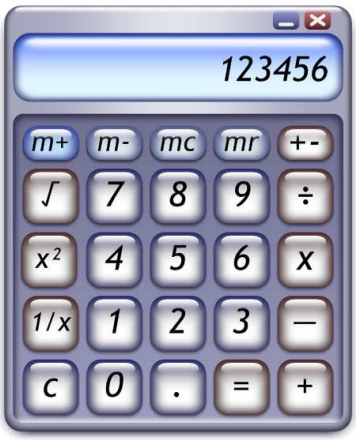
MODUL 6

Interface



Tujuan:

Mahasiswa dapat mengenal dan memahami konsep interface, serta penerapan dalam interface dalam konsep OOP

Materi:

- ✓ Pengantar
- ✓ Interface
- ✓ Implementasi Interface
- ✓ Soal Latihan

Referensi:

- ❖ Fikri, Rijalul. 2005. *Pemrograman Java*. Yogyakarta: Penerbit Andi
- ❖ Hermawan, Benny. 2004. *Menguasai Java 2 & Object Oriented Programming*. Yogyakarta: Penerbit Andi

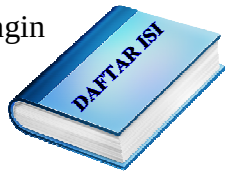
1.1. Pengantar

Ilustrasi 1

Ketika anda diberi tugas menerjemahkan sebuah buku berbahasa Inggris ke dalam bahasa Indonesia, namun ada beberapa kata dalam buku tersebut yang sangat asing bagi anda. Apa yang akan anda lakukan untuk memecahkan masalah tersebut?

Ilustrasi 2

Ketika anda meminjam sebuah buku dan anda ingin mengetahui informasi apa saja yang ada pada buku tersebut, apa yang akan anda lakukan?



Meminjam kamus adalah salah satu alternatif jawaban dalam ilustasi 1. Mengapa? Karena di dalam kamus tersimpan kumpulan kata (atau biasa disebut dengan *vocabulary*) yang anda butuhkan untuk membantu anda dalam mengartikan kata tersebut

Demikian juga untuk ilustrasi 2. Apabila jawaban anda adalah **melihat daftar isi**, berarti anda mengerti kegunaan daftar isi. Karena di dalam daftar isi, terdapat halaman dari tiap bab yang akan membantu anda dalam mempermudah mencari informasi yang anda butuhkan

Kamus dan daftar isi merupakan alat bantu berupa kumpulan informasi sebagai sarana pendukung dalam membantu anda untuk mengolah dan mengembangkan informasi yang anda peroleh. Seperti halnya ketika anda menggunakan Interface pada pemrograman berbasis objek. Interface berisi sekumpulan konstanta/deklarasi method tanpa menyertakan/ menuliskan body methodnya. Method atau variabel yang terdapat pada kelas Inteface dapat

digunakan lebih dari satu kelas dengan cara memanggil kelas interface tersebut. Untuk lebih jelas mengenai konsep interface, simak penjelasannya di bawah ini.

1.2. Interface

Interface adalah kumpulan method yang hanya memuat deklarasi dan struktur method tanpa detail implementasinya. Cara mendeklarasikan *interface* adalah sebagai berikut:

```
interface Nama_Interface
{
    //deklarasi variabel dan/atau method
}
```

Contoh:

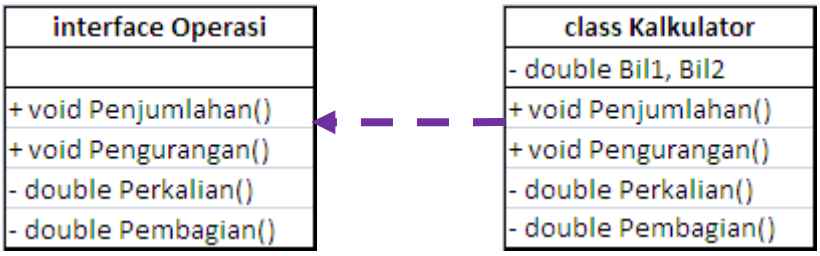
```
interface Operasi
{
    //deklarasi variabel dan/atau method
    public void Penjumlahan();
    public void Pengurangan();
    public double Perkalian();
    public double Pembagian();
}
```

Pada contoh di atas, method yang dideklarasikan pada interface Operasi tidak terdapat statement apapun, baik itu rumus atau hanya sebuah nilai balik di dalamnya. Hal ini dikarenakan interface hanyalah sebuah berisi kumpulan konstanta maupun method tanpa menyertakan/ menuliskan body methodnya.

Perlu diketahui pula, bahwa **an interface is not a class and classes can only implement interfaces** (sebuah interface bukanlah sebuah kelas dan kelas hanya bisa mengimplementasi interface). Sehingga **jangan anda menganggap bahwa interface adalah super class dimana memiliki kelas trurunan.**

1.3. Implementasi Interface

Penggunaan (implementasi) *interface* dalam sebuah kelas dapat anda lihat melalui skema OOP di bawah ini:



Coba anda lihat anak panah yang berwarna ungu tersebut. Anak panah itu merupakan gambaran bahwa **class Kalkulator merupakan implementasi dari interfaces Operasi**, dimana method-method yang terdapat pada interface Operasi harus dideklarasi ulang (overriding method) pada kelas Kalkulator. *Interface* dilambangkan dengan anak panah dengan garis putus-putus, sedangkan *inheritance* dilambangkan dengan anak panah dengan garis lurus (→).

Dalam pemrograman OOP, implementasi interfaces menggunakan keyword **implements**. Berikut adalah cara mengimplementasikan interface ke dalam pemrograman Java:

```
class Nama_Kelas implements Nama_Interface
{
    //isi kelas
}
```

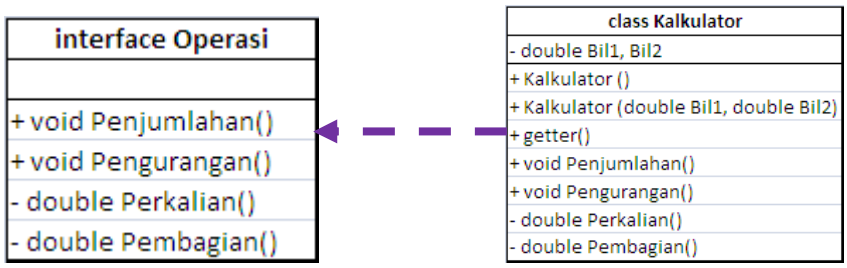
Contoh:

```
class Kalkulator implements Operasi
{
    public void Penjumlahan()
    {
        //isi method Penjumlahan()
    };

    public void Pengurangan()
    {
        //isi method Pengurangan()
    };

    public double Perkalian()
    {
        //isi method Perkalian()
        return 0;
    };

    public double Pembagian()
    {
        //isi method Pembagian()
        return 0;
    };
};
```

Agar anda lebih memahami bagaimana implementasi interface dalam sebuah kelas, simak contoh latihan di bawah ini.

1.4. Soal Latihan

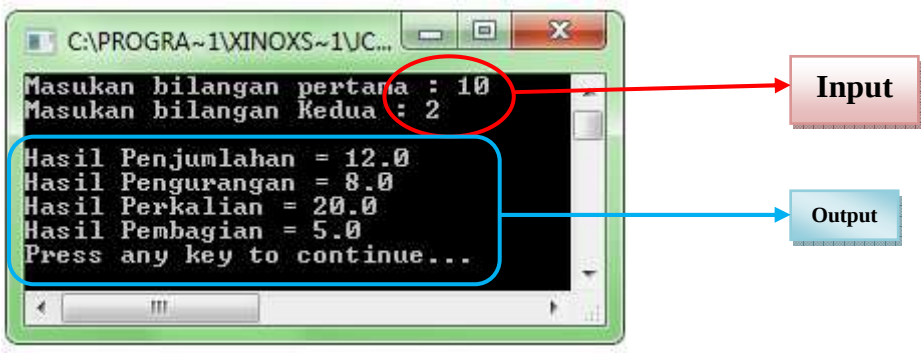
Berdasarkan contoh di atas tentang kalkulator, buatlah program sesuai dengan skema di bawah ini:



Adapun fungsi methodnya adalah sebagai berikut:

- Method **Penjumlahan()** digunakan untuk menjumlahkan dua buah bilangan, yakni Bil1 dan Bil2
- Method **Pengurangan()** digunakan untuk mengurangi dua buah bilangan, yakni Bil1 dan Bil2
- Method **Perkalian()** digunakan untuk mengalikan dua buah bilangan, yakni Bil1 dan Bil2
- Method **Pembagian()** digunakan untuk membagi dua buah bilangan, yakni Bil1 dan Bil2

Tambahkan pula class Utama yang digunakan untuk memanggil class Kalkulator. Ketika class Utama dijalankan, hasilnya akan tampak seperti di bawah ini:



Jawabannya adalah…

Setelah anda membaca soal tersebut dengan baik dan seksama, langkah pertama yang harus anda lakukan adalah mengidentifikasi class atau interface yang harus dibuat terlebih dahulu.

➢ **Langkah 1: interface Operator (ketikkan script berikut)**

a. Membuat kerangka interface Operator

```
1  interface Operasi
2  {
3      //deklarasi method
4  }
```

Setelah anda membuat interface Operasi, simpan file tersebut dengan nama **Operasi.java**. Di dalam interface Operasi, saya juga menyediakan tempat untuk mendeklarasikan method yang akan digunakan pada class Kalkulator.

b. Mendeklarasi method

```
1  interface Operasi
2  {
3      //deklarasi method
4      public void Penjumlahan();
5      public void Pengurangan();
6      public double Perkalian();
7      public double Pembagian();
8  }
```

Dalam interface Operator, anda cukup mendeklarasikan method tanpa isi method (*body method*)

➢ Langkah 2: class Kalkulator (ketikkan script berikut)

a. Membuat kerangka class Kalkulator

```
1  class Kalkulator implements Operasi
2  {
3      //deklarasi variabel
4
5      //constructor
6
7      //getter
8
9      //implementasi method
10
11 };
```

Setelah anda membuat class Kalkulator, simpan file tersebut dengan nama **Kalkulator.java**. Kelas Kalkulator merupakan hasil implementasi dari interface Operasi. Untuk itu, pada *line 1* terdapat keyword "implements". Khusus pada langkah ini, anda jangan merasa bingung apabila anda mendapat 1 error pada saat program di-*compile*. Error yang berisi **Kalkulator is not abstract and does not override abstract method Pembagian() in Operasi** menandakan bahwa method Penjumlahan(), Pengurangan(), Perkalian(), Pembagian() **harus dideklarasikan ulang (overriding method)** ke dalam class Kalkulator. Error ini akan terus ada sampai anda menyelesaikan poin (e).

b. Mendeklarasi variabel yang dibutuhkan

Setelah kita membuat kerangka class, maka diperlukan pendeklarasian variabel yang nantinya digunakan sebagai tempat menyimpan data yang bersifat sementara (*temporary*). Gambar di bawah ini menunjukkan pendeklarasian variabel.


```
class Kalkulator implements Operasi
{
    //deklarasi variabel
    private double Bil1, Bil2;

    //constructor

    //getter

    //implementasi method

};
```

c. Mendeklarasi constructor

Setelah membuat variabel yang dibutuhkan pada class Kalkulator, langkah selanjutnya anda membuat constructor Kalkulator. Constructor ini nantinya akan digunakan dalam class Utama. Gambar di bawah ini menunjukkan deklarasi constructor.

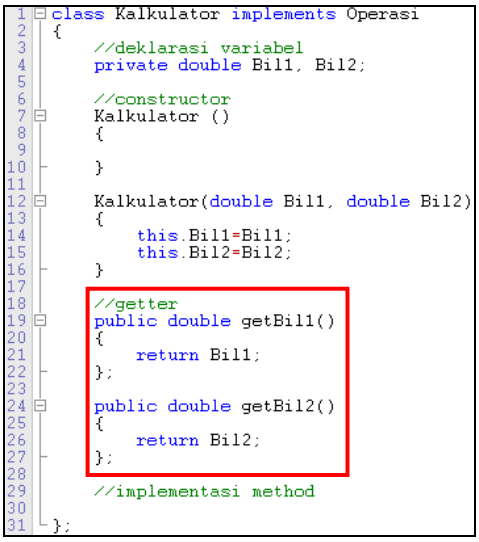
```
class Kalkulator implements Operasi
{
    //deklarasi variabel
    private double Bil1, Bil2;

    //constructor
    Kalkulator ()
    {

    }

    Kalkulator(double Bil1, double Bil2)
    {
        this.Bil1=Bil1;
        this.Bil2=Bil2;
    }

    //getter

    //implementasi method

};
```

Pada gambar di atas, saya menggunakan Overloading Constructor (bagi yang lupa tentang Overloading Constructor, anda dapat melihat kembali pada modul 2 tentang Constructor. Constructor pertama digunakan untuk standard awal dalam melakukan *instance of class*. Sedangkan constructor kedua digunakan untuk mengeset data bilangan pertama dan bilangan kedua yang diperoleh dari kelas Utama.

d. Membuat method getter

Setelah membuat constructor, anda tinggal membuat method getter untuk mengambil nilai dari masing-masing variabel. Hal ini bertujuan untuk mengambil nilai dari variabel pada class Kalkulator yang nantinya akan kita

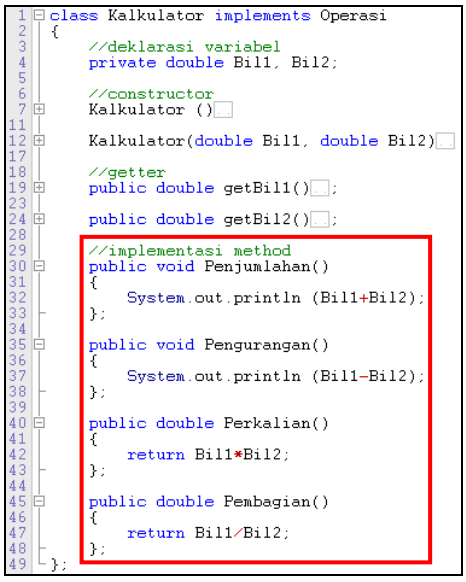
kembalikan ke dalam class Utama. Perlu diketahui pula, bahwa dalam pembuatan variabel pada parameter di method getter, harus sesuai dengan tipe data pada variabel yang dideklarasikan di kelas. Gambar di bawah ini menunjukkan deklarasi getter.



```
class Kalkulator implements Operasi
{
    //deklarasi variabel
    private double Bil1, Bil2;

    //constructor
    Kalkulator ()
    {

    }

    Kalkulator(double Bil1, double Bil2)
    {
        this.Bil1=Bil1;
        this.Bil2=Bil2;
    }

    //getter
    public double getBil1()
    {
        return Bil1;
    };

    public double getBil2()
    {
        return Bil2;
    };

    //implementasi method

};
```

e. Implementasi method

Setelah membuat constructor, anda **wajib melakukan deklarasi ulang (overriding method)** ke dalam class Kalkulator seperti pada script di bawah ini.



```
class Kalkulator implements Operasi
{
    //deklarasi variabel
    private double Bil1, Bil2;

    //constructor
    Kalkulator ()...

    Kalkulator(double Bil1, double Bil2)...

    //getter
    public double getBil1()...;

    public double getBil2()...;

    //implementasi method
    public void Penjumlahan()
    {
        System.out.println (Bil1+Bil2);
    };

    public void Pengurangan()
    {
        System.out.println (Bil1-Bil2);
    };

    public double Perkalian()
    {
        return Bil1*Bil2;
    };

    public double Pembagian()
    {
        return Bil1/Bil2;
    };
};
```


Perlu diketahui pula, bahwa method Penjumlahan() dan Pengurangan merupakan sub program berjenis prosedur. Sedangkan method Perkalian() dan Pembagian() merupakan sub program berjenis fungsi. Untuk itu, ada perbedaan cara memanggil method dalam kelas Utama.

➢ Langkah 3: class Utama (ketikkan script berikut)

Berikut adalah script kelas utama kalkulator:

```
1  import java.io.*;
2  class Utama
3  {
4      public static void main (String [] args) throws Exception
5      {
6          BufferedReader br = new BufferedReader (new InputStreamReader (System.in));
7
8          //instance of class
9          Kalkulator k = new Kalkulator();
10
11         //input
12         System.out.print("Masukan bilangan pertama : ");
13         double a= Double.parseDouble(br.readLine());
14         System.out.print("Masukan bilangan Kedua : ");
15         double b= Double.parseDouble(br.readLine());
16
17         k= new Kalkulator (a,b);
18
19         System.out.println();
20
21         //output
22         System.out.print ("Hasil Penjumlahan = ");
23         k.Penjumlahan();
24
25         System.out.print ("Hasil Pengurangan = ");
26         k.Pengurangan();
27
28         System.out.println("Hasil Perkalian = "+k.Perkalian());
29
30         System.out.println("Hasil Pembagian = "+k.Pembagian());
31     }
32 }
```

Keterangan:

Line 9 = deklarasi instance of class, dimana variabel tersebut bertipe kelas Kalkulator, yang merupakan turunan dari kelas Operasi

Line 12-15 = inputan user, dimana bilangan 1 ditampung ke dalam variabel a dengan tipe data double. Sedangkan bilangan 2 ditampung ke dalam variabel b dengan tipe data double

Line 17 = mentransfer data pada variable a dan b ke dalam constructor Kalkulator

Line 19 = sebagai jarak antara isi input dan output ketika program dijalankan

Line 22-26 = cara memanggil method Penjumlahan dan Pengurangan yang merupakan sub program bertipe void. Karena di dalam isi void terdapat **System.out.println**, maka pemanggilan method dilakukan di luar kelas

Line 28-30 = cara memanggil method Penjumlahan dan Pengurangan yang merupakan sub program bertipe void. Karena di dalam isi function tidak terdapat **System.out.println** dan hanya mengembalikan return (nilai balik), maka pada class Utama, pemanggilan method dilakukan di dalam System.out.println().

