan Employee diurutkan bersama-sama tanpa memerlukan perubahan tambahan di kelas Manager.

Link GitHub :

https://github.com/WildanGumilang/PBO-praktek