

LAPORAN PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

BAB : INTERFACE
 NAMA : Dani Adrian
 NIM : 225150201111009
 ASISTEN : Tengku Muhammad Rafi Rahardiansyah
 Muhammad Bin Djafar Almasyhur
 TGL PRAKTIKUM : 10 Mei 2023

A. Interface

Data dan Hasil Percobaan

Pertanyaan

1. Lakukan percobaan diatas dan benahi jika menemukan kesalahan serta jelaskan!

Source Code

Rectangle.java	
1.	package praktikum1;
2.	interface Colorable {
3.	public void howToColor();
4.	}
5.	
6.	interface Comparable {
7.	public void compareTo(Object obj);
8.	}
9.	public class Rectangle implements Colorable, Comparable {
	// lass rectanggle
10.	private String warna;
11.	private int kategori;
12.	
13.	public Rectangle() {
14.	}
15.	
16.	public Rectangle(String warna) {
17.	this.warna = warna;
18.	}
19.	
20.	public void howToColor() {
21.	if (this.warna == null) {
22.	System.out.println("tidak ada warna, warna
	bangun kotak masih polos");
23.	} else {
24.	System.out.println("bangun kotak sudah
	diwarnai dengan warna " + this.warna);

```

25.     }
26.   }
27.   public void compareTo(Object obj) {
28.       this.kategori = (int) obj;
29.       if (this.kategori == 0) {
30.           System.out.println("ukuran cat yang cocok
untuk bangun kotak dengan ukuran kategori " +
this.kategori + " yaitu 2.5L");
31.       } else {
32.           System.out.println("ukuran cat yang cocok
untuk bangun kotak dengan ukuran kategori " +
this.kategori + " yaitu 6.5L");
33.       }
34.   }
35.
36.   public static void main(String[] args) {
37.       Rectangle kotak1 = new Rectangle("merah");
38.       Rectangle kotak2= new Rectangle();
39.       Rectangle kotak3 = new Rectangle();
40.       kotak1.howToColor();
41.       kotak2.howToColor();
42.       kotak3.compareTo(4);
43.   }
44. }

```

```
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c:: cd 'C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' -cp 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771864e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' praktikump1.Rectangle
bangun kotak sudah diwarnai dengan warna merah
tidak ada warna, warna bangun kotak masih polos
ukuran cat yang cocok untuk bangun kotak dengan ukuran kategori 4 yaitu 6.5L
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan>
```

Program dapat dijalankan, akan tetapi ada kesalahan sedikit pada bagian interface nya. Seharusnya menghilangkan tulisan public pada line 3 dan 7 jika ingin dibuat didalam 1 file.

2. Apakah class yang berbentuk Interface bisa diinstansiasi menjadi sebuah objek? Jelaskan alasannya!

1	Tulis source code di sini pake courier new 12
---	---

Output

Penjelasan

Tidak, sebuah interface tidak dapat diinstansiasi menjadi sebuah objek karena secara konsep, interface hanya berisi definisi metode atau perilaku yang harus diimplementasikan oleh kelas-kelas yang menerapkannya. Interface tidak memiliki implementasi konkrit dari metode-metode tersebut dan tidak memiliki variabel anggota. Oleh karena itu, tidak mungkin untuk menginstansiasi sebuah interface menjadi sebuah objek.

Namun, kelas yang menerapkan interface dapat diinstansiasi menjadi sebuah objek. Ketika sebuah kelas menerapkan sebuah interface, kelas tersebut harus memberikan implementasi konkret untuk semua metode yang didefinisikan dalam interface tersebut. Setelah itu, objek dari kelas tersebut dapat diinstansiasi dan digunakan untuk memanggil metode-metode tersebut.

Pertanyaan

3. Apakah class yang berbentuk interface bisa diinstansiasi menjadi sebuah objek? Jelaskan alasannya!

Source Code

Ouput

Penjelasan

Ya, suatu kelas dalam Java dapat mengimplementasikan lebih dari satu interface sekaligus. Hal ini memungkinkan kelas untuk mengakses metode dan variabel dari banyak interface yang berbeda secara bersamaan.

Alasan mengapa suatu kelas dapat mengimplementasikan banyak interface adalah untuk memungkinkan kelas tersebut untuk mendefinisikan perilaku atau fungsi yang lebih kompleks yang tidak dapat diwakili oleh satu interface saja. Dengan mengimplementasikan banyak interface, kelas tersebut dapat menggabungkan dan menggunakan fungsionalitas dari banyak interface yang berbeda secara bersamaan. Selain itu, dengan menggunakan lebih dari satu interface, kita dapat memisahkan spesifikasi perilaku menjadi beberapa interface yang berbeda, yang dapat membantu meningkatkan modularitas dan kesederhanaan kode.

Pertanyaan

4. Pada interface identitas.java hapus method tampilkan nama, amati apa yang terjadi dan mengapa demikian?

Source Code

Identitas.java	
1.	package latihaninterface;
2.	public interface Identitas {
3.	
4.	public void tampilkanUmur();
5.	}

Output**Penjelasan**

Jika kita menghapus method tampilkanNama() dari interface Identitas, maka akan terjadi error kompilasi pada kelas Manusia. Hal ini terjadi karena kelas Manusia mengimplementasikan interface Identitas dan secara kontrak harus memberikan implementasi untuk semua metode yang didefinisikan dalam interface tersebut. Dengan menghapus method tampilkanNama() dari interface Identitas, berarti kelas Manusia tidak lagi memberikan implementasi untuk metode tersebut. Oleh karena itu, error kompilasi akan muncul.

Pertanyaan

5. Jika pada class hewan kita hanya ingin mengimplements interface MakhlukHidup saja apa yang terjadi? Jelaskan

Source Code

Hewan.java	
1.	package latihaninterface;
2.	public class Hewan
3.	implements MakhlukHidup{
4.	
5.	public void makan() {
6.	System.out.println("Makan pakai tangan dan mulut");
7.	}
8.	
9.	public void berjalan() {
10.	System.out.println("Jalan pakai 4 kaki");
11.	}
12.	
13.	public void bersuara() {
14.	System.out.println("Suaranya nggak jelas");
15.	}
16.	public void tampilkanNama (){}
17.	public void tampilkanUmur (){}
18.	
19.	}

Output

Penjelasan

Sebenarnya jika kita hanya menghapus implements Identitas tidak terjadi error. Tetapi Jika kita menghapus deklarasi implements Identitas pada kelas Hewan, dan tetap mempertahankan implementasi metode tampilkanNama() dan tampilkanUmur(), maka akan terjadi error kompilasi.

Ketika kita mengimplementasikan sebuah interface pada sebuah kelas, kita secara kontrak harus memberikan implementasi untuk semua metode yang didefinisikan dalam interface tersebut. Jika kita menghapus deklarasi implements Identitas, berarti kelas Hewan tidak lagi berkewajiban untuk mengimplementasikan metode tampilkanNama() dan tampilkanUmur() yang terdapat dalam interface Identitas.

Namun, jika kita mempertahankan implementasi metode tampilkanNama() dan tampilkanUmur() tanpa mengimplementasikan interface Identitas, maka metode-metode tersebut tidak memiliki tempat yang valid dalam kelas Hewan. Ini akan menyebabkan error kompilasi karena metode-metode tersebut tidak memiliki definisi yang sesuai dalam konteks kelas Hewan. Untuk menghindari error kompilasi, jika kita tidak ingin mengimplementasikan interface Identitas pada kelas Hewan, maka perlu menghapus implementasi metode tampilkanNama() dan tampilkanUmur() yang ada di kelas Hewan.

Pertanyaan

6. Buatlah konstruktor pada manusia dengan parameter umur dan nama kemudian panggil pada Class Main dengan menginstan objek bernama nama anda!

Source Code

Manusia.java

```
1. package latihaninterface;
2. public class Manusia
3. implements MakhlukHidup, Identitas{
4.     private String nama;
5.     private int umur;
6.
7.     public Manusia(String nama, int umur){
8.         this.nama = nama;
9.         this.umur = umur;
10.    }
11.
12.    public void makan() {
13.        System.out.println("Makan pakai sendok garpu");
14.    }
15.
16.    public void berjalan() {
17.        System.out.println("Jalan pakai dua kaki");
18.    }
19.
20.    public void bersuara() {
21.        System.out.println("Suaranya merdu");
22.    }
23.
24.    public void tampilkanNama() {
25.        System.out.println("Nama saya: " + this.nama);
26.    }
27.
28.    public void tampilkanUmur() {
29.        System.out.println("Umur saya: " + this.umur);
30.    }
31. }
```

Main.java

```
1. package latihaninterface;
2.
3. public class Main {
4.     public static void main(String[] args) {
```

5.	<code>Manusia manusia = new Manusia("Dani Adrian ", 18);</code>
6.	<code>manusia.tampilkanNama();</code>
7.	<code>manusia.tampilkanUmur();</code>
8.	<code>}</code>
9.	<code>}</code>

Output

```

PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c++ cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771064e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' 'latihaninterface.Main'
Nama saya: Dani Adrian
Umur saya: 18
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan>

```

Penjelasan

Jadi disini saya menambahkan konstruktor pada kelas manusia dengan parameter umur dan nama, kemudian saya membuat main classnya di file baru untuk menginstansiasi objek Manusia dengan nama dan umur saya

Pertanyaan

- Ubah source code diatas menjadi proses meminta inputan dari user dan buat menjadi interaktif!

Source Code

Rectangle.java	
1.	<code>package praktikump1;</code>
2.	<code>import java.util.Scanner;</code>
3.	<code>interface Colorable {</code>
4.	<code> public void howToColor();</code>
5.	<code>}</code>
6.	
7.	<code>interface Comparable {</code>
8.	<code> public void compareTo(Object obj);</code>
9.	<code>}</code>
10.	
11.	<code>public class Rectangle implements Colorable, Comparable {</code>
	<code>// lass rectangle</code>
12.	<code> private String warna;</code>
13.	<code> private int kategori;</code>
14.	
15.	<code> public Rectangle() {</code>
16.	<code> }</code>
17.	
18.	<code> public Rectangle(String warna) {</code>
19.	<code> this.warna = warna;</code>

```
20.     }
21.
22.     public void howToColor() {
23.         if (this.warna == null) {
24.             System.out.println("tidak ada warna, warna
bangun kotak masih polos");
25.         } else {
26.             System.out.println("bangun kotak sudah
diwarnai dengan warna " + this.warna);
27.         }
28.     }
29.
30.     public void compareTo(Object obj) {
31.         this.kategori = (int) obj;
32.         if (this.kategori == 0) {
33.             System.out.println("ukuran cat yang cocok
untuk bangun kotak dengan ukuran kategori " +
this.kategori + " yaitu 2.5L");
34.         } else {
35.             System.out.println("ukuran cat yang cocok
untuk bangun kotak dengan ukuran kategori " +
this.kategori + " yaitu 6.5L");
36.         }
37.     }
38. public static void main (String[] args) {
39.     Scanner sc = new Scanner (System.in);
40.     System.out.println("Masukkan warna kotak : ");
41.     String warna = sc.nextLine();
42.     System.out.print("Masukkan kategori kotak : ");
43.     int kategori = sc.nextInt();
44.     if
45.     (warna.toLowerCase().equals("null")) {
46.         warna = null;
47.         Rectangle kotak = new Rectangle(warna);
48.         kotak.howToColor();
49.         kotak.compareTo(kategori);
50.     }else{
51.         Rectangle kotak = new Rectangle(warna);
52.         kotak.howToColor();
53.         kotak.compareTo(kategori);
54.     }
55.     sc.close();
56. }
57. }
```


Output

```

PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c++; cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771064e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' 'praktikum1.Rectangle'
Masukkan warna kotak :
null
Masukkan kategori kotak : 0
tidak ada warna, warna bangun kotak masih polos
ukuran cat yang cocok untuk bangun kotak dengan ukuran kategori 0 yaitu 2.5L
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c++; cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771064e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' 'praktikum1.Rectangle'
Masukkan warna kotak :
merah
Masukkan kategori kotak : 3
bangun kotak sudah diwarnai dengan warna merah
ukuran cat yang cocok untuk bangun kotak dengan ukuran kategori 3 yaitu 6.5L
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c++; cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771064e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' 'praktikum1.Rectangle'
Masukkan warna kotak :
kuning
Masukkan kategori kotak : 0
bangun kotak sudah diwarnai dengan warna kuning
ukuran cat yang cocok untuk bangun kotak dengan ukuran kategori 0 yaitu 2.5L
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan>

```

Penjelasan

Untuk proses meminta inputan dari pengguna dan membuatnya interaktif, kita dapat melakukannya pada kelas utama (main). Pada kelas utama, pertama-tama kita akan meminta input menggunakan Scanner. Input tersebut berupa warna dan kategori dari kotak. Input tersebut akan dimasukkan ke dalam objek Rectangle yang kemudian akan digunakan untuk memanggil method `howToColor()` dan `compareTo()`. Jika warna yang dimasukkan adalah "null", maka warna pada objek Rectangle akan di-set menjadi null, sedangkan jika warna yang dimasukkan bukan "null", maka warna pada objek Rectangle akan di-set sesuai dengan warna yang dimasukkan. Kemudian, method `howToColor()` akan dipanggil untuk menampilkan informasi apakah kotak sudah diwarnai atau belum. Selanjutnya, method `compareTo()` akan dipanggil untuk menampilkan ukuran cat yang cocok untuk kotak dengan kategori yang dimasukkan.

Pertanyaan

8. Buat objek selain objek diatas dengan menggunakan method yang berbeda dengan yang diatas! (min.1 contoh)

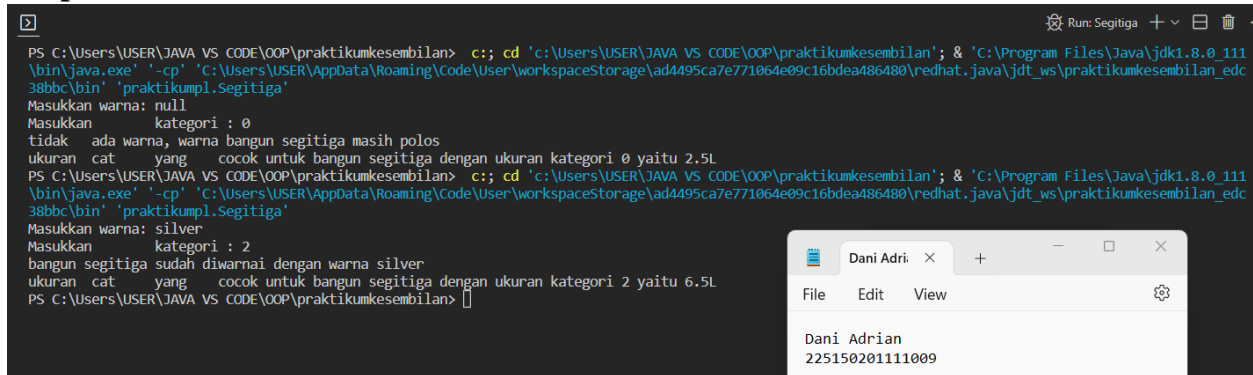
Source Code

Segitiga.java

```
1. package praktikumpl;
2.
3. import java.util.Scanner;
4.
5. interface Colorable {
6.     public void howToColor();
7. }
8.
9. interface Comparable {
10.    public void compareTo(Object obj);
11. }
12.
13. public class Segitiga implements Colorable, Comparable {
14.    private String warna;
15.    private int kategori;
16.
17.    public Segitiga() {
18.    }
19.
20.    public Segitiga(String warna) {
21.        this.warna = warna;
22.    }
23.
24.    public void howToColor() {
25.        if (this.warna == null) {
26.            System.out.println("tidak ada warna, warna
bangun segitiga masih polos");
27.        } else {
28.            System.out.println("bangun segitiga sudah
diwarnai dengan warna " + this.warna);
29.        }
30.    }
31.    public void compareTo(Object obj) {
32.        this.kategori = (int) obj;
33.        if (this.kategori == 0) {
34.            System.out.println(
35.                "ukuran cat yang cocok untuk bangun
```

```
        segitiga dengan ukuran kategori "
36.                + this.kategori + " yaitu
        2.5L");
37.    } else {
38.        System.out.println(
39.            "ukuran cat yang cocok untuk bangun
        segitiga dengan ukuran kategori "
40.                + this.kategori + " yaitu
        6.5L");
41.    }
42. }
43.
44. public static void main(String[] args) {
45.     Scanner p = new Scanner(System.in);
46.     System.out.print("Masukkan warna: ");
47.     String warna = p.nextLine();
48.     System.out.print("Masukkan kategori : ");
49.     int kategori = p.nextInt();
50.     if (warna.toLowerCase().equals("null")) {
51.         warna = null;
52.         Segitiga segitiga = new Segitiga(warna);
53.         segitiga.howToColor();
54.         segitiga.compareTo(kategori);
55.     } else {
56.         Segitiga segitiga = new Segitiga(warna);
57.         segitiga.howToColor();
58.         segitiga.compareTo(kategori);
59.     }
60.     p.close();
61. }
62. }
```

Output



```

PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c:: cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771064e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' 'praktikumpl.Segitiga'
Masukkan warna: null
Masukkan kategori : 0
tidak ada warna, warna bangun segitiga masih polos
ukuran cat yang cocok untuk bangun segitiga dengan ukuran kategori 0 yaitu 2.5L
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c:: cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771064e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' 'praktikumpl.Segitiga'
Masukkan warna: silver
Masukkan kategori : 2
bangun segitiga sudah diwarnai dengan warna silver
ukuran cat yang cocok untuk bangun segitiga dengan ukuran kategori 2 yaitu 6.5L
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan>
  
```

The screenshot shows a terminal window with the following output:

```

PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c:: cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771064e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' 'praktikumpl.Segitiga'
Masukkan warna: null
Masukkan kategori : 0
tidak ada warna, warna bangun segitiga masih polos
ukuran cat yang cocok untuk bangun segitiga dengan ukuran kategori 0 yaitu 2.5L
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c:: cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771064e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' 'praktikumpl.Segitiga'
Masukkan warna: silver
Masukkan kategori : 2
bangun segitiga sudah diwarnai dengan warna silver
ukuran cat yang cocok untuk bangun segitiga dengan ukuran kategori 2 yaitu 6.5L
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan>
  
```

Overlaid on the terminal is a small application window titled "Dani Adri". It has a menu bar with "File", "Edit", and "View". The main content area displays the name "Dani Adrian" and the ID "225150201111009".

Penjelasan

Kelas Segitiga menerapkan interface Colorable. Kelas Segitiga memiliki konstruktor yang menerima parameter warna. Metode howToColor() diimplementasikan untuk menampilkan pesan berdasarkan apakah Segitiga sudah diwarnai atau masih polos. Di dalam metode main(), kita membuat objek Segitiga dengan warna "Silver" dan memanggil metode howToColor() untuk menampilkan pesan terkait warna Segitiga tersebut.

Tugas Praktikum

Perusahaan NV. Meneer memiliki koperasi karyawan yang memungkinkan karyawannya berbelanja di koperasi tersebut. Tentunya, karyawan tersebut bisa membayar belanjanya tersebut di akhir bulan melalui pemotongan gaji. Ada 2 kelas yang terlibat disini, Invoice dan Karyawan. Kedua class tadi mengimplementasikan **interface Pembayaran** yang mana class tersebut hanya memiliki satu method yang harus diimplementasikan di kedua class, yaitu **getTotalPembayaran()**.

Program harus bisa mengolah gaji karyawan di akhir bulan beserta invoice belanjaan karyawan yang nantinya gaji karyawan perbulannya dikurangi total harga belanjanya. Tampilkan informasi dari karyawan tersebut beserta total gaji sebelum dan setelah dipotong hutang total belanjaan di koperasi dan tampilkan pula detail belanjanya.

1. Atribut dari Invoice:

String namaProduk, Integer kuantitas, Integer hargaPerItem

2. Atribut dari Employee:

Integer id, String nama, Integer gaji, Invoice[] invoices

Source Code

Invoice.java	
1.	public class Invoice implements Pembayaran {
2.	private String namaProduk;
3.	private int kuantitas;
4.	private int hargaPerItem;
5.	
6.	Invoice(String namaProduk, int kuantitas, int
	hargaPerItem) {
7.	this.namaProduk = namaProduk;
8.	this.kuantitas = kuantitas;
9.	this.hargaPerItem = hargaPerItem;
10.	}
11.	
12.	public String getNamaProduk() {
13.	return namaProduk;
14.	}
15.	
16.	public void setNamaProduk(String namaProduk) {
17.	this.namaProduk = namaProduk;
18.	}
19.	
20.	public int getKuantitas() {

```

21.         return kuantitas;
22.     }
23.
24.     public void setKuantitas(int kuantitas) {
25.         this.kuantitas = kuantitas;
26.     }
27.
28.     public int getHargaPerItem() {
29.         return hargaPerItem;
30.     }
31.
32.     public void setHargaPerItem(int hargaPerItem) {
33.         this.hargaPerItem = hargaPerItem;
34.     }
35.
36.     public int getTotalPembayaran() {
37.         return getKuantitas() * getHargaPerItem();
38.     }
39.
40.     public String print() {
41.         return "===== BELANJA
KARYAWAN===== " + "\n" + "Nama Produk :
" + getNamaProduk() + "\n" + "Kuantitas : " +
getKuantitas() + "\n" + "Harga per barang: " +
getHargaPerItem() + "\n" + "Total harga : " +
getTotalPembayaran() + "\n";
42.     }
43. }

```

Pembayaran.java

```

1. public interface Pembayaran {
2.     public int getTotalPembayaran();
3. }

```

Karyawan.java

```

1. public class Karyawan implements Pembayaran {
2.     private int id;
3.     private int gaji;
4.     private String nama;
5.     private Invoice[]
6.     invoices = new Invoice[1000];
7.     private int invoiceCount = 0;
8.     public Karyawan(int id, int gaji, String nama) {

```

```
9.      this.id = id;
10.     this.gaji = gaji;
11.     this.nama = nama; }
12. public int getTotalPembayaran() {
13.     int total = 0;
14.     for (int i = 0; i < invoiceCount; i++) {
15.         total += invoices[i].getTotalPembayaran();
16.     }
17. return total;
18.     }
19. public Invoice[] getInvoices() {
20.     Invoice[] result = new Invoice[invoiceCount];
21.     for (int i = 0; i < invoiceCount; i++)
22.     {
23.         result[i] = invoices[i];
24.     }
25.     return result;
26. }
27. public int getInvoiceCount() {
28.     return invoiceCount;
29. }
30. public int getId() {
31.     return id;
32. }
33. public void setId(int id) {
34.     this.id = id;
35. }
36. public int getGaji() {
37.     return gaji;
38. }
39. public void setGaji(int gaji) {
40.     this.gaji = gaji;
41. }
42. public String getNama() {
43.     return nama;
44. }
45. public void setNama(String nama) {
46.     this.nama = nama;
47. }
48.     public int gajiFinal() {
49.         return      getGaji() - getTotalPembayaran();
50.     }
51. public void addInvoice(Invoice invoice) {
52.     invoices[invoiceCount++] = invoice;
```

```

53. }
54. public String print() {
55.     return "===== Karyawan
===== " + "\n" + "ID Karyawan: " +
getId() + "\n" + "Nama Karyawan: " + getName() + "\n" +
"Gaji Karyawan: " + getGaji() + "\n";
56. }
57. public String getDetailBelanja() {
58. return "==== Total gaji sebelum & setelah dipotong hutang
total belanja di koperasi ===== " + "\n" + "Total gaji : " +
getGaji() + "\n" + "Total gaji Karyawan setelah di potong :
" + gajiFinal();
59. }
60. }

```

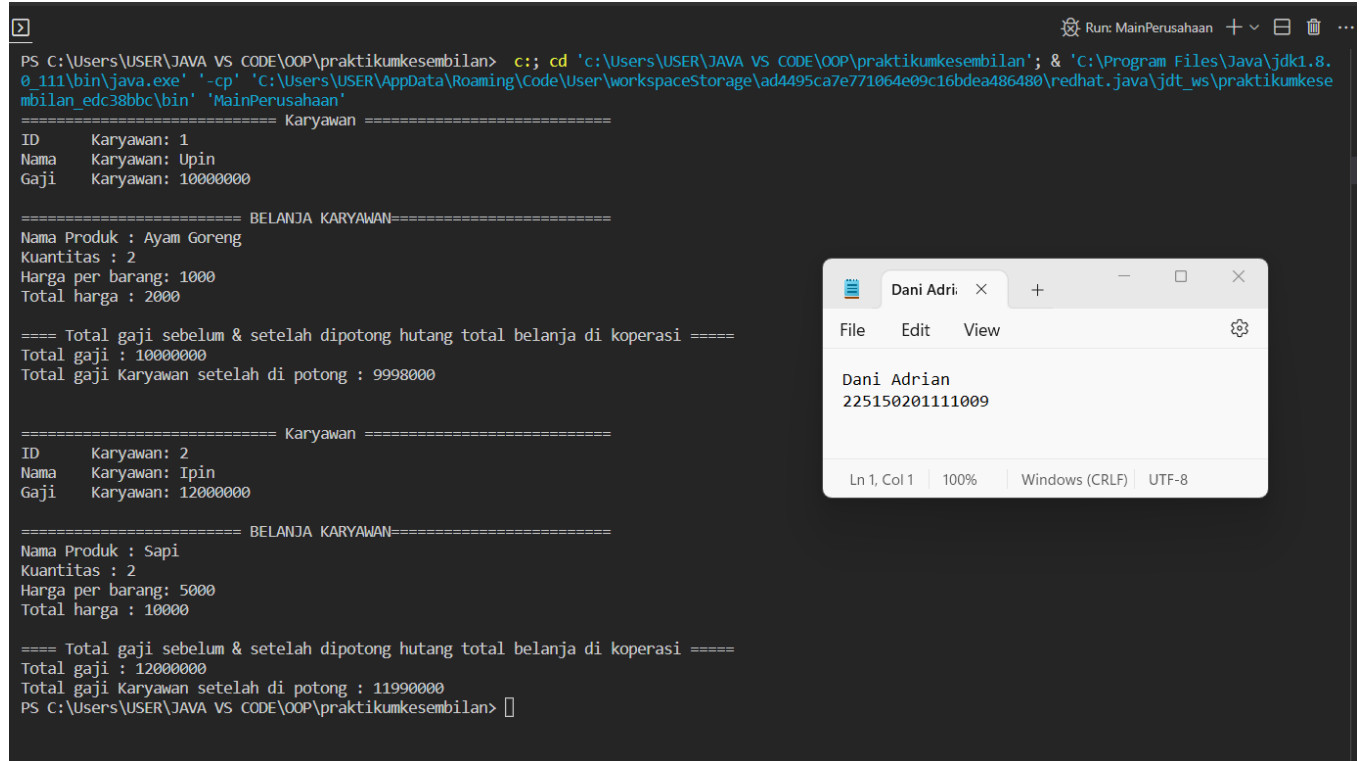
MainPerusahaan.java

```

1. public class MainPerusahaan {
2. public static void main(String[] args) {
3.     Invoice p1 = new Invoice("Ayam Goreng", 2, 1000);
4.     Karyawan q1 = new Karyawan(01, 10000000, "Upin");
5.     Invoice p2 = new Invoice("Sapi", 2, 5000);
6.     Karyawan q2 = new Karyawan(02, 12000000, "Ipin");
7.     q1.addInvoice(p1);
8.     System.out.println(q1.print());
9.     for (Invoice invoice : q1.getInvoices()) {
10. System.out.println(invoice.print());
11. }
12. System.out.println(q1.getDetailBelanja());
13. System.out.println();
14. System.out.println();
15. q2.addInvoice(p2);
16. System.out.println(q2.print());
17. for (Invoice invoice : q2.getInvoices()) {
18. System.out.println(invoice.print());
19. }
20. System.out.println(q2.getDetailBelanja());
21. }
22. }

```


Output



The screenshot shows a Java IDE with a command prompt window open. The command prompt displays the output of a Java program. The program first processes a transaction for a worker named Upin, showing their salary and a purchase of Ayam Goreng. It then calculates the total salary after deductions. Next, it processes a transaction for a worker named Ipin, showing their salary and a purchase of Sapi. It also calculates the total salary after deductions. A web browser window is also open, displaying a page with the name 'Dani Adrian' and the ID '225150201111009'.

```
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan> c:: cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan'; & 'c:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'c:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\ad4495ca7e771064e09c16bdea486480\redhat.java\jdt_ws\praktikumkesembilan_edc38bbc\bin' 'MainPerusahaan'
```

```
===== Karyawan =====
ID      Karyawan: 1
Nama    Karyawan: Upin
Gaji    Karyawan: 10000000

===== BELANJA KARYAWAN=====
Nama Produk : Ayam Goreng
Kuantitas : 2
Harga per barang: 1000
Total harga : 2000

==== Total gaji sebelum & setelah dipotong hutang total belanja di koperasi ====
Total gaji : 10000000
Total gaji Karyawan setelah di potong : 9998000

===== Karyawan =====
ID      Karyawan: 2
Nama    Karyawan: Ipin
Gaji    Karyawan: 12000000

===== BELANJA KARYAWAN=====
Nama Produk : Sapi
Kuantitas : 2
Harga per barang: 5000
Total harga : 10000

==== Total gaji sebelum & setelah dipotong hutang total belanja di koperasi ====
Total gaji : 12000000
Total gaji Karyawan setelah di potong : 11990000
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkesembilan>
```

Penjelasan

Interface Pembayaran:

Interface ini menyediakan metode getTotalPembayaran() yang akan diimplementasikan oleh kelas-kelas lain.

Class Invoice:

Kelas ini mengimplementasikan interface Pembayaran.

Memiliki atribut-atribut seperti namaProduk, kuantitas, dan hargaPerItem yang digunakan untuk menghitung total pembayaran.

Terdapat konstruktor yang digunakan untuk menginisialisasi nilai-nilai atribut.

Terdapat getter dan setter yang digunakan untuk mengakses dan mengubah nilai atribut.

Implementasi metode getTotalPembayaran() yang menghitung total pembayaran dengan mengalikan kuantitas dengan hargaPerItem.

Metode print() digunakan untuk mencetak informasi detail tentang pembelian.

Class Karyawan:

Kelas ini juga mengimplementasikan interface Pembayaran.

Memiliki atribut-atribut seperti id, gaji, dan nama yang berhubungan dengan informasi karyawan.

Terdapat array invoices yang menyimpan objek Invoice yang terkait dengan karyawan.

Terdapat konstruktor yang digunakan untuk menginisialisasi nilai-nilai atribut.

Terdapat getter dan setter yang digunakan untuk mengakses dan mengubah nilai atribut.

Implementasi metode getTotalPembayaran() yang menghitung total pembayaran karyawan dengan menjumlahkan total pembayaran dari setiap invoice yang dimilikinya.

Metode addInvoice() digunakan untuk menambahkan invoice ke dalam array invoices.

Metode print() digunakan untuk mencetak informasi detail tentang karyawan.

Metode getDetailBelanja() digunakan untuk mencetak informasi total gaji karyawan sebelum dan setelah dipotong total pembelian.

Class MainPerusahaan:

Pada metode main, terdapat pembuatan objek Invoice p1 dan p2, serta objek Karyawan q1 dan q2.

Invoice p1 dan p2 digunakan untuk merepresentasikan pembelian yang terkait dengan karyawan q1 dan q2.

Selanjutnya, invoice p1 ditambahkan ke karyawan q1 menggunakan metode addInvoice().

Setelah itu, dilakukan pencetakan informasi detail tentang karyawan q1 dan setiap invoice yang dimilikinya menggunakan perulangan for-each.

Kemudian, dilakukan pencetakan informasi mengenai total gaji sebelum dan setelah dipotong total pembelian karyawan q1 menggunakan metode getDetailBelanja().

Langkah-langkah ini diulang untuk karyawan q2.