LAPORAN TUGAS PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK

Laporan ini disusun untuk memenuhi Tugas Mata Kuliah Pemrograman Berorientasi Objek

Disusun Oleh :

Benny Yoga Suhardi 211511035

PROGRAM STUDI D3 TEKNIK INFORMATIKA

JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA

POLITEKNIK NEGERI BANDUNG

TAHUN 2022

Task 1

Class Circle

```java
public class Circle {
    private double radius;
    private String color;
     // Constructors (overloaded)
     /** Constructs a Circle instance with default value for radius and color */
    // 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";
     }

     public Circle(double r, String color) { // 2nd constructor
         radius = r;
         color = "red";
     }

```

```java
     /** 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 + "]";
     }

    public String getColor() {
        return color;
    }

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

```

Class Cylinder extend Circle

```
public class Cylinder extends Circle{
    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, "Black"); // 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 getArea()*height;
     }


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

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

Main Class

```
public class TestCylinder {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        // Declare and allocate a new instance of cylinder
         // with default color, radius, and height
         Cylinder c1 = new Cylinder();
         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());

         // 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());
    }

}
```

Result

```
Cylinder: subclass of Circle[radius=1.0 color=red] height=1.0
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
```

Task 1.1 (Modify Circle Class)

```
    public Circle(double r, String color) { // 2nd constructor
        radius = r;
        color = "red";
    }

    public String getColor() {
        return color;
    }

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

Task 1.2 (Overriding GetArea)

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

Task 1.3 Provide toString() method

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

Task 2

Class Shape

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

    public Shape() {
        color = "red";
        filled = true;
    }

    public Shape(String Color, boolean Filled) {
        color = Color;
        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() {
        return "Shape[Color = " + this.color + ", filled = " + this.filled +"]";
    }
```

Class Circle extends Shape

```java
public class Circle extends Shape{
    private double radius;

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

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

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

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

Class Rectangle Extend Shape

```java
package Task2;

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

    public Rectangle() {
        super();
        this.width = 1.0;
        this.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;
    }
```

```java
    public double getLength() {
        return length;
    }
    public void setLength(double length) {
        this.length = length;
    }

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

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

    public String toString() {
        return "Rectangle[Shape[Color = "+ this.getColor() +", Filled "+ this.isFilled()
        +", Width "+ this.width +", Length "+ this.length +"]";
    }

}
```

Class Square Extend Rectangle

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

    @Override
    public void setLength(double Side) {
        getSide(Side);
    }

    public void setWidth(double Side) {
        getSide(Side);
    }

    public String toString() {
        return "Square[Shape[Color = "+ this.getColor() +", Filled "+ this.isFilled()
        +", Width "+ super.getWidth() +", Length "+ this.getLength()+"]";
    }
}
```

Main Class

```java
package Task2;

public class main {

    public static void main(String[] args) {
        Square kotak = new Square(15);
        System.out.println(kotak.toString());

        Circle lingkaran = new Circle(14);
        lingkaran.setColor("Blue");
        System.out.println(lingkaran.toString());
    }

}
```

Result

```
Square[Shape[Color = red, Filled true, Width 15.0, Length 15.0]
Shape[Color = Blue, filled = true]
```

Task 3

Class Employee

```java
package Task3;

public class Employee {
    public Employee(String n, double s, int day, int month, int year){
         name = n;
         salary = s;
         hireday = day;
         hiremonth = month;
         hireyear = year;
        }
        public void print(){
         System.out.println(name + " " + salary + " " + hireYear());
        }
        public void raiseSalary(double byPercent){
         salary *= 1 + byPercent / 100;
        }
        public int hireYear(){
        return hireyear;
        }
        private String name;
        private double salary;
        private int hireday;
        private int hiremonth;
        private int hireyear;

}
```

Class EmployeeTest

```java
package Task3;

public class EmployeeTest {
    public static void main (String[] args){
         Employee[] staff = new Employee[3];
         staff[0] = new Employee("Antonio Rossi", 2000000, 1, 10, 1989);
         staff[1] = new Employee("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();
        }
}

```

Result

```
Antonio Rossi 2100000.0 1989
Maria Bianchi 2625000.0 1991
Isabel Vidal 3150000.0 1993
```

Class Manager

```java
package Task3;

import java.util.Calendar;
import java.util.GregorianCalendar;

public class Manager extends Employee{
        public Manager (String n, double s, int d, int m, int y){
        super(n, s, d, m, y);
        secretaryName = "";
        }
        public void raiseSalary(double byPercent){
        // add 1/2% bonus for every year of service
        GregorianCalendar todaysDate = new GregorianCalendar();
        int currentYear = todaysDate.get(Calendar.YEAR);
        double bonus = 0.5 * (currentYear - hireYear());
        super.raiseSalary(byPercent + bonus);
        }
        public String getSecretaryName(){
        return secretaryName;
        }
        private String secretaryName;
}
```

Class ManagerTest

```java
package Task3;

public class ManagerTest {
    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();
        }

}
```

Result

```
<terminated> ManagerTest [Java Application]
Antonio Rossi 2100000.0 1989
Maria Bianchi 3012500.0 1991
Isabel Vidal 3150000.0 1993
```

Case 1 (adding ShellSort)

Class ShellShort

```java
package Task3;

public abstract class Sortable {
    public abstract int compare(Sortable b);
    public static void shell_sort(Sortable[] a){
        int n = a.length;
        for (int interval = n / 2; interval > 0; interval /= 2) {
            for (int i = interval; i < n; i += 1) {
                Sortable temp = a[i];
                int j;

                for (j = i; j >= interval; j -= interval) {
                    a[j] = a[j - interval];
                }

                a[j] = temp;
                }
            }
        }

}
```

Main Program

```java
package Task3;

public class EmployeeTest {
    public static void main (String[] args){
         Employee[] staff = new Employee[3];
         staff[0] = new Employee("Antonio Rossi", 2000000, 1, 10, 1989);
         staff[1] = new Employee("Maria Bianchi", 2500000, 1, 12, 1991);
         staff[2] = new Employee("Isabel Vidal", 3000000, 1, 11, 1993);
         Sortable.shell_sort(staff);

         int i;
         for (i = 0; i < 3; i++) staff[i].raiseSalary(5);
         for (i = 0; i < 3; i++) staff[i].print();
        }

}
```

Result With ShellSort

```
<terminated> EmployeeTest [Java Application] C:\Program
Isabel Vidal 3150000.0 1993
Maria Bianchi 2625000.0 1991
Antonio Rossi 2100000.0 1989
```

Link Github : [BennyYoga/PBO_Praktikum (github.com)]