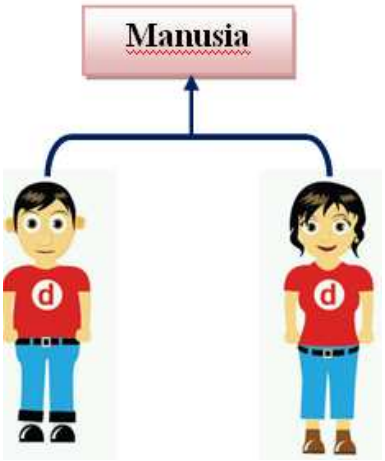
MODUL 5

Abstract Class



Tujuan:

Mahasiswa dapat mengenal dan memahami konsep abstract class, abstract method dan keyword "*final*" serta penerapannya dalam konsep OOP

Materi:

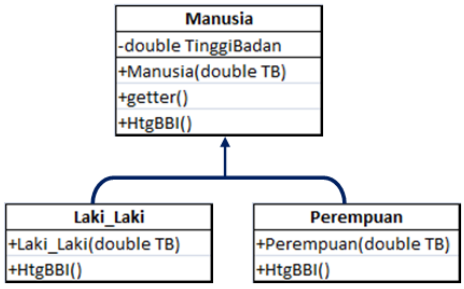
- ✓ Pengantar
- ✓ Abstract Class
- ✓ Keyword "*final*"
- ✓ Soal Latihan

Referensi:

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

1.1. Pengantar

Anda pasti masih ingat contoh program yang menghitung berat badan ideal berdasarkan tinggi badan (bagi yang tidak ingat, bisa membuka kembali soal latihan pada modul 4). Di dalam contoh tersebut, terdapat class Manusia, class Laki_Laki, dan class Perempuan seperti contoh bagan di bawah ini.



Dalam abstract class, class tertinggi merupakan kelas Manusia. Apabila dalam kelas abstract terdapat **abstract method**, maka method tersebut hanya mendeskripsi method tanpa isi methodnya. Untuk lebih jelas, simak konsep abstract method di bawah ini.

1.2. Abstract Class

Abstract Class merupakan kelas yang berada pada posisi tertinggi dalam sebuah hierarki kelas. Sesuai dengan namanya, abstract class dapat didefinisikan pada class itu sendiri. Berikut adalah cara mendeklarasikan abstract class:

```
1  abstract class Nama_Kelas
2  {
3      //isi abstract class
4  }
```

Contoh:

```
abstract class Manusia
{
    //isi abstract class
}
```

Di dalam abstract class dapat juga diberi **abstract method** (optional). Penggunaan abstract method tidak diperlukan statement dalam method tersebut. Berikut adalah cara mendeklarasikan abstract method pada abstract class:

```
abstract class Nama_Kelas
{
    //untuk sub program berjenis prosedur
    [access_modifier] abstract void [nama_method]();

    //untuk sub program berjenis function
    [access_modifier] abstract [tipe_data] [nama_method]();
}
```

Contoh:

```
abstract class Manusia
{
    //untuk sub program berjenis prosedur
    public abstract void cetak();

    //untuk sub program berjenis function
    public abstract double HtgBBI();
}
```

Catatan:

- ✓ Apabila dalam abstract class terdapat abstract method dan kelas tersebut diturunkan ke kelas turunannya, maka method tersebut harus dideklarasikan ulang (overriding method) dengan diberi statement pada isi methodnya. Untuk lebih jelas penggunaan overriding method, dapat dilihat pada soal latihan
- ✓ Apabila class tersebut merupakan abstract class, maka class tersebut bisa terdapat abstract method atau tidak (optional). Sedangkan apabila kelas tersebut, terdapat abstract method, maka kelas tersebut **wajib** berbentuk abstract class

1.3. Keyword "*final*"

Keyword "*final*" digunakan untuk mencegah suatu class diturunkan atau suatu method dilakukan pendeklarasian ulang (overriding method). Berikut adalah cara mendeklarasikan *final* dalam class:


```
final class Nama_Kelas
{
    //isi class
}
```

Contoh:

```
final class Manusia
{
    //isi class
}
```

Sedangkan cara mendeklarasikan “final” pada method adalah sebagai berikut:

```
final class Nama_Kelas
{
    //sub program berjenis prosedur
    [access_modifier] final void [nama_method]()
    {
        //isi method
    }

    //sub program berjenis function
    [access_modifier] final [tipe_data] [nama_method]()
    {
        //isi method
        return [isi_nilai];
    }
}
```

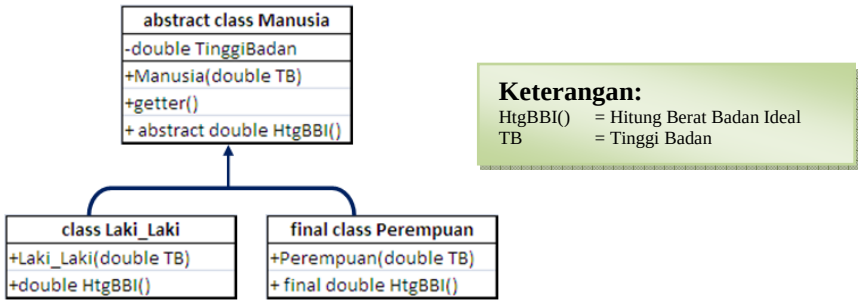
Contoh

```
final class Manusia
{
    //untuk sub program berjenis prosedur
    public final void cetak()
    {
        //isi method
    }

    //untuk sub program berjenis function
    public final double HtgBBI()
    {
        //isi method
        return 0;
    }
}
```

Untuk lebih detail dalam penggunaan keyword “*final*”, akan dibahas lebih lanjut melalui soal latihan

1.4. Soal Latihan

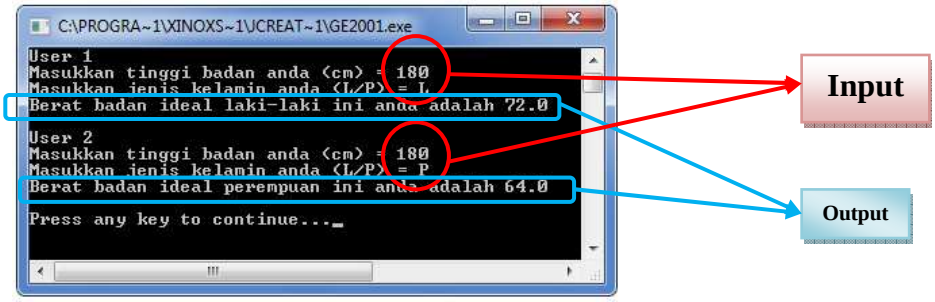
Seperti halnya dengan soal latihan pada modul 4 tentang “Polymorphism”, soal latihan pada modul 5 tidak berbeda jauh. Perhatikan gambar di bawah ini!



Buatlah class untuk menghitung Berat Badan Ideal sesuai dengan rancangan gambar di atas! Rumus hitung berat badan ideal adalah sebagai berikut:

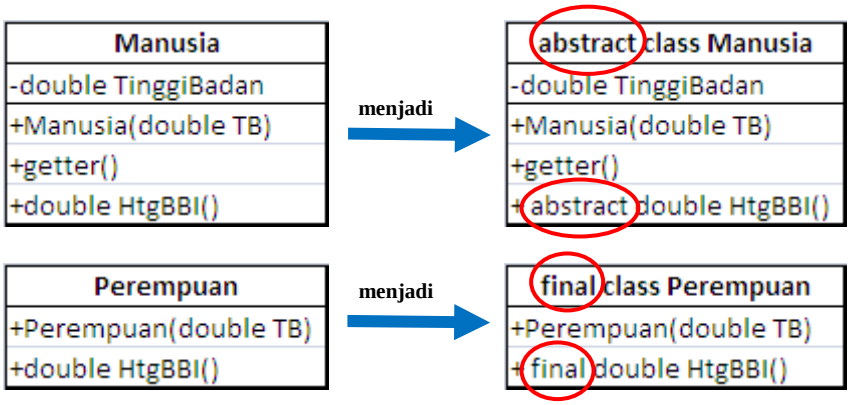
- ❖ Laki-Laki = (tinggi badan(cm)-100) kg x 90%
- ❖ Untuk Perempuan = (tinggi badan(cm)-100) kg x 80%

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



Jawabannya adalah…

Ketika anda perhatikan baik-baik bagan soal latihan modul 5 dengan soal latihan modul 4, maka anda akan menemukan perbedaan sebagai berikut:



➢ Langkah 1: class Manusia (ketikkan script berikut)

a. Membuat kerangka class Manusia

```
abstract class Manusia
{
    //deklarasi variabel

    //constructor

    //getter

    //Method HtgBBI

}
```

Setelah anda membuat class Manusia, simpan file tersebut dengan nama **Manusia.java**. Di dalam class Manusia, saya juga menyediakan tempat untuk mendeklarasikan variabel, constructor, getter dan method HtgBBI() untuk menghitung berat badan ideal.

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.

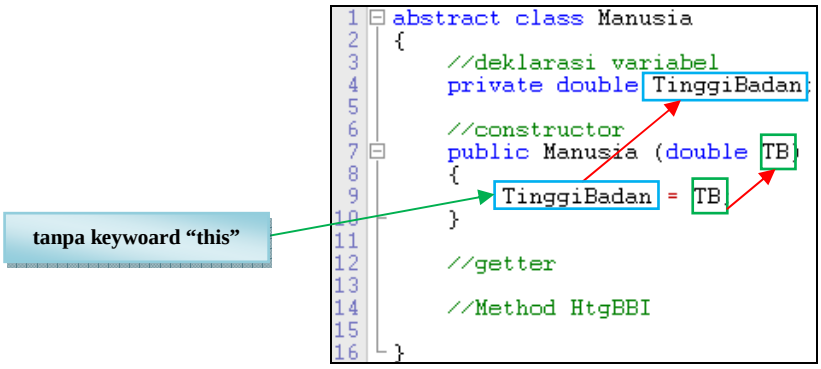
```
abstract class Manusia
{
    //deklarasi variabel
    private double TinggiBadan;

    //constructor

    //getter

    //Method HtgBBI

}
```

c. Mendeklarasi constructor

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

```
1  abstract class Manusia
2  {
3      //deklarasi variabel
4      private double TinggiBadan;
5
6      //constructor
7      public Manusia (double TB)
8      {
9          TinggiBadan = TB;
10     }
11
12     //getter
13
14     //Method HtgBBI
15
16 }
```

Coba perhatikan kembali script yang telah anda buat seperti gambar di atas. Karena terdapat perbedaan deklarasi nama variabel pada class Manusia dan deklarasi variabel pada constructor Manusia, maka keyword ***this*** boleh digunakan atau tidak (*optional*). Untuk lebih jelas mengenai keyword ***this***, anda dapat melihat modul 1.



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 Manusia yang nantinya akan kita gunakan ke dalam class Laki_Laki maupun class Perempuan. 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.


```
abstract class Manusia
{
    //deklarasi variabel
    private double TinggiBadan;

    //constructor
    public Manusia (double TB)
    {
        TinggiBadan = TB;
    }

    //getter
    public double getTB()
    {
        return TinggiBadan;
    }

    //Method HtgBBI

}
```

e. Membuat method HtgBBI()

Setelah membuat method getter(), anda tinggal membuat method HtgBBI() yang bertugas untuk menghitung berat badan ideal berdasarkan tinggi badan. Karena method HtgBBI() merupakan abstract method, maka method tersebut tidak terdapat isi method. Gambar di bawah ini menunjukkan deklarasi method HtgBBI().

```
abstract class Manusia
{
    //deklarasi variabel
    private double TinggiBadan;

    //constructor
    public Manusia (double TB)
    {
        TinggiBadan = TB;
    }

    //getter
    public double getTB()
    {
        return TinggiBadan;
    }

    //Method HtgBBI
    public abstract double HtgBBI();
}
```

➢ **Langkah 2: class Laki_Laki (ketikkan script berikut)**

Karena class Laki_Laki pada soal latihan 4 = soal latihan 5, maka script ini tidak mengalai perubahan. Adapun scriptnya adalah sebagai berikut:

```
class Laki_Laki extends Manusia
{
    //constructor
    public Laki_Laki (double TB)
    {
        super(TB);
    }

    //Method HtgBBI() merupakan method overidding dari superclass-nya
    public double HtgBBI()
    {
        return (super.getTB()-100)*0.9;
    }
}
```

Setelah anda membuat class Laki_Laki, simpan file tersebut dengan nama **Laki_Laki.java**. Di dalam class Laki_Laki, terdapat penggunaan **extends** yang menunjukkan bahwa class Laki_Laki merupakan turunan dari class Manusia.

Coba perhatikan kembali script yang telah anda buat seperti gambar di atas. Pada constructor Laki_Laki, terdapat keyword "***super***". Keyword ini akan memanggil constructor Manusia (sesuai isi parameter) yang merupakan class induk. Sedangkan pada method HtgBBI() dilakukan pendeklarasian kembali (overriding method) sesuai dengan kelas induknya, dimana method HtgBBI() diberi rumus untuk menghitung berat badan ideal laki-laki.

➢ Langkah 3: class Perempuan (ketikkan script berikut)

```
final class Perempuan extends Manusia
{
    //constructor
    public Perempuan (double TB)
    {
        super(TB);
    }

    //Method HtgBBI() merupakan method overidding dari superclass-nya
    public final double HtgBBI()
    {
        return (super.getTB()-100)*0.8;
    }
}
```

Setelah anda membuat class Perempuan, simpan file tersebut dengan nama **Perempuan.java**. Di dalam class Perempuan, terdapat penggunaan **extends** yang menunjukkan bahwa class Perempuan merupakan turunan dari class Manusia.

Seperti halnya dengan class Laki_Laki, class Perempuan memiliki constructor dan method HtgBBI() yang sama. Perbedaannya terletak pada nama class, nama constructor, dan isi rumus method HtgBBI().

Keyword "final" pada *line 1* digunakan untuk mencegah pembuatan kelas baru dari kelas turunan perempuan. Sedangkan keyword "final" pada *line 10* digunakan untuk mencegah pendeklarasian ulang (overriding method) pada kelas turunannya.

➢ Langkah 4: class Utama (ketikkan script berikut)

Karena class Utama pada soal latihan 4 = soal latihan 5, maka script ini tidak mengalai perubahan. Adapun scriptnya adalah sebagai berikut:

```
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          //instance of class
8          Manusia[] m = new Manusia[2];
9
10         //deklarasi variabel
11         int x=0;
12
13         do
14         {
15             //input
16             System.out.println("User "+(x+1));
17             System.out.print("Masukkan tinggi badan anda (cm) = ");
18             double t = Double.parseDouble(br.readLine());
19             System.out.print("Masukkan jenis kelamin anda (L/P) = ");
20             String jk=br.readLine();
21
22             //proses + output
23             if (jk.toUpperCase().equals("L"))
24             {
25                 m[x]=new Laki_Laki(t);
26                 System.out.println("Berat badan ideal laki-laki ini anda adalah "+m[x].HtgBBI());
27                 System.out.println();
28             }
29             else
30             {
31                 m[x]=new Perempuan(t);
32                 System.out.println("Berat badan ideal perempuan ini anda adalah "+m[x].HtgBBI());
33                 System.out.println();
34             }
35             x++;
36         }while (x<2);
37
38     }
39 }
```

Untuk penjelasan script pada class Utama, dapat dilihat pada soal latihan modul 4. Perlu diketahui pula, bahwa ***instance of class*** **pada class Utama <u>tidak wajib</u> menggunakan array of Class (konsep Polymorphism), tapi anda bisa juga membuat objek berdasarkan kelas turunannya.**