



PEWARISAN (INHERITANCE)

Pertemuan ke-8

Sub-CPMK

- *Mahasiswa mampu menyelesaikan masalah dengan menggunakan salah satu pilar OOP yaitu Pewarisan dalam konsep Pemrograman Berorientasi Objek (PBO). (C4, A4).*

Materi

1. Konsep Pewarisan
2. Keuntungan Konsep Pewarisan
3. Aksesabilitas Nilai Pada Pewarisan
4. Jenis-Jenis Pewarisan
5. Keyword Super
6. Overriding & Overloading



1. Konsep Pewarisan

- Salah satu keunggulan konsep pemrograman berorientasi objek adalah pewarisan.
- Perwarisan adalah kemampuan untuk membuat kelas turunan yang mewarisi atribut, data, method, behavior dari kelas induk (base class / super class) ke kelas turunannya.
- Kelas turunan disebut dengan kelas anak (subclass atau derived class).

Lanj...

- Proses pewarsian dari kelas induk ke kelas anak disebut dengan deriving.
- Pada dasarnya pewarisan adalah membuat kelas baru yang masih memiliki sifat dari kelas induknya.
- Untuk melakukan pewarsian pada bahasa pemrograman Java menggunakan kata kunci (keyword) **extends**, perhatikan sintaks berikut:

Lanj...

Sintaks: perwarisan

```
[access modifier] class NamaObjek extends NamaKelasInduk  
{  
    ...  
    ...  
}
```

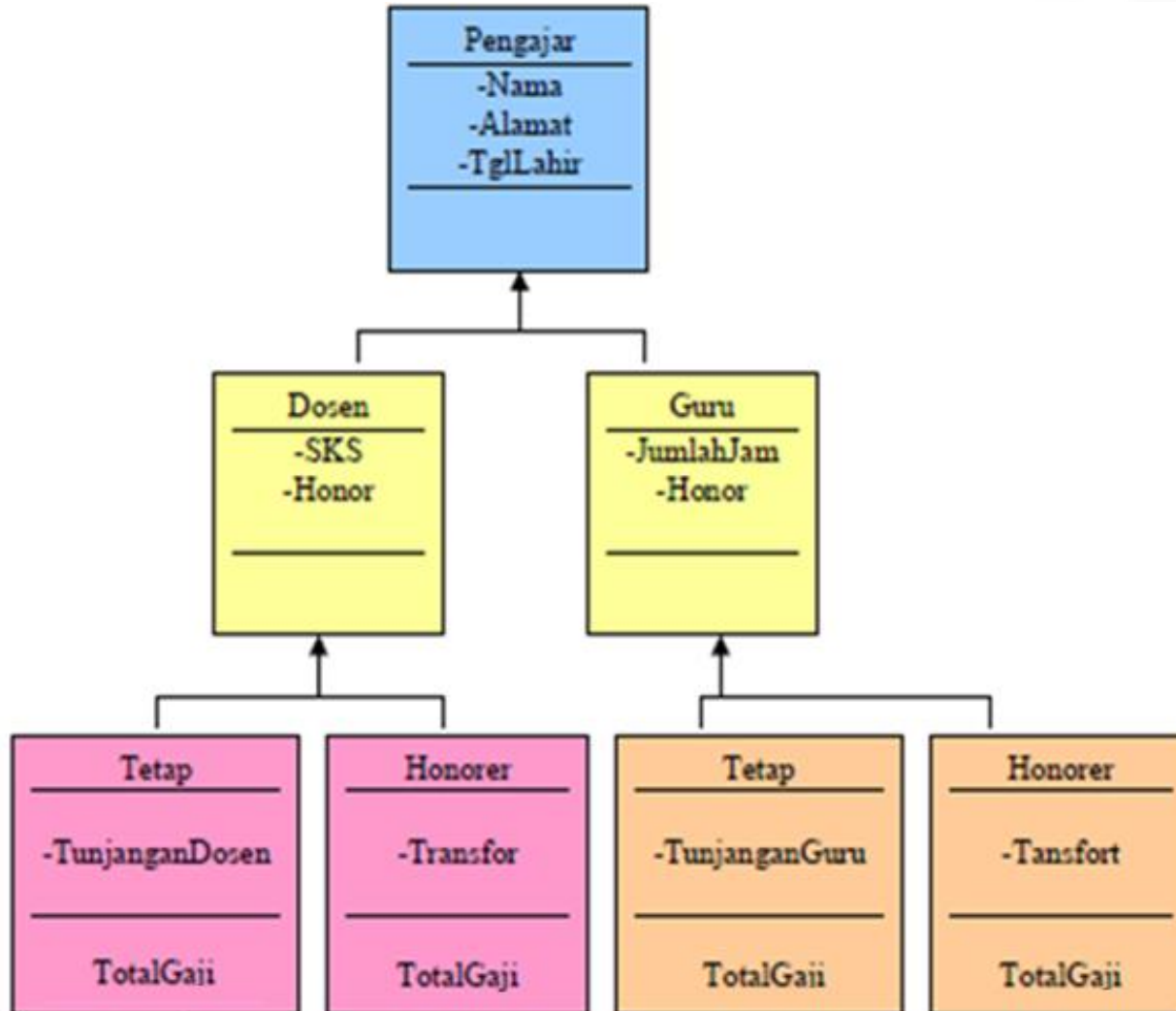
Contoh:

Contoh: Pewarisan

```
public class clsDokterSpesialis extends clsDokter  
{  
    ...  
    ...  
}
```

Lanj...

Ilustrasi pewarisan





2. Keuntungan Konsep Pewarisan

- Keuntungan penggunaan perwarisan adalah:
 - Memberikan ciri khas pada masing-masing subclass.
 - Superclass mewariskan atribut dan method-nya ke subclass sehingga menerapkan reuse (penggunaan kembali).

Lanj...

- Pada pewarisan juga dikenal adanya overriding, overriding pada konsep pewarisan seperti:
 - Method dengan nama sama dan tipenya, tetapi di kelas berbeda. Namun masih dalam satu hubungan keturunan.
 - Jika ada method di kelas parent yang sudah didefinisikan dan didefinisikan ulang, maka method pada kelas anak akan menimpa method parent, kecuali dibuat final.



3. Aksesabilitas Nilai Pada Pewarisan

- Aksesabilitas nilai pada konsep pewarisan adalah: **private, protected, public**.
- Kelas turunan dapat mengakses nilai yang memiliki jangkauan nilai **protected** atau **public**.
- Atribut pada kelas induk bila dideklarasikan dengan jangkauan nilai **private** tidak diturunkan dari kelas induk ke kelas anak.

Lanj...

Jangkauan Nilai	Keterangan
Private	Tidak dapat diakses oleh kelas lain, untuk mengaksesnya melalui fungsi anggota.
Protected	Dapat diakses oleh kelas turunan.
Public	Dapat diakses oleh sembarang kelas.



4. Jenis-Jenis Pada Pewarisan

- Ada beberapa jenis pewarisan yang dikenal, seperti:
 1. Single inheritance.
 2. Multilevel inheritance.
 3. Hierarchical inheritance.
 4. Multiple inheritance.
 5. Hybrid inheritance.
 6. Multi-path inheritance.

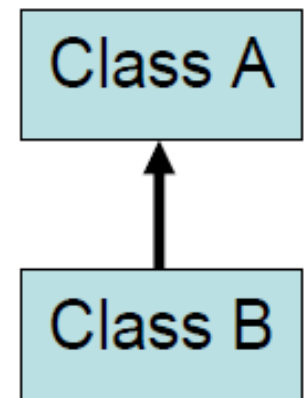
Lanj...

- **Catatan:**

Tidak semua jenis pewarisan dapat diimplementasikan ke dalam bahasa pemrograman, ada beberapa jenis pewarisan secara konsep program tidak mungkin untuk diterapkan.

4.1 Single Inheritance

- Jenis pewarisan single inheritance adalah: sebuah kelas turunan implementasi hanya dari satu kelas induk.
- Contoh: jika kelas B turunan dari kelas A, maka kelas B akan mewarisi semua yang ada pada kelas A.
- Perhatikan ilustrasi pada gambar.



4.1 Single Inheritance (Lanj..)

Nama class induk: clsDokter

```
1.  package dokter;
2.
3.  public class clsDokter
4.  {
5.      String IdDokter;
6.      String Nama;
7.      int Gaji;
8.
9.      public double Tunjangan (int mGaji)
10.     {
11.         return mGaji/100 * 10;
12.     }
13. }
```

4.1 Single Inheritance (Lanj..)

Nama class turunan: clsDokterSpesialis

```
1.  package dokter;
2.  public class clsDokterSpesialis extends clsDokter
3.  {
4.      int TunjanganSpesialis;
5.      public double UangMakan(int mGaji)
6.      {
7.          return mGaji/100 * 10;
8.      }
9.      public double TotalGaji(int mGaji, double mTunjangan,
10.         double mTunSpesialis, double mUangMakan)
11.      {
12.          return mGaji + mTunjangan + mTunSpesialis +
13.             mUangMakan;
14.      }
15. }
```

4.1 Single Inheritance (Lanj..)

Main program: Dokter

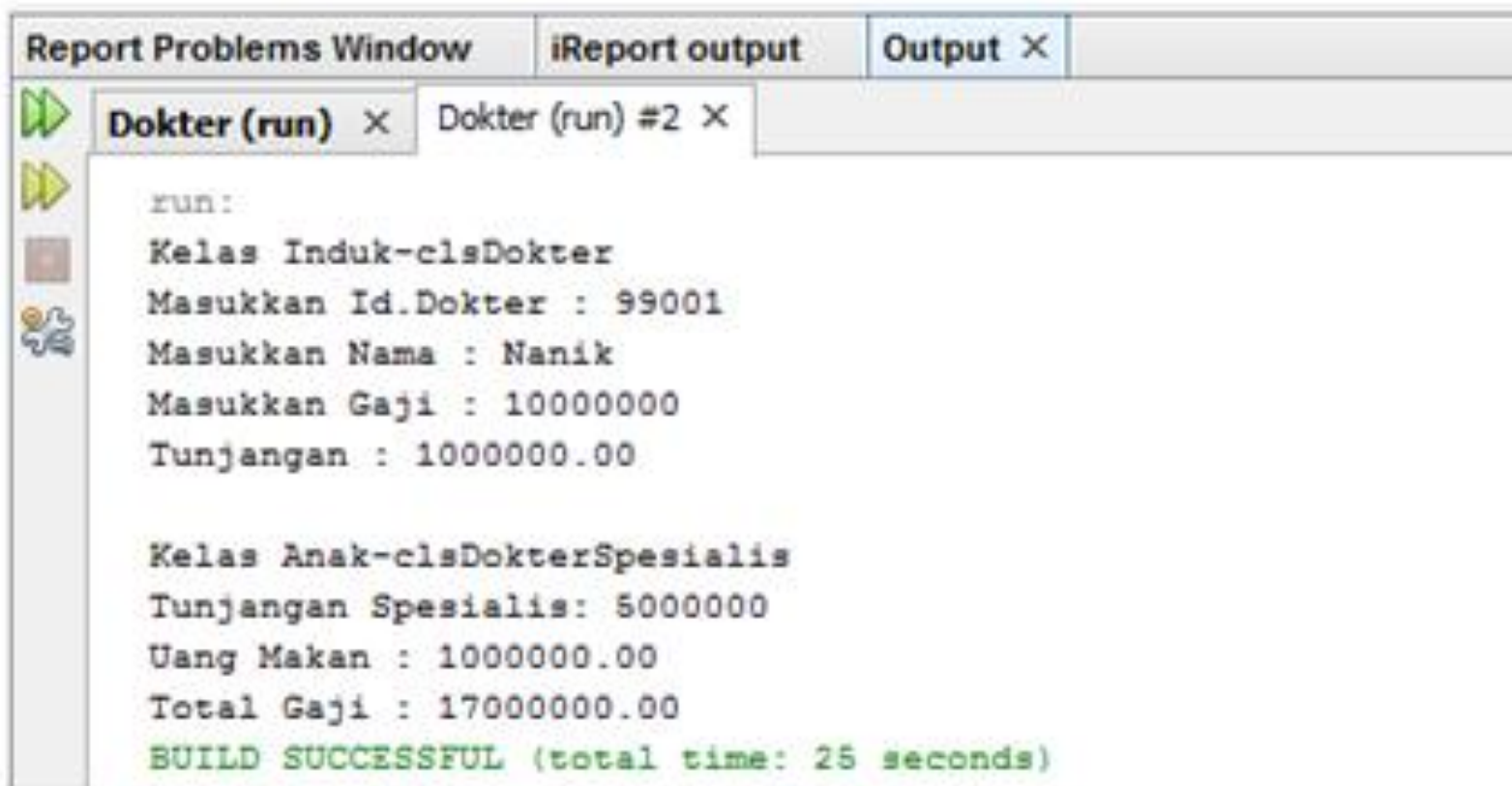
```
1.  package dokter;
2.  import java.util.Scanner;
3.
4.  public class Dokter {
5.
6.      public static void main(String[] args)
7.      {
8.          double Tunjangan;
9.          double UangMakan;
10.         Scanner input = new Scanner (System.in);
11.         // Membuat objek
12.         clsDokterSpesialis objDokSpesialis = new
            clsDokterSpesialis();
13.         System.out.println("Kelas Induk-clsDokter");
14.         System.out.printf("Masukkan Id.Dokter : ");
15.         objDokSpesialis.IdDokter = input.next();
16.         System.out.printf("Masukkan Nama : ");
```

4.1 Single Inheritance (Lanj..)

```
17.         objDokSpesialis>Nama = input.next();
18.         System.out.printf("Masukkan Gaji : ");
19.         objDokSpesialis.Gaji = input.nextInt();
20.         Tunjangan =
            objDokSpesialis.Tunjangan(objDokSpesialis.Gaji);
21.         System.out.printf("Tunjangan : %.2f\n", Tunjangan);
22.         System.out.println();
23.         System.out.println("Kelas Anak-clsdokterSpesialis");
24.         System.out.printf("Tunjangan Spesialis: ");
25.         objDokSpesialis.TunjanganSpesialis = input.nextInt();
26.         UangMakan =
            objDokSpesialis.UangMakan(objDokSpesialis.Gaji);
27.         System.out.printf("Uang Makan : %.2f\n", UangMakan);
28.         System.out.printf("Total Gaji : %.2f\n",
            objDokSpesialis.TotalGaji(objDokSpesialis.Gaji,
            Tunjangan,objDokSpesialis.TunjanganSpesialis ,UangMakan ));
29.     }
30. }
```

4.1 Single Inheritance (Lanj..)

- Contoh keluaran program.



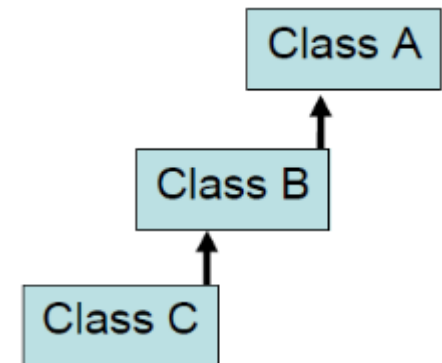
The screenshot shows an IDE window with three tabs: 'Report Problems Window', 'iReport output', and 'Output X'. The 'Output X' tab is active and displays the following text:

```
run:
Kelas Induk-clsDokter
Masukkan Id.Dokter : 99001
Masukkan Nama : Nanik
Masukkan Gaji : 10000000
Tunjangan : 1000000.00

Kelas Anak-clsDokterSpesialis
Tunjangan Spesialis: 5000000
Uang Makan : 1000000.00
Total Gaji : 17000000.00
BUILD SUCCESSFUL (total time: 25 seconds)
```

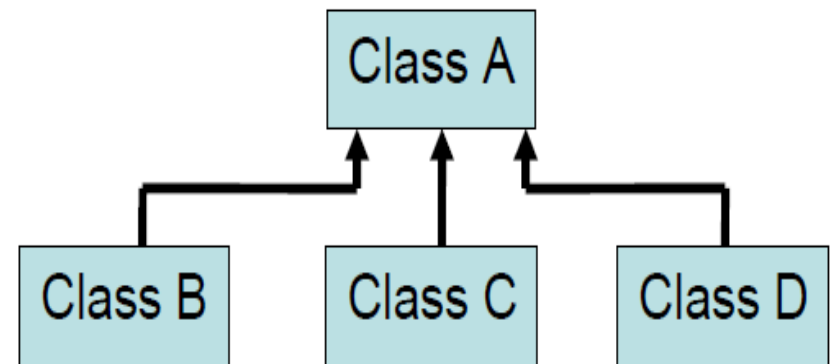
4.2 Multilevel Inheritance

- Sebuah kelas turunan dari sebuah kelas atau dari sub kelas. Jika kelas C inheritance dari kelas B, dan kelas B inherits dari kelas A, kelas C akan mewarisi semua member yang dideklarasikan di kelas B dan member yang dideklarasikan di kelas A.
- Perhatikan ilustrasi pada gambar.



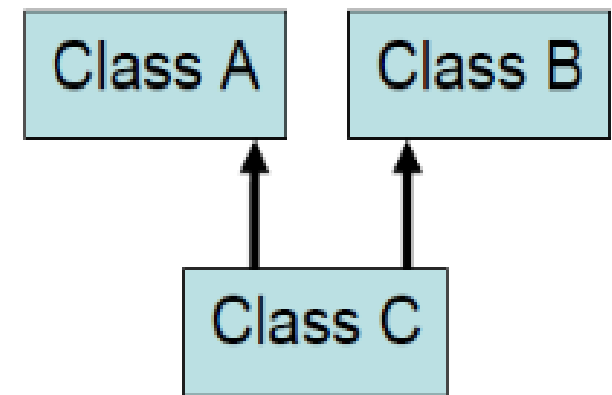
4.3 Hierarchical Inheritance

- Jenis perwarisan hierarchical adalah banyak memiliki sub kelas dari sebuah kelas induk. Bila kelas B, C, dan D inherit dari kelas A, maka kelas B, C, D akan mewariskan semua member yang dideklarasikan di kelas A. Perhatikan ilustrasi pada gambar.



4.4 Multiple Inheritance

- Jika kelas C inherits dari kelas A dan kelas B, maka kelas C akan acquire semua member yang dideklarasikan di kelas A dan kelas B. Kelas C mendapatkan semua member yang ada di kelas A dan kelas B. Perhatikan ilustrasi seperti pada gambar.



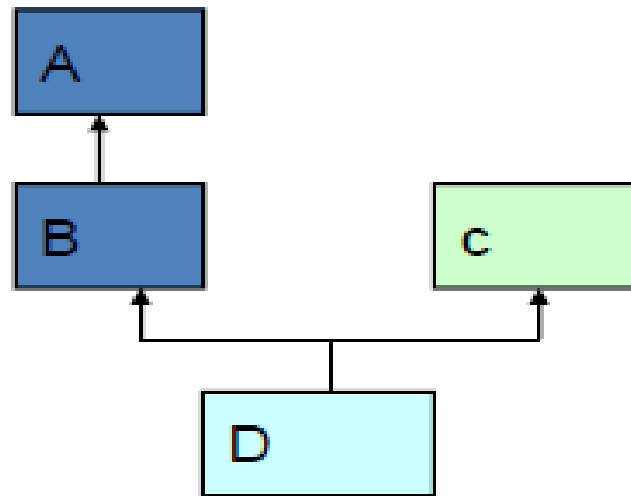
4.4 Multiple Inheritance (Lanj..)

- **Catatan:**

Di dalam mengimplementasikan multiple inheritance harus didukung oleh bahasa pemrograman, bila tidak maka konsep ini tidak bisa dilakukan. Java tidak mendukung multiple inheritance. Penerapan multiple inheritance sebaiknya dihindari karena dapat menimbulkan ambiguitas (diamond problem). Namun hal ini dapat disiasati dengan penerapan konsep interface.

4.5 Hybrid Inheritance

- Hybrid inheritance merupakan pewarisan yang menggabungkan beberapa jenis pewarisan. Perhatikan ilustrasi pewarisan hybrid seperti diilustrasikan pada gambar.



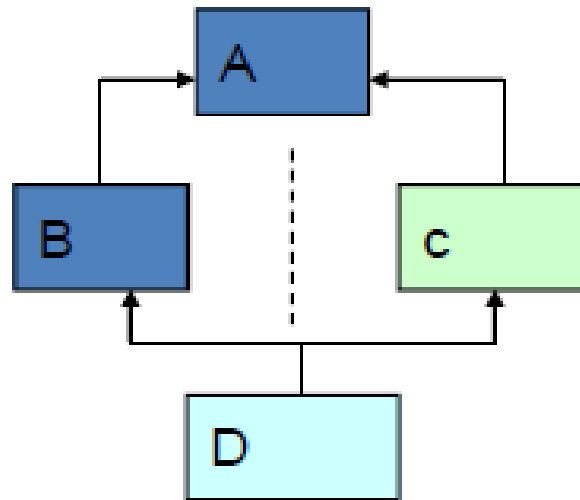
4.5 Hybrid Inheritance (Lanj..)

- **Catatan:**

Di dalam mengimplementasikan hybrid inheritance harus didukung oleh bahasa pemrograman. Bila tidak mendukung maka konsep ini tidak bisa dilakukan. Java tidak mendukung multiple inheritance. Penerapan hybrid inheritance sebaiknya dihindari karena dapat menimbulkan ambiguitas.

4.6 Multipath Inheritance

- Multipath inheritance merupakan pewarisan yang memiliki beberapa jalur pewarisan. Perhatikan jenis pewarisan multipath seperti diilustrasikan pada gambar.



4.6 Multipath Inheritance (Lanj..)

- **Catatan:**

Di dalam mengimplementasikan multipath inheritance harus didukung oleh bahasa pemrograman. Bila tidak mendukung maka konsep ini tidak bisa dilakukan. Penerapan multipath inheritance sebaiknya dihindari karena dapat menimbulkan ambiguitas.



5. Keyword Super

- Jika mendeklarasikan member (property atau method) dari subclass dengan nama yang sama dengan yang dimiliki superclass, maka hanya dapat mengakses member superclass tersebut dengan menggunakan keyword `super`.
- Keyword `super` menunjukkan bahwa kita ingin merujuk superclass dari class yang bersangkutan.

Lanj...

- Berikut contoh penggunaan keyword super. Buat sebuah kelas dengan nama **A**, kemudian buat program berikut ini.

Nama class: A

```
1.  package demoinheritance;  
2.  
3.  public class A  
4.  {  
5.      int x,y;  
6.  }
```

Lanj...

- Buat sebuah kelas dengan nama B.

Nama class: B

```
1. package demoinheritance;
2.
3. public class B extends A
4. {
5.     int x,y;
6.     void setxySuperClass(int x, int y)
7.     {
8.         super.x = x;
9.         super.y = y;
10.    }
11.    void setxy(int x, int y)
12.    {
13.        this.x = x;
14.        this.y = y;
15.    }
```

Lanj...

```
16.  
17.     void displayxySuperClass()  
18.     {  
19.         System.out.println("Nilai dari x dan y dari superclass :"  
    + super.x + " dan " + super.y);  
20.     }  
21.  
22.     void displayxy()  
23.     {  
24.         System.out.println("Nilai dari x dan y :" + super.x + "  
    dan " + super.y);  
25.     }  
26. }
```

- Selanjutnya membuat program di main program.




Lanj...

Main program: DemoInheritance

```
1.  package demoinheritance;
2.  import java.util.HashSet;
3.  import java.util.Set;
4.  public class DemoInheritance {
5.
6.      public static void main(String[] args)
7.      {
8.          // TODO code application logic here
9.          B subOb = new B();
10.
11.          subOb.setxy(10,20);
12.          subOb.setxySuperClass(30, 30);
13.          subOb.displayxy();
14.          subOb.displayxySuperClass();
15.      }
16. }
```

Lanj...

- Berikut contoh keluaran program.

Report Problems Window	iReport output	Output - Demolnheri
	<code>run:</code>	
	<code>Nilai dari x dan y :30 dan 30</code>	
	<code>Nilai dari x dan y dari superclass :30 dan 30</code>	
	<code>BUILD SUCCESSFUL (total time: 1 second)</code>	



6. Overriding & Overloading

6.1 Overriding

- Overriding adalah kemampuan subkelas (kelas turunan) untuk memodifikasi atribut dan method milik kelas induknya.
- Proses ini akan mengubah data dan method dari kedua class tersebut, class anak dan induknya.
- Apabila atribut dan method dideklarasikan sebagai private atau final tidak dapat dilakukan overriding.

6.1 Overriding (Lanj..)

- Override method merupakan method yang sama persis dengan method yang sudah ada di super kelasnya, biasanya perbedaannya adalah pada implementasi (program body).
- Overriding tidak bisa dilakukan dalam kelas itu sendiri.
- Jadi Overriding erat kaitannya dengan inheritance (pewarisan).

6.1 Overriding (Lanj..)

- Ada 2 (dua) alasan mengapa melakukan overriding:
 - Mendefinisikan kembali method class induknya secara total.
 - Menambahkan behavior (tingkah laku) tertentu pada method class induknya.

6.1 Overriding (Lanj..)

- Aturan yang perlu diperhatikan dalam melakukan overriding, yaitu:
 - Daftar argumen harus persis sama seperti pada method yang di override.
 - Tipe nilai balikan harus sama atau merupakan subtype dari tipe nilai balikan yang dideklarsikan dalam method asli pada superkelas.
 - Level akses tidak boleh lebih terbatas dari method yang dioverride.

6.1 Overriding (Lanj..)

- Method instance dapat dioverride hanya jika method tersebut diwarsisi oleh subkelas.
- Method yang dideklarasikan final tidak bisa dioverride.
- Konstruktor tidak bisa dioverride.

6.1 Overriding (Lanj..)

- Buat keas dengan nama clsNama_Asli.

Nama class: clsNama_Asli

```
1. package lat_overriding_1;
2.
3. public class clsNama_Asli
4. {
5.     public void Namanya()
6.     {
7.         System.out.println("Nama: Dewi");
8.     }
9. }
```

6.1 Overriding (Lanj..)

- Buat kelas dengan nama clsNama_Panggilan.

Nama class: clsNama_Panggilan

```
1.  package lat_overriding_1;
2.
3.  public class clsNama_Panggilan extends clsNama_Asli
4.  {
5.      public void Namanya()
6.      {
7.          System.out.println("Nama: Wati");
8.      }
9.  }
```

6.1 Overriding (Lanj..)

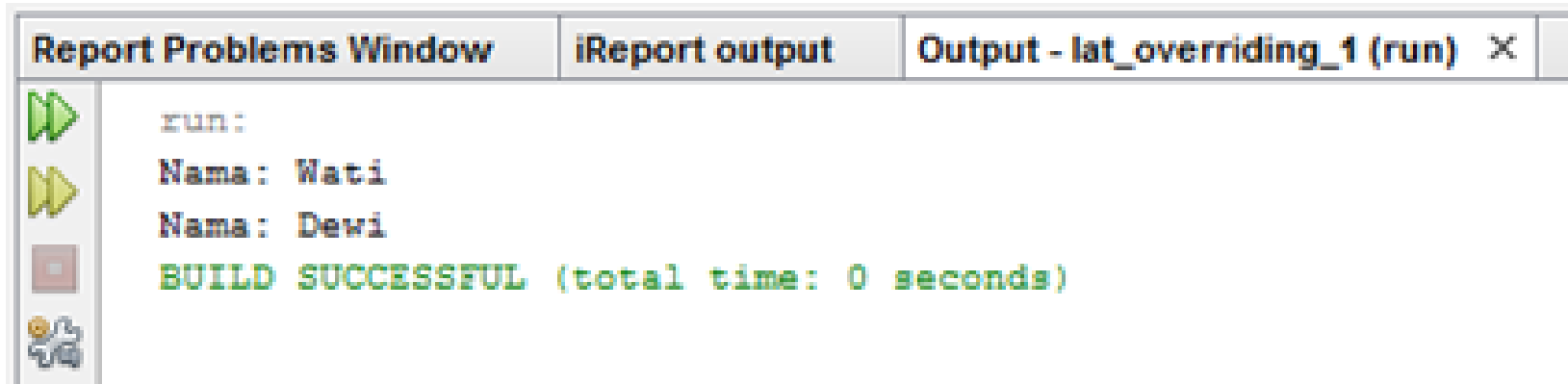
- Buat program berikut di main program.

Main Program: Lat_overriding_1

```
1.  package lat_overriding_1;
2.  public class Lat_overriding_1 {
3.      public static void main(String[] args)
4.      {
5.          // TODO code application logic here
6.          clsNama_Panggilan obj_panggilan = new
clsNama_Panggilan();
7.          clsNama_Asli obj_namaasli = new clsNama_Asli();
8.
9.          //memanggil method Namanya() pada kelas clsNama_Panggilan
10.         obj_panggilan.Namanya();
11.         //memanggil method Namanya() pada kelas clsNama_Asli
12.         obj_namaasli.Namanya();
13.     }
14. }
```

6.1 Overriding (Lanj..)

- Contoh keluaran program seperti berikut.



The screenshot shows a window titled "Output - lat_overriding_1 (run)" with a close button. On the left side of the window, there are four icons: a green play button, a yellow play button, a red stop button, and a blue bug icon. The main area of the window displays the following text:

```
run:  
Nama: Wati  
Nama: Dewi  
BUILD SUCCESSFUL (total time: 0 seconds)
```

6.2 Overloading

- Overloading memungkinkan suatu kelas memiliki beberapa method dengan nama sama tetapi memiliki implementasi atau argumen yang berbeda, sepanjang deklarasi dan parameternya berbeda, atau disebut signaturenya berbeda.
- Hal ini dimungkinkan asalkan deklarasi method membuat penanda berbeda di satu kelas.

6.2 Overloading (Lanj..)

- Penanda adalah kombinasi nama fungsi / method ditambah daftar parameter.
- Dengan penanda berbeda, bahasa pemrograman Java mampu membedakan metode mana yang perlu dieksekusi dengan mengenali tipe parameter-parameter yang dilewatkan.

6.2 Overloading (Lanj..)

- Aturan pendeklarasian overloading terhadap metode:
 - Nama method harus sama.
 - Daftar parameter harus berbeda.
 - Return type boleh sama, juga boleh berbeda.
- Perbedaan daftar parameter bukan hanya terjadi pada perbedaan banyaknya parameter, tetapi juga urutan dari parameter tersebut.

6.2 Overloading (Lanj..)

- Misalnya saja dua buah parameter berikut ini:
 - `function_member(int x, String n)`
 - `function_member(String n, int x)`
- Dua parameter tersebut dianggap berbeda.

6.2 Overloading (Lanj..)

<u>return type</u>	<u>nama method</u>	<u>daftar parameter</u>
void	Coba	(int t1)
void	Coba	(int t1, int t2)
void	Coba	(int t1, int t2, int t3)
void	Coba	(int t1, int t2, int t3, int t4)
<hr/>	<hr/>	<hr/>
↓	↓	↓
sama	sama	berbeda

6.2 Overloading (Lanj..)

Nama class: clsHitung

```
1.  package operasimatematik;
2.  public class clsHitung
3.  {
4.      public double hasil(int a, int b)
5.      {
6.          return a*b;
7.      }
8.      public double hasil(double a, int b, int c)
9.      {
10.         return a+b*c;
11.     }
12.     public double hasil( int a, double b, int c, int d)
13.     {
14.         return a+b+c+d;
15.     }
16. }
```

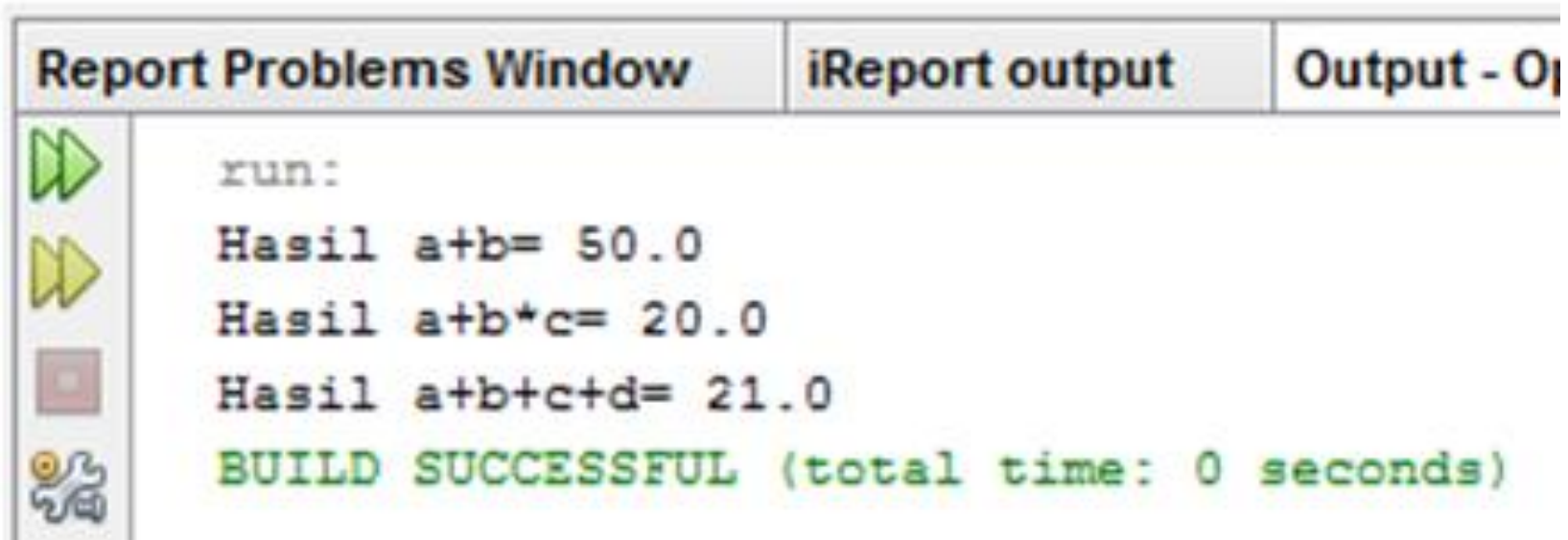
6.2 Overloading (Lanj..)

Main program: OperasiMatematik

```
1. package operasimatematik;
2. public class OperasiMatematik
3. {
4.     public static void main(String[] args)
5.     {
6.         // TODO code application logic here
7.         clsHitung objHitung = new clsHitung();
8.
9.         System.out.println("Hasil a+b="+objHitung.hasil(10,
10.         5));
11.         System.out.println("Hasil a+b*c="+objHitung.hasil(10,
12.         5, 2));
13.         System.out.println("Hasil
a+b+c+d="+objHitung.hasil(10, 5, 2, 4));
14.     }
15. }
```

6.2 Overloading (Lanj..)

- Keluaran program seperti berikut.



The screenshot shows a window titled "Report Problems Window" with a sub-tab "iReport output". The output text is as follows:

```
run:  
Hasil a+b= 50.0  
Hasil a+b*c= 20.0  
Hasil a+b+c+d= 21.0  
BUILD SUCCESSFUL (total time: 0 seconds)
```

Ringkasan:

- Salah satu keunggulan berorientasi objek adalah pada konsep pewarisan, dengan menggunakan konsep pewarisan program yang sama tidak perlu kita buat lagi, cukup mewariskan segala atribut dan method dengan konsep pewarisan.
- Jenis pewarisan yang dapat diimplementasikan dalam konsep program adalah: single inheritance, multilevel inheritance, hierarchical inheritance.
- Dalam konsep perwarisan juga dapat menerapkan konsep overriding, overloading.

Latihan Mandiri

1. Buat sebuah program yang mengimplementasikan single inheritance.
2. Buat sebuah program yang mengimplementasikan multilevel inheritance.
3. Buat sebuah program yang mengimplementasikan hierarchical inheritance.

Latihan Mandiri (Lanj...)

4. Buat sebuah program yang mengimplementasikan penggunaan keyword super pada inheritance.
5. Buat sebuah program yang mengimplementasikan overriding.
6. Buat sebuah program yang mengimplementasikan overloading.



TERIMA KASIH

U N I V E R S I T A S B U N D A M U L I A