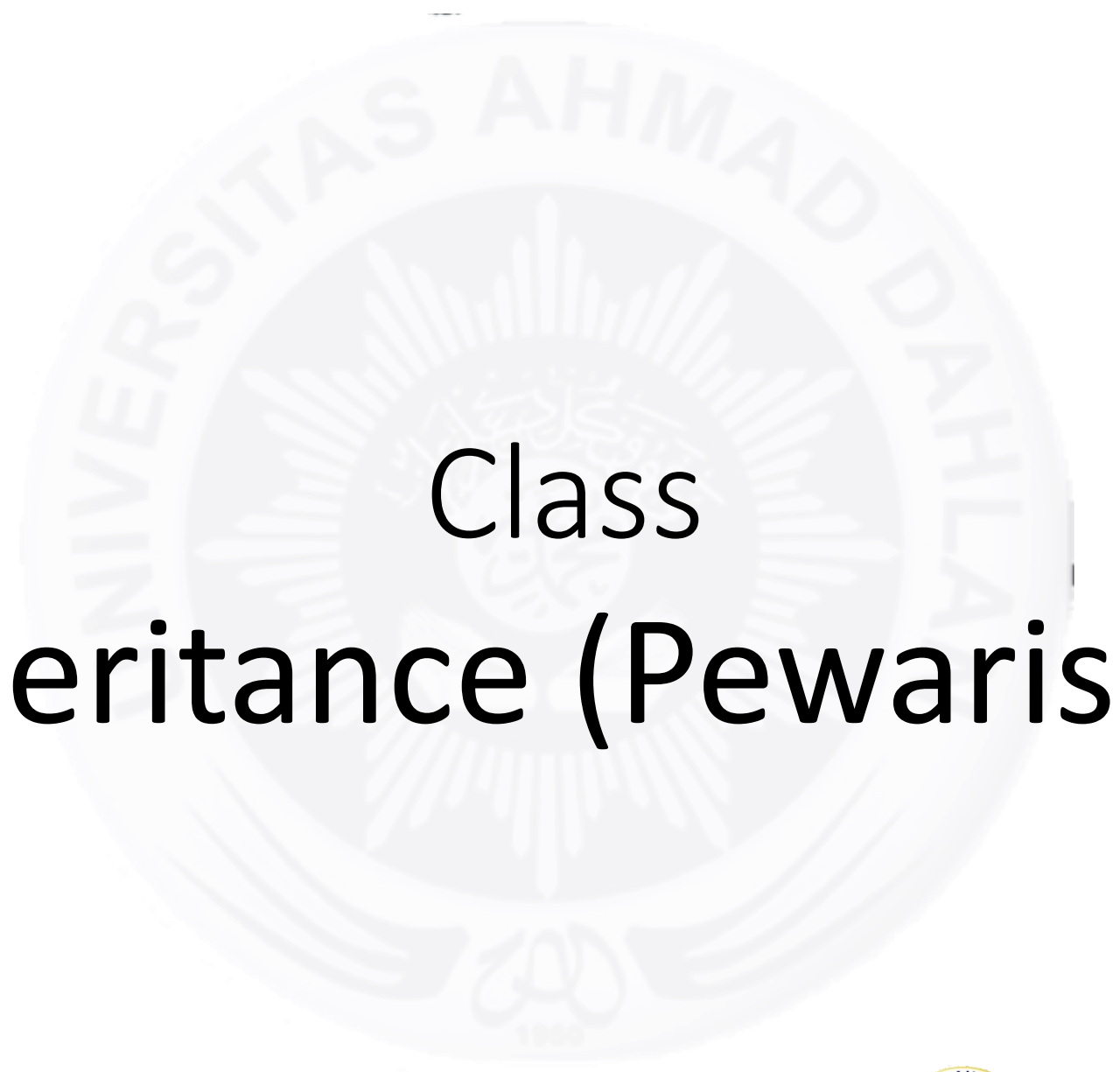


Temu 05

Inheritance (Pewarisan)

Class

Pelengkap materi *PBO2019-05-Inheritance*



Class

Inheritance (Pewarisan)

Pengertian Inheritance (Pewarisan Class)

- Inheritance adalah konsep OOP dimana sebuah class dapat menurunkan property dan method yang dimilikinya kepada class lain.
- Konsep inheritance dipakai untuk memanfaatkan fitur code reuse, yakni menghindari terjadinya duplikasi kode program.
- Dalam bahasa Indonesia, inheritance ini disebut juga sebagai pewarisan atau penurunan.
- Konsep inheritance membuat sebuah struktur atau hierarki class dalam kode program.
- Class yang akan diturunkan bisa disebut sebagai class induk, parent class, super class, atau base class.
- Sedangkan class yang menerima penurunan bisa disebut sebagai class anak, child class, sub class, derived class atau heir class.

Cara Menurunkan Class di Java

- Cara menurunkan sebuah class:
 - Tentukan class mana yang akan diturunkan (harus sudah ada dan bisa diakses oleh program yang akan diisi anak/child class ybs (ruang lingkup/scope sesuai)
 - Tulis nama child class,
 - Tambahkan perintah **extends** atau **implements**
 - Tulis nama **parent class**.
 - Lengkapi dengan blok koding tanda kurung **()** dan tanda kurung kurawal **{}**
 - Isi dari koding class (tambahan property atau method baru)

Berikut contoh kode program yang di maksud:

Contoh Inheritance class

```
1 class Komputer {
2     // kode untuk class Komputer
3 };
4
5 class Laptop extends Komputer {
6     // kode untuk class Laptop
7 };
8
9 class BelajarJava {
10     public static void main(String args[]){
11         Komputer komputerAndi = new Komputer();
12         Laptop laptopAndi = new Laptop();
13     }
14 }
```

Keterangan:

- Parent class: **Komputer**
- Child class: **Laptop** (mewarisi semua milik class Komputer)
- Metode pewarisan: **extends**
- Baris 6: bisa diisi dengan property & method yg hanya dimiliki oleh **class Laptop**.
- Baris 11 & 12: pembuatan objek dari class yg berbeda.

Perlu Perhatian Khusus

- Dalam konsep OOP, hubungan antara **parent class** dan **child class** seharusnya memiliki makna tertentu.
 - Biasanya hubungan yang terjadi adalah “**is-a**”, atau “**adalah sebuah**”.
 - Maksudnya, jika **class Laptop** adalah turunan dari **class Komputer**, maka seharusnya **Laptop** adalah sebuah **Komputer**.
 - Dalam hal ini **Laptop** adalah bentuk yang lebih khusus dari **Komputer**.
 - Selain **Laptop**, **class Komputer** bisa saja memiliki **child class** lain misal: **KomputerDesktop**, **MiniComputer**, **KomputerServer**, dll sesuai kebutuhan.
- Contoh pewarisan lain, misal:
 - antara **class Binatang** dengan **Kucing**, **Ayam**, **Ular**, **Sapi**, dll. Keempatnya adalah termasuk **Binatang**.
 - Atau **class Mobil**, **Motor**, **Sepeda**, **Andong**, **Kereta**, dll, yang memiliki **parent class Kendaraan**.

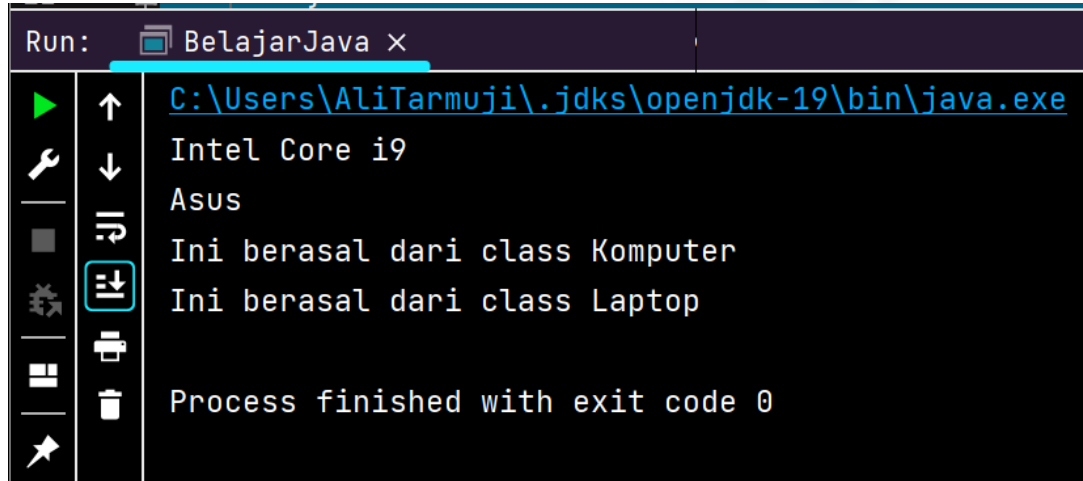
Cara Mengakses Data Parent Class

```
1  class Komputer {
2      String processor = "Intel Core i9";
3      String cekKomputer() {
4          return "Ini berasal dari class Komputer";
5      }
6  };
7
8  class Laptop extends Komputer {
9      String merk = "Asus";
10     String cekLaptop() {
11         return "Ini berasal dari class Laptop";
12     }
13 };
14
15 class BelajarJava {
16     public static void main(String args[]){
17         Laptop laptopAndi = new Laptop();
18
19         System.out.println(laptopAndi.processor);
20         System.out.println(laptopAndi.merk);
21         System.out.println(laptopAndi.cekKomputer());
22         System.out.println(laptopAndi.cekLaptop());
23     }
24 }
```

Keterangan:

- Kedua kelas sudah diberi property & method
- Baris 2-5: dimiliki juga (diwariskan) ke class **Laptop**
- Baris 9-13: hanya dimiliki oleh class **Laptop**
- Baris 15-24: cara pengaksesan child class

Hasil Luaran Program Contoh



The screenshot shows a Java IDE console window titled 'Run: BelajarJava x'. The console output is as follows:

```
C:\Users\AliTarmuji\.jdk\openjdk-19\bin\java.exe
Intel Core i9
Asus
Ini berasal dari class Komputer
Ini berasal dari class Laptop
Process finished with exit code 0
```

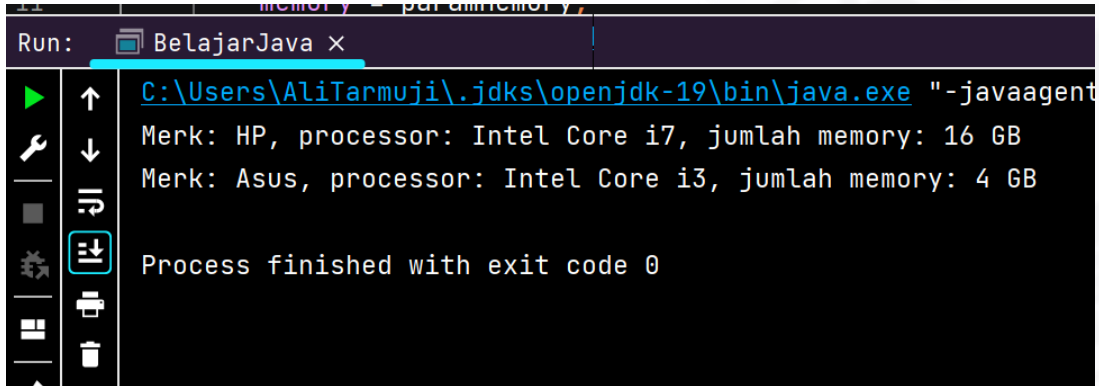
Keterangan:

- Class Komputer, tidak dipakai di program luaran, tetapi **class Laptop** yang mewakili kepemilikan **class Komputer**

Mengisi Property Menggunakan Constructor

```
1  class Komputer {
2      String processor;
3      String merk;
4      String memory;
5  };
6
7  class Laptop extends Komputer {
8      Laptop(String paramProcessor, String paramMerk, String paramMemory) {
9          processor = paramProcessor;
10         merk = paramMerk;
11         memory = paramMemory;
12     }
13
14     String lihatSpec() {
15         return "Merk: " + merk + ", processor: " + processor +
16             ", jumlah memory: " + memory;
17     }
18 };
19
20 class BelajarJava {
21     public static void main(String args[]){
22         Laptop laptopAndi = new Laptop("Intel Core i7","HP","16GB");
23         Laptop laptopRudi = new Laptop("Intel Core i3","Asus","8GB");
24
25         System.out.println(laptopAndi.lihatSpec());
26         System.out.println(laptopRudi.lihatSpec());
27     }
28 }
```

Luaran Program



The screenshot shows a Java IDE console window titled 'Run: BelajarJava x'. The command executed is `C:\Users\AliTarmuji\.jdk\openjdk-19\bin\java.exe "-javaagent`. The output displays two lines of data: `Merk: HP, processor: Intel Core i7, jumlah memory: 16 GB` and `Merk: Asus, processor: Intel Core i3, jumlah memory: 4 GB`. The console also shows `Process finished with exit code 0`. The IDE interface includes a toolbar with icons for running, debugging, and other development tasks.

Keterangan:

- Pendefinisian property **processor**, **merk** dan **memory** tetap dilakukan dari **class Komputer**.
- Proses pengisian ketiga data dilakukan oleh **constructor class Laptop** seperti di (cek baris 8-12)
- **Constructor class Laptop** butuh 3 buah parameter, yakni **paramProcessor**, **paramMerk** dan **paramMemory**.
- Ketiga parameter ini harus diisi pada saat proses instansiasi class Laptop (cek baris 22 dan 23)
- Hasilnya, pada saat method **lihatSpec()** diakses dari **object laptopAndi** dan **laptopRudi**, akan tampil nilai yang berbeda-beda.

Tips Membaca Koding Java yang Kompleks

- Koding Java yang kompleks, akan terdiri dari program utama (main) dan beberapa sub program atau dalam bentuk class dan juga bisa dalam bentuk library (class/program/objek di luar koding ybs).
- Ketika mempelajari koding yang agak panjang dan melibatkan banyak object, **mulailah membaca kode program dari main()**.
- Jika terdapat proses instansiasi object dengan argument, baru masuk ke dalam pendefinisian class tersebut
- ikuti alur pengiriman argumen hingga mengisi property dari sebuah class.
- Apabila kita mulai membaca dari pendefinisian class terlebih dahulu, biasanya memang lebih rumit.

Referensi

- [Tutorial OOP Java: Cara Membuat Pewarisan Class \(Inheritance\)](http://duniailkom.com)
(duniailkom.com)