MODUL 10 – Interface

10.1. CAPAIAN PEMBELAJARAN

1 Mahasiswa mampu Memahami konsep dan tujuan penggunaan interface dalam PBO

2 Mahasiswa Mampu membuat dan mengimplementasikan interface pada kelas Java

3 Mahasiswa mampu Menggunakan interface untuk menciptakan abstraksi yang dapat diterapkan pada berbagai kelas

10.2. ALAT DAN BAHAN

1 Seperangkat komputer lengkap/Laptop dengan koneksi internet

2 Web Browser (Chrome/Firefox/Opera/Edge/Safari/dll)

3 Aplikasi Kantor (Microsoft Office/Libre Office/WPS Office/etc)

4 JDK (https://www.oracle.com/java/technologies/downloads/)

5 Netbeans (https://netbeans.apache.org/front/main/download/)

10.3. DASAR TEORI

10.3.1. Interface

Interface dalam Java adalah bentuk kontrak dalam pemrograman berorientasi objek yang mendefinisikan metode tanpa implementasi. Interface mirip dengan kelas abstrak, tetapi interface hanya mendeklarasikan metode yang perlu diimplementasikan oleh kelas yang menggunakannya. Interface juga tidak dapat memiliki atribut non-final, dan semua metodenya adalah public dan abstract secara default.

Interface berbeda dengan class karena di dalamnya hanya berisi method kosong dan konstanta. Method-method dalam interface tidak memiliki statement atau implementasi, sehingga deklarasi method dalam interface sama dengan deklarasi abstract method pada abstract class. Semua method yang dideklarasikan dalam interface secara otomatis bersifat public dan abstract, sementara variabel dalam interface adalah public, static, dan final secara default.

10.3.2. Mengimplementasikan interface

Ketika sebuah class mengimplementasikan suatu interface, class tersebut akan memperoleh semua konstanta dan method yang ada di dalam interface. Berbeda dengan pewarisan class, interface hanya menyediakan kontrak tanpa implementasi, sehingga class yang mengimplementasikan interface bertanggung jawab untuk menyediakan kode atau logika pada setiap method yang ada dalam

interface tersebut. Konstanta dalam interface, yang bersifat public, static, dan final, akan dapat diakses oleh class yang mengimplementasikannya seperti layaknya variabel global.

Jika sebuah class memilih untuk tidak mengimplementasikan semua method yang dideklarasikan dalam interface, class tersebut harus dideklarasikan sebagai abstract. Deklarasi ini menandakan bahwa class belum sepenuhnya mengimplementasikan kontrak yang diberikan oleh interface dan tidak bisa diinstansiasi secara langsung. Class abstract ini kemudian dapat diwariskan ke subclass lain yang akan melengkapi implementasi method-method yang masih kosong.

Deklarasi Implementasi Interface :

```
<modifier> class <name> [extends <super class>]
[implements <interface> [, <interface>]* ] {
<declarations>*
}
```

Contoh kode program :

```
1. public class Line implements Relation {
2. private double x1;
3. private double x2;
4. private double y1;
5. private double y2;
6.
7. public Line(double x1, double x2, double y1, double y2){
8.   this.x1 = x1;
9.   this.x2 = x2;
10.  this.y1 = y1;
11.  this.y2 = y2;
12.}
13.
14.public double getLength(){
15.  double length = Math.sqrt((x2-x1)*(x2-x1) + (y2-y1)* (y2-
y1));
16.  return length;
17.}
```

10.3.3. Inheritance pada interface

Kita dapat membuat subinterface dengan menggunakan kata kunci extends, yang memungkinkan satu interface mewarisi dari interface lainnya. Selain itu, satu class memiliki fleksibilitas untuk mengimplementasikan lebih dari satu interface, yang memungkinkan penggunaan beberapa kontrak atau fungsi dalam satu class

tanpa mengikuti pewarisan class tunggal.

Walaupun interface tidak termasuk dalam hierarki class, interface tetap dapat memiliki hubungan melalui konsep inheritance atau pewarisan. Artinya, sebuah interface bisa mewarisi banyak interface lainnya melalui extends, menggabungkan method-method dari beberapa interface tersebut untuk memperluas fungsionalitasnya.

```
1. public interface PersonInterface {
2.       void doSomething();
3.     }
4.
5.     public interface StudentInterface extends PersonInterface {

6.       void doExtraSomething();
7.     }
```

10.3.4. Implementasi Multiple Interface

Sebuah class dapat mengimplementasikan lebih dari satu interface, memungkinkan penerapan berbagai fungsi atau kontrak dalam satu class. Jika class tersebut juga akan menjadi subclass dan mewarisi dari class lain sambil mengimplementasikan interface, maka urutan penulisan yang benar adalah menuliskan kata kunci extends terlebih dahulu diikuti oleh implements untuk interface.

10.3.5. Kegunaan Interface

Ketika sebuah class mengimplementasikan interface tertentu, class tersebut harus menyediakan implementasi untuk semua method yang didefinisikan oleh interface tersebut. Hal ini berarti setiap class yang mengadopsi interface tersebut memiliki kumpulan method dengan nama dan signatur yang sama, sesuai dengan kontrak yang ditentukan oleh interface.

Dengan memiliki method-method yang identik, class-class tersebut memiliki dasar fungsionalitas yang sama meskipun mungkin diterapkan dengan cara yang berbeda sesuai kebutuhan masing-masing class. Interface memungkinkan berbagai class untuk beroperasi dengan metode serupa, yang mendukung pola desain yang konsisten di seluruh aplikasi atau sistem.

Pada dasarnya, semua class yang mengimplementasikan sebuah interface tertentu membagikan kemampuan atau tanggung jawab yang sama, memungkinkan

mereka untuk digunakan secara interchangeable dalam konteks yang memerlukan fungsionalitas tersebut. Ini memberikan keuntungan fleksibilitas dan skalabilitas, terutama ketika diperlukan perluasan fitur pada aplikasi yang kompleks.

10.3.6. Karakteristik Metode

- Deklarasi Metode Tanpa Implementasi: Interface hanya memiliki deklarasi metode, sedangkan implementasi dilakukan oleh kelas yang mengimplementasikan interface tersebut.
- Multiple Inheritance: Sebuah kelas dapat mengimplementasikan lebih dari satu interface, sehingga interface mendukung pewarisan berganda.
- Kata Kunci implements: Kelas yang ingin menggunakan interface harus menggunakan keyword implements.

10.3.7. Perbedaan Interface dan kelas abstrak

Meskipun dalam interface mempunyai kemiripan secara konsep dengan kelas abstrak, akan tetapi interface berbeda dengan kelas abstrak. Perbedaan interface dengan kelas abstrak dapat kita lihat dalam tabel berikut :

Tabel 1. Perbedaan Interface dengan Kelas Abstrak

Interface	Kelas Abstrak
Semua metode bersifat abstract dan public	Dapat memiliki metode konkret dan abstrak
Mendukung multiple inheritance	Hanya mendukung single inheritance
Tidak memiliki konstruktor	Dapat memiliki konstruktor
Digunakan untuk menyusun kontrak implementasi	Digunakan untuk generalisasi behavior

10.4. PRAKTIKUM

10.4.1. Perisapan

1) Buka Netbeans yang sudah terinstall pada komputer

2) Buat proyek baru dengan nama PraktikumPBO_10.

3) Buat package baru dengan nama praktikum10.

10.4.2. Membuat dan mengimplementasikan Interface

1) Buat sebuah interface bernama OperasiHitung yang mendefinisikan satu metode hitung(int a, int b).

```
// Interface
public interface OperasiHitung {
    int hitung(int a, int b);
}
```

2) Buat kelas Penjumlahan yang mengimplementasikan interface OperasiHitung dan isikan metode hitung() untuk melakukan penjumlahan.

```
// Kelas Penjumlahan
public class Penjumlahan implements OperasiHitung {
    @Override
    public int hitung(int a, int b) {
        return a + b;
    }
}
```

3) Buat kelas Pengurangan yang juga mengimplementasikan interface OperasiHitung dan isikan metode hitung() untuk melakukan pengurangan.

```
// Kelas Pengurangan
public class Pengurangan implements OperasiHitung {
    @Override
    public int hitung(int a, int b) {
        return a - b;
    }
}
```

4) Buat main class dengan kode sebagai berikut :

```
// Kelas Main
public class Main {
    public static void main(String[] args) {
        // Objek Penjumlahan
        OperasiHitung penjumlahan = new Penjumlahan();
        System.out.println("Penjumlahan: " +
penjumlahan.hitung(10, 5));  // Output: 15

        // Objek Pengurangan
        OperasiHitung pengurangan = new Pengurangan();
        System.out.println("Pengurangan: " +
pengurangan.hitung(10, 5));  // Output: 5
    }
}
```

10.5. TUGAS MODUL 9

10.5.1. Soal

Anda diminta untuk membuat program sistem pembayaran untuk sebuah toko yang menjual dua jenis produk: Elektronik dan Makanan. Setiap produk memiliki metode untuk menghitung pajak, di mana Elektronik dikenakan pajak sebesar 10% dan Makanan sebesar 5%.

Instruksi :

- Buat interface Pembayaran dengan metode hitungPajak(double harga).
- Buat kelas Elektronik yang mengimplementasikan interface Pembayaran dan mengisi metode hitungPajak() untuk menghitung pajak sebesar 10% dari harga.
- Buat kelas Makanan yang mengimplementasikan interface Pembayaran dan mengisi metode hitungPajak() untuk menghitung pajak sebesar 5% dari harga.
- Buat kelas Main yang membuat objek dari Elektronik dan Makanan, lalu panggil metode hitungPajak() untuk setiap produk dan tampilkan hasilnya.

Petunjuk Pengerjaan

a) Laporan:

- Buatlah laporan akhir sesuai dengan praktikum yang dilakukan
- Tuliskan laporan ke dalam akun masing-masing web medium
- Publikasikan laporan tersebut
- Kirimkan tautan laporan ke elita.
- **Batas Pengumpulan**: Sebelum Pertemuan Praktik Ke 11.