

**LAPORAN PRAKTIKUM  
PEMROGRAMAN II  
MODUL 5**



**POLIMORFISME**

**Oleh:**

**Ryan Muhammad Irfan      NIM. 2210817310013**

**PROGRAM STUDI TEKNOLOGI INFORMASI  
FAKULTAS TEKNIK  
UNIVERSITAS LAMBUNG MANGKURAT  
NOVEMBER 2023**

**LEMBAR PENGESAHAN**  
**LAPORAN PRAKTIKUM PEMROGRAMAN II**  
**MODUL 5**

Laporan Praktikum Pemrograman II Modul 5: Polimorfisme ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman II. Laporan Praktikum ini dikerjakan oleh:

Nama Praktikan : Ryan Muhammad Irfan  
NIM : 2210817310013

Menyetujui,  
Asisten Praktikum

Mengetahui,  
Dosen Penanggung Jawab Praktikum

Bachrul Uluum  
NIM. 2010817210025

Andreyan Rizky Baskara, S.Kom., M.Kom.  
NIP. 19930703 201903 1 011

## DAFTAR ISI

LEMBAR PENGESAHAN .....	2
DAFTAR ISI .....	3
DAFTAR GAMBAR.....	4
DAFTAR TABEL .....	5
SOAL 1 .....	6
A.    Source Code .....	10
B.    Output Program.....	14
C.    Pembahasan.....	14
D.    Tautan GIT.....	26

## **DAFTAR GAMBAR**

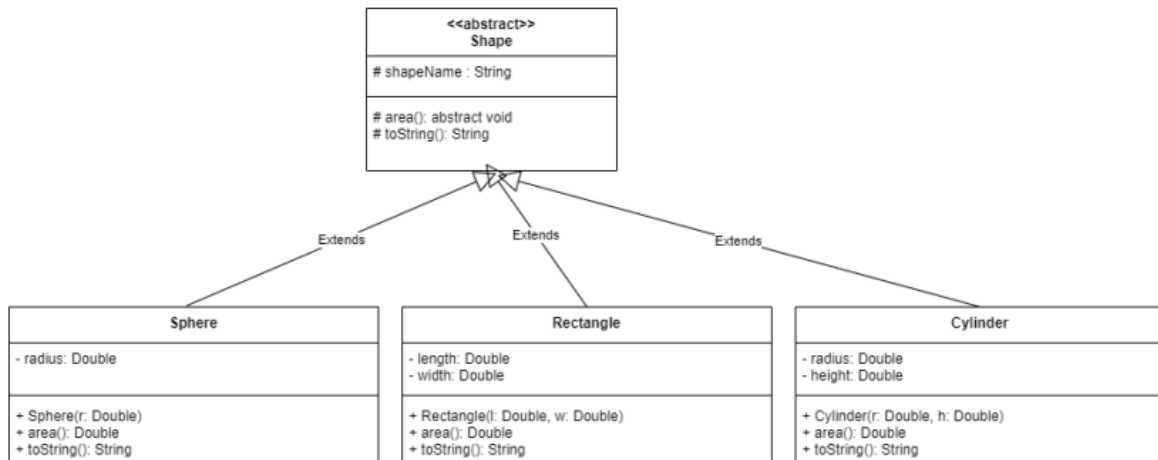
Gambar 1. Soal 1(diagram kelas) .....	6
Gambar 2. Soal 1(diagram kelas Paint.java) .....	7
Gambar 3. Screenshot Hasil Jawaban Soal 1 .....	14

## **DAFTAR TABEL**

Tabel 1. Soal 1 (Ilustrasi Code Sphere.Java).....	7
Tabel 2. Soal 1(Paint.java) .....	8
Tabel 3. Soal 1(Petunjuk PaintThings.java) .....	9
Tabel 4. Source Code Soal 1(Shape.java) .....	10
Tabel 5. Source Code Soal 1 (Sphere.java) .....	10
Tabel 6. Source Code Soal 1 (Rectangle.java) .....	11
Tabel 7. Source Code Soal 1 (Cylinder.java) .....	12
Tabel 8. Source Code Soal 1 (Paint.java).....	12
Tabel 9. Source Code Soal 1 (PaintThings.java).....	13

## SOAL 1

Pada praktikum kali ini anda akan diminta untuk membuat sebuah program yang dapat menghitung banyaknya liter cat yang digunakan untuk mewarnai bentuk ruang yang beragam. Buatlah sebuah hierarki kelas abstrak Shape dimana memiliki 3 kelas anak yaitu Sphere, Rectangle, dan Cylinder seperti ditunjukkan oleh diagram kelas berikut.



Gambar 1. Soal 1(diagram kelas)

Method `area()` digunakan untuk menghitung luas masing-masing objek. Berikut adalah formula yang digunakan untuk menghitung luas masing-masing bangun yang harus diimplementasikan.

Sphere:  $4 \times \pi \times radius^2$

Rectangle:  $length \times width$

Cylinder:  $\pi \times radius^2 \times height$

Method `toString()` digunakan untuk mengembalikan nilai String dari nama bangun

Berikut adalah ilustrasi dari kelas `Sphere.java`. Implementasikan kelas lainnya untuk `Shape`, `Rectangle` dan `Cylinder`

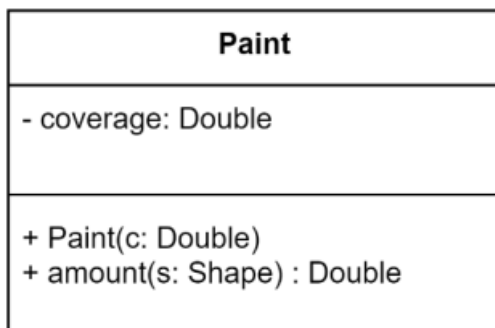
### Contoh Ilustrasi Sphere.java

```
public class Sphere extends Shape
{
    private double radius; //radius in feet
```

```
//-----
// Constructor: Sets up the sphere.
//-----
public Sphere(double r)
{
    super("Sphere");
    radius = r;
}
//-----
// Returns the surface area of the sphere.
//-----
public double area()
{
    return 4*Math.PI*(radius*radius);
}
//-----
// Returns the sphere as a String.
//-----
public String toString()
{
    return super.toString() + " of radius " + radius;
}
}
```

Tabel 1. Soal 1 (Ilustrasi Code Sphere.Java)

Selanjutnya, Buatlah kelas Paint.java seperti ditunjukkan diagram kelas berikut.



Gambar 2. Soal 1(diagram kelas Paint.java)

Method amount digunakan untuk menghitung banyaknya liter cat yang digunakan dengan persamaan berikut:

$$amount\ of\ paint = \frac{area\ of\ shape}{coverage}$$

Lengkapi kode dibawah supaya menghasilkan keluaran yang diinginkan

**Paint.java**

```
public class Paint
{
private double coverage; //number of square feet per gallon
//-----
// Constructor: Sets up the paint object.
//-----
public Paint(double c)
{
coverage = c;
}
//-----
// Returns the amount of paint (number of gallons)
// needed to paint the shape given as the parameter.
//-----
public double amount(Shape s)
{
System.out.println ("Computing amount for " + s);
return 0;
}
}
```

Tabel 2. Soal 1(Paint.java)

Terakhir, Buatlah kelas main bernama PaintThings.java. Tambahkan beberapa hal berikut agar program berjalan sesuai yang diinginkan.

1. Instansiasi 3 bentuk objek:
  - a. objek bernama deck berbentuk persegi panjang dengan ukuran Panjang 20cm dan lebar 30cm.
  - b. objek bernama bigBall berbentuk bola dengan ukuran radius 15cm.
  - c. objek bernama tank berbentuk silinder dengan ukuran radius 10cm dan tinggi 30cm.
2. Panggil fungsi yang tepat agar dapat menghitung jumlah cat yang diperlukan.

Petunjuk untuk kelas main PaintThings.java

```
import java.text.DecimalFormat;
public class PaintThings
```



```

{
//-----
// Creates some shapes and a Paint object
// and prints the amount of paint needed
// to paint each shape.
//-----
public static void main (String[] args)
{
final double COVERAGE = 350;
Paint paint = new Paint(COVERAGE);
Rectangle deck;
Sphere bigBall;
Cylinder tank;
double deckAmt, ballAmt, tankAmt;
// Instantiate the three shapes to paint
// Compute the amount of paint needed for each shape
// Print the amount of paint for each.
DecimalFormat fmt = new DecimalFormat("0.##");
System.out.println ("\nNumber of gallons of paint
needed...");
System.out.println ("Deck " + fmt.format(deckAmt));
System.out.println ("Big Ball " + fmt.format(ballAmt));
System.out.println ("Tank " + fmt.format(tankAmt));
}
}

```

Tabel 3. Soal 1(Petunjuk PaintThings.java)

1. Jalankan program dan perhatikan hasil untuk ketiga bentuk yang berbeda, screenshot hasil yang didapatkan dan lampirkan di dalam source code.
2. Simpan coding anda dengan nama package: **soal1**
3. Pastikan terdapat screenshoot pada repositori github

## A. Source Code

Shape.java	
1	package soall;
2	
3	public abstract class Shape {
4	protected String shapeName;
5	
6	public Shape(String shapeName){
7	this.shapeName = shapeName;
8	}
9	protected abstract double area();
10	public String toString(){
11	return shapeName;
12	}
13	}

Tabel 4. Source Code Soal 1(Shape.java)

Sphere.java	
1	package soall;
2	
3	public class Sphere extends Shape
4	{
5	private double radius; //radius in feet
6	//-----
7	// Constructor: Sets up the sphere.
8	//-----
9	public Sphere(double r)
10	{
11	super("Sphere");
12	radius = r;
13	}
14	//-----
15	// Returns the surface area of the sphere.
16	//-----
17	public double area()
18	{
19	return 4*Math.PI*(radius*radius);
20	}
21	//-----
22	// Returns the sphere as a String.
23	//-----
24	public String toString()
25	{
26	return super.toString() + " of radius " + radius;
27	}
28	}

Tabel 5. Source Code Soal 1 (Sphere.java)

Rectangle.java	
1	package soall;
2	
3	public class Rectangle extends Shape
4	{
5	private double length;
6	private double width;
7	public Rectangle(double l , double w)
8	{
9	super("Rectangle");
10	length = l;
11	width = w;
12	}
13	public double area()
14	{
15	return length * width;
16	}
17	public String toString()
18	{
19	return super.toString() + " of length " + length
20	+ " and width " + width;
21	}

Tabel 6. Source Code Soal 1 (Rectangle.java)

<b>Cylinder.java</b>	
1	package soall;
2	
3	public class Cylinder extends Shape
4	{
5	private double radius;
6	private double height;
7	public Cylinder(double r , double h)
8	{
9	super("Cylinder");
10	radius = r;
11	height = h;
12	}
13	public double area()
14	{
15	return Math.PI * (radius * radius) * height;
16	}
17	public String toString()
18	{
19	return super.toString() + " of radius " + radius
20	+ " and height " + height;
21	}

Tabel 7. Source Code Soal 1 (Cylinder.java)

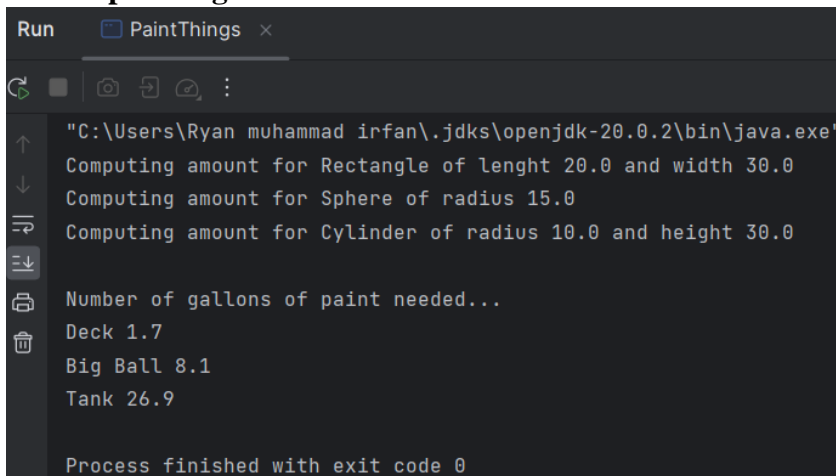
<b>Paint.java</b>	
1	package soall;
2	
3	public class Paint
4	{
5	private double coverage; //number of square feet per gallon
6	//-----
7	// Constructor: Sets up the paint object.
8	//-----
9	public Paint(double c)
10	{
11	coverage = c;
12	}
13	//-----
14	// Returns the amount of paint (number of gallons)
15	// needed to paint the shape given as the parameter.
16	//-----
17	public double amount(Shape s)
18	{
19	System.out.println ("Computing amount for " + s);
20	return s.area()/coverage;
21	}
22	}

Tabel 8. Source Code Soal 1 (Paint.java)

<b>PaintThings.java</b>	
1	package soall;
2	
3	import java.text.DecimalFormat;
4	public class PaintThings
5	{
6	//-----
7	// Creates some shapes and a Paint object
8	// and prints the amount of paint needed
9	// to paint each shape.
10	//-----
11	public static void main (String[] args)
12	{
13	final double COVERAGE = 350;
14	Paint paint = new Paint(COVERAGE);
15	Rectangle deck;
16	Sphere bigBall;
17	Cylinder tank;
18	double deckAmt, ballAmt, tankAmt;
19	// Instantiate the three shapes to paint
20	deck = new Rectangle(20,30);
21	bigBall = new Sphere(15);
22	tank = new Cylinder(10,30);
23	// Compute the amount of paint needed for each shape
24	deckAmt = paint.amount(deck);
25	ballAmt = paint.amount(bigBall);
26	tankAmt = paint.amount(tank);
27	// Print the amount of paint for each.
28	DecimalFormat fmt = new DecimalFormat("0.##");
29	System.out.println ("\nNumber of gallons of paint
30	needed...");
31	System.out.println ("Deck " + fmt.format(deckAmt));
32	System.out.println ("Big Ball " +
33	fmt.format(ballAmt));
34	System.out.println ("Tank " + fmt.format(tankAmt));
	}
	}

Tabel 9. Source Code Soal 1 (PaintThings.java)

## B. Output Program



```
Run PaintThings x
"C:\Users\Ryan muhammad irfan\.jdk\openjdk-20.0.2\bin\java.exe"
Computing amount for Rectangle of lenght 20.0 and width 30.0
Computing amount for Sphere of radius 15.0
Computing amount for Cylinder of radius 10.0 and height 30.0
Number of gallons of paint needed...
Deck 1.7
Big Ball 8.1
Tank 26.9
Process finished with exit code 0
```

Gambar 3. Screenshot Hasil Jawaban Soal 1

## C. Pembahasan

### Shape.java

Pada baris [1], `package soal1;`

“package soal1” berfungsi untuk mendeklarasikan package dengan nama soal1. Class Shape berada dalam package soal1.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [3-13] `public abstract class Shape{...}`

“public” berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

“abstract” berfungsi untuk menyatakan bahwa class ini masih dalam bentuk abstrak tidak bisa dibuat menjadi objek.

“class” berfungsi untuk membuat class yang dalam baris ini diberi nama Shape.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [4], `protected String shapeName;`

“protected” berfungsi sebagai penanda bahwa atribut dapat memiliki akses terbatas. Hanya bisa diakses dari dalam class itu sendiri, child class, dan class yang berada dalam package yang sama.

“String” merupakan tipe data dari atribut shapeName.

“shapeName” merupakan atribut dari sebuah class Shape.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [6-8], `public Shape(String shapeName){...}`

“public” berfungsi sebagai penanda bahwa constructor dapat diakses dari class lain.

“Shape(String shapeName){...}” merupakan sebuah constructor dari class Shape. Constructor ini menerima parameter String shapeName.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [7], `this.shapeName = shapeName;`

“this.shapeName = shapeName” berfungsi untuk menginisialisasi atribut “shapeName” dengan nilai yang diterima oleh parameter “shapeName”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [9], `protected abstract double area();`

“protected” berfungsi sebagai penanda bahwa method ini dapat diakses dari class lain, child class, dan class yang berada dalam package yang sama.

“double” berfungsi menyatakan method tersebut mengembalikan nilai dengan tipe data double.

“area()” merupakan sebuah method dengan nama area.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [10-12], `public String toString() {...}`

“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“String” merupakan tipe data yang dikembalikan oleh method toString().

“toString()” merupakan sebuah method dengan nama toString.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [11], `return shapeName;`

“return shapeName” berfungsi untuk mengembalikan nilai atribut ‘shapeName’ ketika method toString() dipanggil.

“;” berfungsi untuk mengakhiri sebuah perintah.

## **Sphere.java**

Pada baris [1], `package soal1;`

“package soal1” berfungsi untuk mendeklarasikan package dengan nama soal1. Class Sphere berada dalam package soal1.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [3-28], `public class Sphere extends Shape { ... }`

“public” berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

“class” berfungsi untuk membuat class yang dalam baris ini diberi nama Sphere.

“extends Shape” berfungsi untuk mewariskan method atau atribut dari kelas induk. Dalam hal ini kelas induknya Shape.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [5], `private double radius;`

“private” berfungsi sebagai penanda bahwa atribut dapat memiliki akses terbatas. Hanya bisa diakses dari dalam class itu sendiri.

“double” merupakan tipe data dari atribut radius.

“radius” merupakan atribut dari sebuah class Sphere.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [9-13], `public Sphere(double r)`

“public” berfungsi sebagai penanda bahwa constructor dapat diakses dari class lain.

`Sphere(double r){...}` merupakan sebuah constructor dari class Sphere. Constructor ini menerima parameter double r.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [11], `super("Sphere");`

“super(“Sphere”)” berfungsi untuk memanggil constructor parent dan memberikan argument “Sphere”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [12], `radius = r;`

“radius = r” berfungsi untuk menginisialisasi atribut “radius” dengan nilai yang diterima oleh parameter “r”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [17-20], `public double area() {...}`

“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“double” merupakan tipe data yang dikembalikan oleh method area().

“area()” merupakan sebuah method dengan nama area. Isi method ini berfungsi menghitung area dari sphere.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.



Pada baris [19], `return 4*Math.PI*(radius*radius);`

“`return 4*Math.PI*(radius*radius)`” berfungsi untuk mengembalikan nilai ‘`4 * Math.PI * (value radius * value radius)`’ ketika method `area()` dipanggil.

“`Math.PI`” berfungsi untuk menyatakan value phi.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [24-27], `public String toString() {...}`

“`public`” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“`String`” merupakan tipe data yang dikembalikan oleh method `toString()`.

“`toString()`” merupakan sebuah method dengan nama `toString`.

“`{...}`” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [26], `return super.toString() + " of radius " + radius;`

“`return super.toString() + " of radius " + radius`” berfungsi untuk mengembalikan nilai dengan mengakses kelas induk + “ `of radius` ” + value radius ketika method `toString()` dipanggil.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

## **Rectangle.java**

Pada baris [1], `package soal1;`

“`package soal1`” berfungsi untuk mendeklarasikan package dengan nama `soal1`. Class `Rectangle` berada dalam package `soal1`.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [3-21], `public class Rectangle extends Shape {...}`

“`public`” berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

“`class`” berfungsi untuk membuat class yang dalam baris ini diberi nama `Rectangle`.

“`extends Shape`” berfungsi untuk mewariskan method atau atribut dari kelas induk. Dalam hal ini kelas induknya `Shape`.

“`{...}`” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [5], `private double length;`

“`private`” berfungsi sebagai penanda bahwa atribut dapat memiliki akses terbatas. Hanya bisa diakses dari dalam class itu sendiri.

“`double`” merupakan tipe data dari atribut `length`.

“`length`” merupakan atribut dari sebuah class `Rectangle`.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [6], `private double width;`

“private” berfungsi sebagai penanda bahwa atribut dapat memiliki akses terbatas. Hanya bisa diakses dari dalam class itu sendiri.

“double” merupakan tipe data dari atribut width.

“width” merupakan atribut dari sebuah class Rectangle.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [7-12], `public Rectangle(double l , double w){...}`

“public” berfungsi sebagai penanda bahwa constructor dapat diakses dari class lain.

`Rectangle(double l , double w){...}` merupakan sebuah constructor dari class Rectangle. Constructor ini menerima parameter double l, double w.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [9], `super("Rectangle");`

“super(“Rectangle”)” berfungsi untuk memanggil constructor parent dan memberikan argument “Rectangle” .

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [10], `length = l;`

“length = l” berfungsi untuk menginisialisasi atribut “length” dengan nilai yang diterima oleh parameter “l”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [11], `width = w;`

“width = w” berfungsi untuk menginisialisasi atribut “width” dengan nilai yang diterima oleh parameter “w”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [13-16], `public double area() {...}`

“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“double” merupakan tipe data yang dikembalikan oleh method area().

“area()” merupakan sebuah method dengan nama area. Isi method ini berfungsi menghitung area dari Rectangle.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [15], `return length * width;`

“return length \* width” berfungsi untuk mengembalikan nilai value length \* value width ketika method area() dipanggil.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [17-20], `public String toString() {...}`

“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“String” merupakan tipe data yang dikembalikan oleh method `toString()`.

“`toString()`” merupakan sebuah method dengan nama `toString`.

“`{...}`” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [19], `return super.toString() + " of length " + length + " and width " + width;`

“`return super.toString() + " of length " + length + " and width " + width`” berfungsi untuk mengembalikan nilai dengan mengakses kelas induk + “ `of length` ” + value `length` + “ `and width` ” + value `width` ketika method `toString()` dipanggil.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

### **Cylinder.java**

Pada baris [1], `package soal1;`

“`package soal1`” berfungsi untuk mendeklarasikan package dengan nama `soal1`. Class `Cylinder` berada dalam package `soal1`.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [3-21], `public class Cylinder extends Shape{...}`

“public” berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

“class” berfungsi untuk membuat class yang dalam baris ini diberi nama `Cylinder`.

“`extends Shape`” berfungsi untuk mewariskan method atau atribut dari kelas induk. Dalam hal ini kelas induknya `Shape`.

“`{...}`” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [5], `private double radius;`

“private” berfungsi sebagai penanda bahwa atribut dapat memiliki akses terbatas. Hanya bisa diakses dari dalam class itu sendiri.

“double” merupakan tipe data dari atribut `radius`.

“radius” merupakan atribut dari sebuah class `Cylinder`.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [6], `private double height;`

“private” berfungsi sebagai penanda bahwa atribut dapat memiliki akses terbatas. Hanya bisa diakses dari dalam class itu sendiri.

“double” merupakan tipe data dari atribut height.

“Height” merupakan atribut dari sebuah class Cylinder.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [7-12], `public Cylinder(double r , double h) {...}`

“public” berfungsi sebagai penanda bahwa constructor dapat diakses dari class lain.

`Cylinder(double r , double h){...}` merupakan sebuah constructor dari class Rectangle. Constructor ini menerima parameter double r, double h.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [9], `super("Cylinder") ;`

“super(“Cylinder”)” berfungsi untuk memanggil constructor parent dan memberikan argument “Cylinder” .

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [10], `radius = r;`

“radius = r” berfungsi untuk menginisialisasi atribut “radius” dengan nilai yang diterima oleh parameter “r”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [11], `height = h;`

“height = h” berfungsi untuk menginisialisasi atribut “height” dengan nilai yang diterima oleh parameter “h”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [13-16], `public double area() {...}`

“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“double” merupakan tipe data yang dikembalikan oleh method area().

“area()” merupakan sebuah method dengan nama area. Isi method ini berfungsi menghitung area dari Cylinder.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [15], `return Math.PI * (radius * radius) * height;`

“return Math.PI \* (radius \* radius) \* height” berfungsi untuk mengembalikan nilai Math.PI + (value radius \* value radius) \* value height ketika method area() dipanggil.

“Math.PI” berfungsi untuk menyatakan value phi.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [17-20], `public String toString() {...}`

“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“String” merupakan tipe data yang dikembalikan oleh method toString().

“toString()” merupakan sebuah method dengan nama toString.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [19], `return super.toString() + " of radius " + radius + " and height " + height;`

“return super.toString() + " of radius " + radius + " and height " + height” berfungsi untuk mengembalikan nilai dengan mengakses kelas induk + “ of radius ” + value radius + “ and height ” + value height ketika method toString() dipanggil.

“;” berfungsi untuk mengakhiri sebuah perintah.

### **Paint.java**

Pada baris [1], `package soal1;`

“package soal1” berfungsi untuk mendeklarasikan package dengan nama soal1. Class Paint berada dalam package soal1.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [3-22], `public class Paint{...}`

“public” berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

“class” berfungsi untuk membuat class yang dalam baris ini diberi nama Paint.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [5], `private double coverage;`

“private” berfungsi sebagai penanda bahwa atribut dapat memiliki akses terbatas. Hanya bisa diakses dari dalam class itu sendiri.

“double” merupakan tipe data dari atribut coverage.

“coverage” merupakan atribut dari sebuah class Paint.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [9-12], `public Paint(double c){...}`

“public” berfungsi sebagai penanda bahwa constructor dapat diakses dari class lain.

`Paint(double c){...}` merupakan sebuah constructor dari class Paint. Constructor ini menerima parameter double c.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [11], `coverage = c;`

“coverage = c” berfungsi untuk menginisialisasi atribut “coverage” dengan nilai yang diterima oleh parameter “c”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [17-21], `public double amount(Shape s) {...}`

“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“double” merupakan tipe data yang dikembalikan oleh method amount().

“amount(Shape s)” merupakan sebuah method dengan nama amount. Menerima parameter ‘Shape’ dengan nama ‘s’

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [19], `System.out.println ("Computing amount for " + s);`

“System.out.println(...)” berfungsi untuk mencetak angka atau karakter dengan pindah ke baris berikutnya. Dalam baris ini mencetak "Computing amount for " + value s.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [20], `return s.area()/coverage;`

“return s.area()/coverage” berfungsi untuk mengembalikan nilai dengan memanggil parameter Shape dengan nama s pada method area / coverage.

“;” berfungsi untuk mengakhiri sebuah perintah.

## **PaintThings.java**

Pada baris [1], `package soal1;`

“package soal1” berfungsi untuk mendeklarasikan package dengan nama soal1. Class PaintThings berada dalam package soal1.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [3], `import java.text.DecimalFormat;`

Mengimpor class DecimalFormat yang terdapat di dalam package java,text.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [4-34], `public class PaintThings{...}`

“public” berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

“class” berfungsi untuk membuat class yang dalam baris ini diberi nama Main.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [11-33], `public static void main(String[] args){...}`

“public” berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

“static” berfungsi membuat suatu method tanpa perlu melakukan instansiasi terlebih dahulu.

“void” berfungsi untuk tidak mengembalikan nilai apapun.

“main” merupakan nama fungsi yang digunakan oleh java sebagai awal masuk ke program.

“String[] args” berfungsi sebagai parameter yang diperlukan oleh fungsi main. Parameter ini adalah array dari argument perintah yang bisa diteruskan ke program java.

“{...}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [13], `final double COVERAGE = 350;`

“final” berfungsi untuk bahwa suatu variabel tidak dapat diubah.

“double” merupakan tipe datanya, “COVERAGE” merupakan variabel yang diberi sebuah penamaan, Lalu setelah itu adalah valuenya 350. <tipe data> <nama variabel> = <nilai variabel>.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [14], `Paint paint = new Paint(COVERAGE);`

“Paint” merupakan sebuah class.

“paint” merupakan nama variabel. Digunakan untuk merujuk ke objek yang akan dibuat dengan menggunakan class Paint.

“new Paint” membuat objek baru dari kelas Paint.

“(COVERAGE)” berfungsi untuk mengisi value atribut dari parameter yang sudah ditentukan. Pada baris ini mengisi value atribut dengan parameter c: value variabel COVERAGE.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [15], `Rectangle deck;`

Berfungsi untuk mendeklarasikan variabel ‘deck’ yang dapat menyimpan objek dari kelas Rectangle.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [16], `Sphere bigBall;`

Berfungsi untuk mendeklarasikan variabel ‘bigBall’ yang dapat menyimpan objek dari kelas Sphere.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [17], `Cylinder tank;`

Berfungsi untuk mendeklarasikan variabel 'tank' yang dapat menyimpan objek dari kelas Cylinder.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [18], `double deckAmt, ballAmt, tankAmt;`

“Double” merupakan tipe data dari atribut deckAmt, ballAmt, tankAmt.

“deckAmt, ballAmt, tankAmt” merupakan variabel.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [20], `deck = new Rectangle(20, 30);`

“deck” merupakan nama variabel. Digunakan untuk merujuk ke objek yang akan dibuat dengan menggunakan class yang sudah ditentukan sebelumnya.

“new Rectangle” membuat objek baru dari kelas Rectangle.

“Rectangle(20,30)” berfungsi untuk mengisi value atribut dari parameter yang sudah ditentukan. Pada baris ini mengisi value atribut dengan parameter l: 20, w: 30.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [21], `bigBall = new Sphere(15);`

“bigBall” merupakan nama variabel. Digunakan untuk merujuk ke objek yang akan dibuat dengan menggunakan class yang sudah ditentukan sebelumnya.

“new Sphere” membuat objek baru dari kelas Sphere.

“Sphere(15)” berfungsi untuk mengisi value atribut dari parameter yang sudah ditentukan. Pada baris ini mengisi value atribut dengan parameter r: 15.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [22], `tank = new Cylinder(10, 30);`

“tank” merupakan nama variabel. Digunakan untuk merujuk ke objek yang akan dibuat dengan menggunakan class yang sudah ditentukan sebelumnya.

“new Cylinder” membuat objek baru dari kelas Cylinder.

“Cylinder(10, 30)” berfungsi untuk mengisi value atribut dari parameter yang sudah ditentukan. Pada baris ini mengisi value atribut dengan parameter r: 10, h:30.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [24], `deckAmt = paint.amount(deck);`

Berfungsi untuk menginisialisasi variabel deckAmt dengan memanggil method amount dengan parameter deck pada objek “paint”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [25], `ballAmt = paint.amount(bigBall);`



Berfungsi untuk menginisialisasi variabel `ballAmt` dengan memanggil method `amount` dengan parameter `bigBall` pada objek “`paint`”.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [26], `tankAmt = paint.amount(tank);`

Berfungsi untuk menginisialisasi variabel `tankAmt` dengan memanggil method `amount` dengan parameter `tank` pada objek “`paint`”.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [28], `DecimalFormat fmt = new DecimalFormat("0.#");`

“`DecimalFormat`” merupakan sebuah class dalam java yang digunakan untuk memformat angka desimal.

“`fmt`” merupakan nama variabel. Digunakan untuk merujuk ke objek yang akan dibuat dengan menggunakan class yang sudah ditentukan.

“`new DecimalFormat`” membuat objek baru dari kelas `DecimalFormat`.

“`DecimalFormat("0.#")`” berfungsi untuk mengisi value atribut dari parameter yang sudah ditentukan. Pada baris ini mengisi value atribut dengan parameter pattern: `0.#` . Dengan pattern: `0.#` dapat meformat angka decimal 1 angka dibelakang koma.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [29], `System.out.println ("\nNumber of gallons of paint needed...");`

“`System.out.println(...)`” berfungsi untuk mencetak angka atau karakter dengan pindah ke baris berikutnya. Dalam baris ini mencetak “`\nNumber of gallons of paint needed...`”.

“`\n`” berfungsi untuk pindah ke baris berikutnya (newline).

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [30], `System.out.println ("Deck " + fmt.format(deckAmt));`

“`System.out.println(...)`” berfungsi untuk mencetak angka atau karakter dengan pindah ke baris berikutnya. Dalam baris ini mencetak “`Deck` ” + value `deckAmt` dengan format yang digunakan objek ‘`fmt`’.

“`;`” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [31], `System.out.println ("Big Ball " + fmt.format(ballAmt));`

“`System.out.println(...)`” berfungsi untuk mencetak angka atau karakter dengan pindah ke baris berikutnya. Dalam baris ini mencetak “`Big Ball` ” + value `ballAmt` dengan format yang digunakan objek ‘`fmt`’.

“`;`” berfungsi untuk mengakhiri sebuah perintah.=

Pada baris [32], `System.out.println ("Tank " + fmt.format(tankAmt));`

“System.out.println(...)” berfungsi untuk mencetak angka atau karakter dengan pindah ke baris berikutnya. Dalam baris ini mencetak "Tank " + value tankAmt dengan format yang digunakan objek ‘fmt’.

“;” berfungsi untuk mengakhiri sebuah perintah.=

#### **D. Tautan GIT**

<https://github.com/ryanmi04/Praktikum-Pemrograman-2-Paralel-1/tree/main/PRAKTIKUM%205/src/soal1>