

JOBSHEET 7
PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK
INHERITANCE & POLYMORPHISM
Dosen Pengajar: Vit Zuraida, S.Kom., MT.



Oleh:
Oddis Nur Alifathur Razaaq
2241760015
SIB – 2C

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
2024

1. KOMPETENSI

1. Memahami konsep dasar inheritance dan polymorphism
2. Mampu membuat suatu subclass dari suatu superclass tertentu.
3. Mampu membuat objek dari suatu subclass dan melakukan pengaksesan terhadap atribut dan method baik yang dimiliki sendiri atau turunan dari superclass nya.
4. Mampu membuat method overloading
5. Mampu membuat method overriding

2. PENDAHULUAN

Inheritance pada object oriented programming merupakan konsep **pewarisan** dari suatu class yang lebih umum ke suatu class yang lebih spesifik. Kelas yang menurunkan disebut kelas dasar (**base class/super class/parent class**), sedangkan kelas yang diturunkan disebut kelas turunan (**derived class/sub class/child class**). Setiap **subclass** akan “mewarisi” atribut dan method dari **superclass** yang bersifat *public* ataupun *protected*. Manfaat pewarisan adalah *reusability* atau penggunaan kembali baris kode.

Pada bahasa pemrograman Java, deklarasi inheritance dilakukan dengan cara menambahkan kata kunci **extends** setelah deklarasi nama class, kemudian diikuti dengan nama parent class nya. Kata kunci **extends** tersebut memberitahu kompiler Java bahwa kita ingin melakukan **extension/ perluasan** class. Berikut adalah contoh deklarasi inheritance.

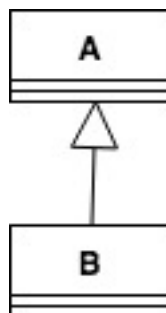
```
public class B extends A {  
    ...  
}
```

Contoh diatas memberitahukan kompiler Java bahwa class B meng**extend** class A. Artinya, class B adalah subclass dari class A dengan melakukan extension/perluasan. Extension atau perluasan ini akan dilakukan dengan penambahan atribut dan method khusus yang hanya dimiliki oleh class B.

Terdapat 3 bentuk pewarisan: single inheritance, multilevel inheritance, dan multiple inheritance.

1. Single Inheritance

Single inheritance adalah inheritance dimana suatu subclass hanya mempunyai satu parent class.

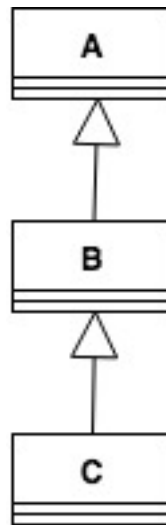


Gambar 1. Contoh Single Inheritance

2. Multilevel Inheritance

Multilevel inheritance adalah inheritance dengan subclass yang menjadi superclass bagi class yang lain.

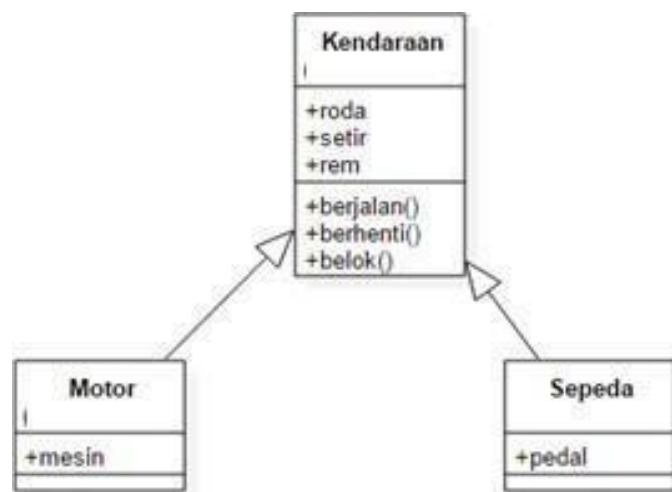
Contoh:



Gambar 2. Contoh Multilevel Inheritance

Pada Gambar 2 di atas dapat dilihat bahwa class B merupakan subclass dari class A, namun dia juga merupakan superclass dari class C. Inheritance dengan 2 level ini disebut multilevel inheritance.

Pada class diagram, inheritance digambarkan dengan sebuah garis solid dengan segitiga di ujungnya. Class yang dekat pada segitiga merupakan superclass, sedangkan class yang jauh dari segitiga merupakan subclass. Berikut ini adalah contoh class diagram dengan relasi inheritance:



Gambar 3 Contoh class diagram dalam inheritance

Suatu parent class bisa membatasi atribut dan method yang akan diwariskan kepada subclass-nya. Pembatasan tersebut dilakukan melalui penentuan access level modifier. Di dalam java, access level modifier atribut dan method dirangkum dalam tabel berikut ini:

| Modifier | class yang sama | package yang sama | subclass | class manapun |
|-----------|-----------------|-------------------|----------|---------------|
| private | √ | | | |
| default | √ | √ | | |
| protected | √ | √ | √ | |
| public | √ | √ | √ | √ |

Atribut dan method yang akan diwariskan dari parent class ke child class adalah atribut dan method dengan modifier protected atau public.

Kata kata kunci **this** dipakai untuk merujuk pada object/class itu sendiri. Sementara itu, kunci **super** dipakai untuk merujuk pada parent object/class. Format penulisannya adalah sebagai berikut:

- **super.<namaAtribut>**
Mengakses atribut parent
- **super.<namaMethod>()**
Memanggil method parent
- **super()**
Memanggil constructor parent, hanya dapat dilakukan pada baris pertama dalam constructor child
- **super(parameter1, parameter2,dst)**
Memanggil constructor parent class dengan parameter, hanya dapat dilakukan pada baris pertama dalam constructor child

Saat instansiasi objek dari subclass dilakukan, objek pada superclass juga akan terbentuk. Dengan kata lain, ketika constructor subclass dijalankan, pada “baris pertama” (atau sebelum baris- baris lainnya dalam constructor subclass dieksekusi) constructor superclass akan dijalankan terlebih dahulu.

Polymorphism terdiri dari 2 kata, yaitu poly (banyak), morph (bentuk). Konsep polimorfisme pada OOP membolehkan sebuah aksi diimplementasikan secara berbeda. Ada 2 bentuk polimorfisme, yaitu:

1. Overloading

- Method overloading berarti kondisi dimana ada method dengan nama yang sama, tetapi memiliki method signature yang berbeda.
- Method signature: jumlah, tipe data dan susunan parameter
- Method overloading dapat terjadi pada kelas yang sama atau kelas lain yang terkait dalam hierarki pewarisan.
- Karakteristik overloading: nama method sama, method signature berbeda, return type boleh sama atau berbeda.
- JVM menentukan method mana yang akan dipanggil pada compile-time \Rightarrow compile-time polymorphism
- Disebut juga static binding atau early binding

2. Overriding

- Overriding terjadi ketika child class memiliki method dengan nama dan signature yang samadengan parent class nya.
- Karakteristik: terjadi pada child class/kelas turunan, nama method sama, method signature sama.
- JVM menentukan method mana yang akan dipanggil pada saat run-time ² run-time polymorphism
- Disebut juga dynamic binding atau late binding

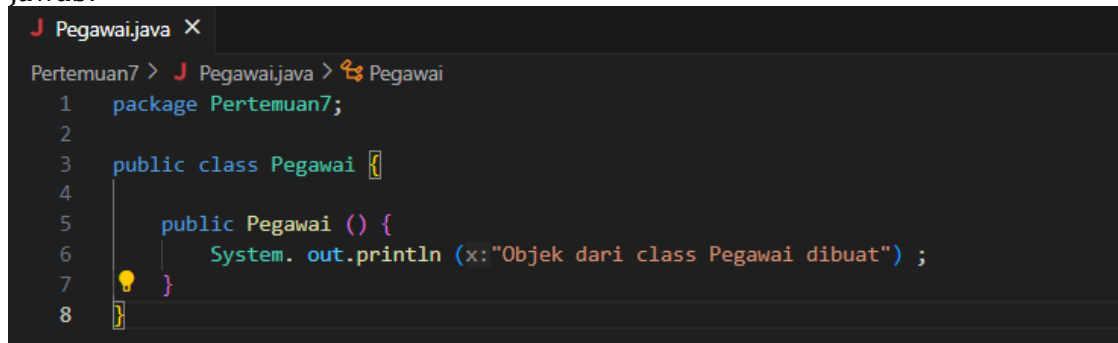
3. PERCOBAAN 1 (extends)

A. TAHAPAN PERCOBAAN

1. Buatlah sebuah parent class dengan nama Pegawai. Lalu buat constructor tanpa parameterdengan baris kode sebagai berikut:

```
public class Pegawai {  
  
    public Pegawai() {  
        System.out.println("Objek dari class Pegawai dibuat");  
    }  
}
```

Jawab:



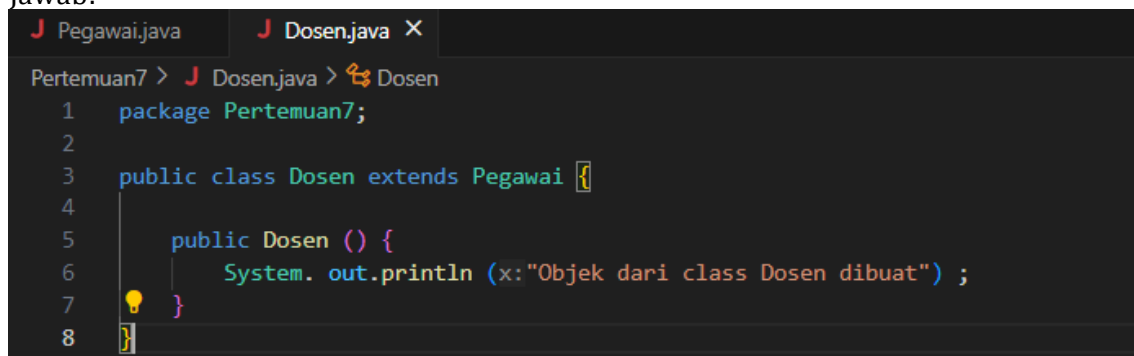
The screenshot shows an IDE window titled 'Pertemuan7 > J Pegawai.java > Pegawai'. The code is as follows:

```
1 package Pertemuan7;  
2  
3 public class Pegawai {  
4  
5     public Pegawai () {  
6         System.out.println (x:"Objek dari class Pegawai dibuat") ;  
7     }  
8 }
```

2. Buatlah subclass dari class Pegawai dengan nama Dosen, kemudian buat juga constructortanpa parameter dengan baris kode berikut:

```
public class Dosen extends Pegawai {  
  
    public Dosen() {  
        System.out.println("Objek dari class Dosen dibuat");  
    }  
}
```

Jawab:



The screenshot shows an IDE window titled 'Pertemuan7 > J Dosen.java > Dosen'. The code is as follows:

```
1 package Pertemuan7;  
2  
3 public class Dosen extends Pegawai {  
4  
5     public Dosen () {  
6         System.out.println (x:"Objek dari class Dosen dibuat") ;  
7     }  
8 }
```

3. Buatlah main class, misal InheritanceDemo.java, lakukan instansiasi objek baru bernamadosen1 dari class Dosen sebagai berikut:

```
public static void main(String[] args) {
    Dosen dosen1 = new Dosen();
}
```

Jawab:

```
J Pegawai.java      J Dosen.java      J InheritanceDemo.java 1 X
Pertemuan7 > J InheritanceDemo.java > InheritanceDemo
1  package Pertemuan7;
2
3  public class InheritanceDemo {
4      Run | Debug
5      public static void main(String[] args) {
6          Dosen dosen1 = new Dosen();
7      }
8  }
```

- Run programnya kemudian amati hasilnya.

Jawab:

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' -Xmx512m -Xms128m -Djre.properties -Duser.workspaceStorage\59d2aa27edbc67c7e9901581e059
itanceDemo'
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat
PS D:\PBO>
```

B. PERTANYAAN

1. Pada percobaan 1 diatas, tentukan child class dan parent class!

Jawab: Child classnya adalah class Dosen. Sedangkan untuk parent classnya adalah class Pegawai.

2. Kata kunci apa yang membuat child class dan parent class tersebut memiliki relasi?

Jawab: Kata kunci `extends`, dimana kata kunci `extends` menunjukkan bahwa class `Dosen` mewarisi semua atribut dan method dari class `Pegawai`. Sehingga yang membuat child class = class `Dosen` dengan parent class = class `Pegawai` memiliki sebuah relasi yaitu karena terdapat kata kunci `extends`.

3. Berdasarkan hasil yang ditampilkan oleh program, ada berapa constructor yang dieksekusi? Constructor class mana yang lebih dulu dieksekusi?

Jawab: Ada dua constructor yang dieksekusi , constructor yang dieksekusi terlebih dahulu adalah class Pegawai kemudian baru constructor class Dosen yang dieksekusi.

4. PERCOBAAN 2 (Pewarisan)

A. TAHAPAN PERCOBAAN

1. Tambahkan atribut nip, nama, dan gaji serta method getInfo() pada class Pegawai

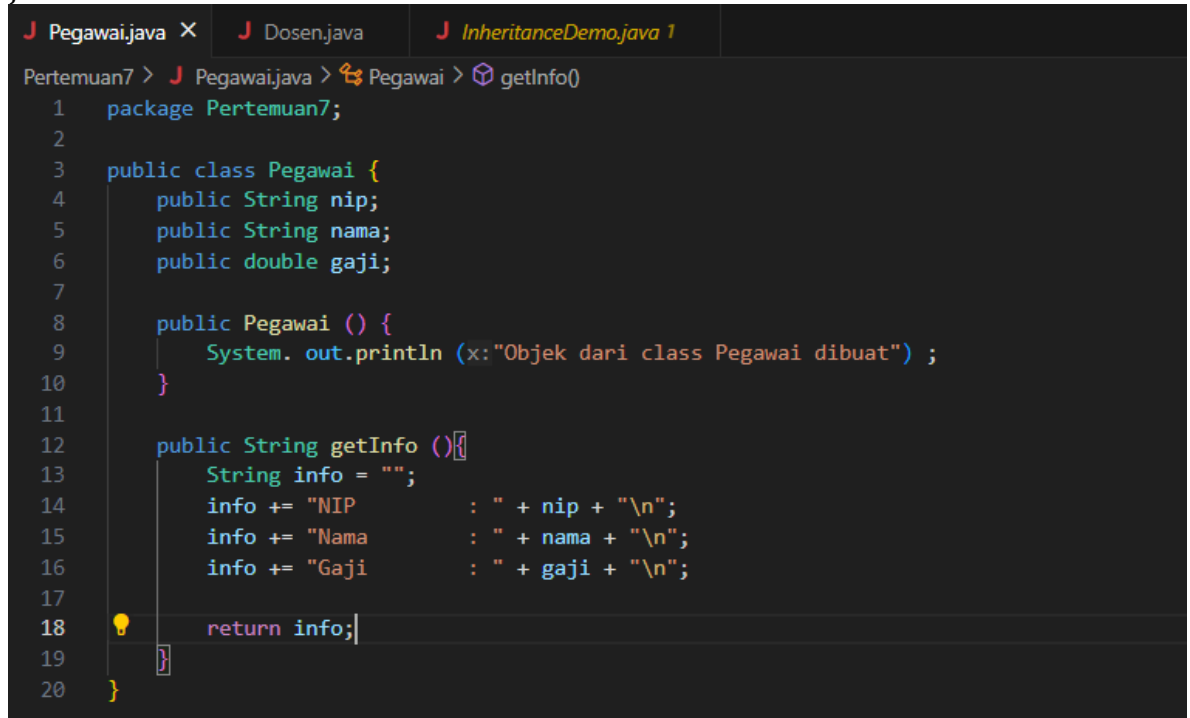
```
public class Pegawai {
    public String nip;
    public String nama;
    public double gaji;

    public Pegawai() {
        System.out.println("Objek dari class Pegawai dibuat");
    }

    public String getInfo() {
        String info = "";
        info += "NIP          : " + nip + "\n";
        info += "Nama          : " + nama + "\n";
        info += "Gaji           : " + gaji + "\n";

        return info;
    }
}
```

Jawab:



The screenshot shows an IDE with three tabs: 'Pegawai.java', 'Dosen.java', and 'InheritanceDemo.java 1'. The 'Pegawai.java' tab is active, showing the following code:

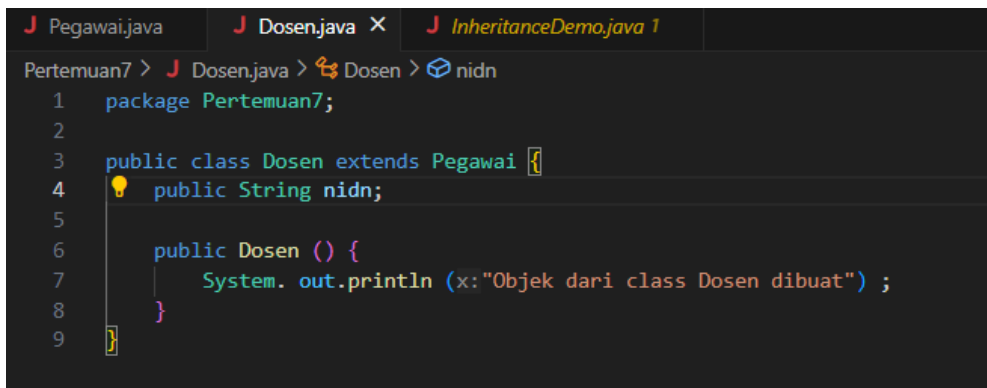
```
Pertemuan7 > J Pegawai.java > Pegawai > getInfo()
1  package Pertemuan7;
2
3  public class Pegawai {
4      public String nip;
5      public String nama;
6      public double gaji;
7
8      public Pegawai () {
9          System.out.println (x:"Objek dari class Pegawai dibuat") ;
10     }
11
12     public String getInfo () {
13         String info = "";
14         info += "NIP          : " + nip + "\n";
15         info += "Nama          : " + nama + "\n";
16         info += "Gaji           : " + gaji + "\n";
17
18         return info;
19     }
20 }
```

2. Tambahkan pula atribut NIDN pada class Dosen

```
public class Dosen extends Pegawai {
    public String nidn;

    public Dosen() {
        System.out.println("Objek dari class Dosen dibuat");
    }
}
```

Jawab:



```
Pertemuan7 > J Dosen.java > Dosen > nidn
1 package Pertemuan7;
2
3 public class Dosen extends Pegawai {
4     public String nidn;
5
6     public Dosen () {
7         System.out.println (x:"Objek dari class Dosen dibuat") ;
8     }
9 }
```

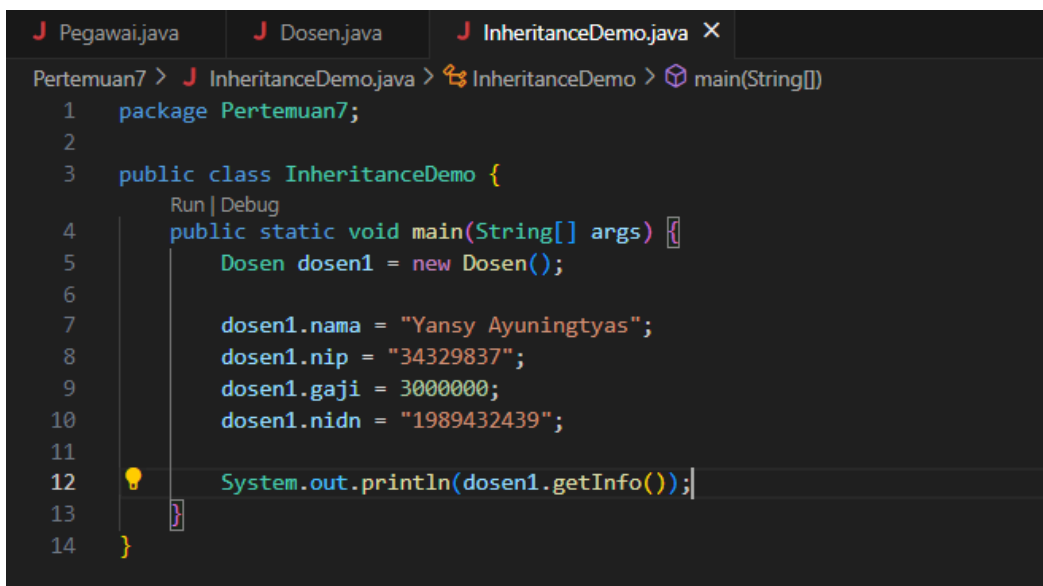
3. Pada class InheritanceDemo.java tuliskan baris kode berikut:

```
public static void main(String[] args) {
    Dosen dosen1 = new Dosen();

    dosen1.nama = "Yansy Ayuningtyas";
    dosen1.nip = "34329837";
    dosen1.gaji = 3000000;
    dosen1.nidn = "1989432439";

    System.out.println(dosen1.getInfo());
}
```

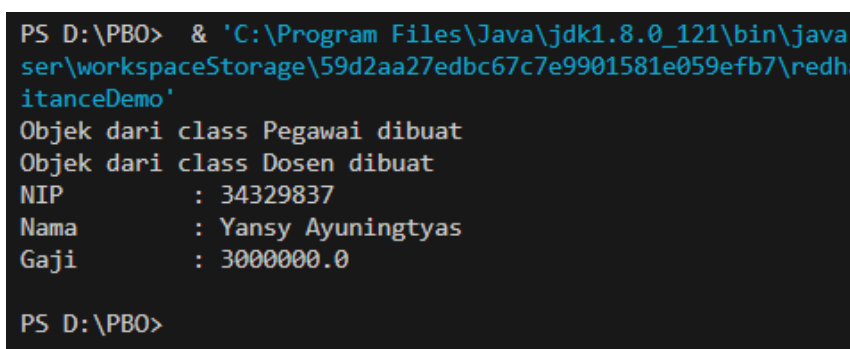
Jawab:



```
Pertemuan7 > J InheritanceDemo.java > InheritanceDemo > main(String[])
1 package Pertemuan7;
2
3 public class InheritanceDemo {
4     public static void main(String[] args) {
5         Dosen dosen1 = new Dosen();
6
7         dosen1.nama = "Yansy Ayuningtyas";
8         dosen1.nip = "34329837";
9         dosen1.gaji = 3000000;
10        dosen1.nidn = "1989432439";
11
12        System.out.println(dosen1.getInfo());
13    }
14 }
```

4. Run program kemudian amati hasilnya

Jawab:



```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java'
ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redh
itanceDemo'
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
PS D:\PBO>
```


B. PERTANYAAN

1. Pada percobaan 2 diatas, apakah program dapat berhasil dijalankan atautkah terjadi error?

Jawab: Program dapat berhasil dijalankan, namun atribut NIDN tidak ditampilkan karena atribut tersebut tidak ditampilkan atau dicetak info di `getInfo()`, dimana dalam class `inheritanceDemo` yang dicetak adalah `getInfo()`. Sehingga atribut NIDN tidak ditampilkan.

2. Jika program berhasil dijalankan, mengapa tidak terjadi error pada assignment/pengisian nilai atribut nip, gaji, dan NIDN pada object `dosen1` padahal tidak ada deklarasi ketiga atribut tersebut pada class `Dosen`?

Jawab: Karena class `Dosen` mewarisi semua atribut yang ada didalam class `Pegawai`. Sehingga atribut nip, nama dan gaji yang telah dideklarasikan class `Pegawai` dapat diakses dan diisi nilainya oleh objek `dosen1` yang merupakan instance dari class `Dosen`.

3. Jika program berhasil dijalankan, mengapa tidak terjadi error pada pemanggilan method `getInfo()` oleh object `dosen1` padahal tidak ada deklarasi method `getInfo()` pada class `Dosen`?

Jawab: Karena method `getInfo()` juga merupakan bagian dari class `Pegawai` yang telah diwarisi oleh class `Dosen`. Sehingga objek `dosen1` dapat memanggil method `getInfo()` tanpa perlu mendeklarasikan ulang method tersebut diclass `Dosen`.

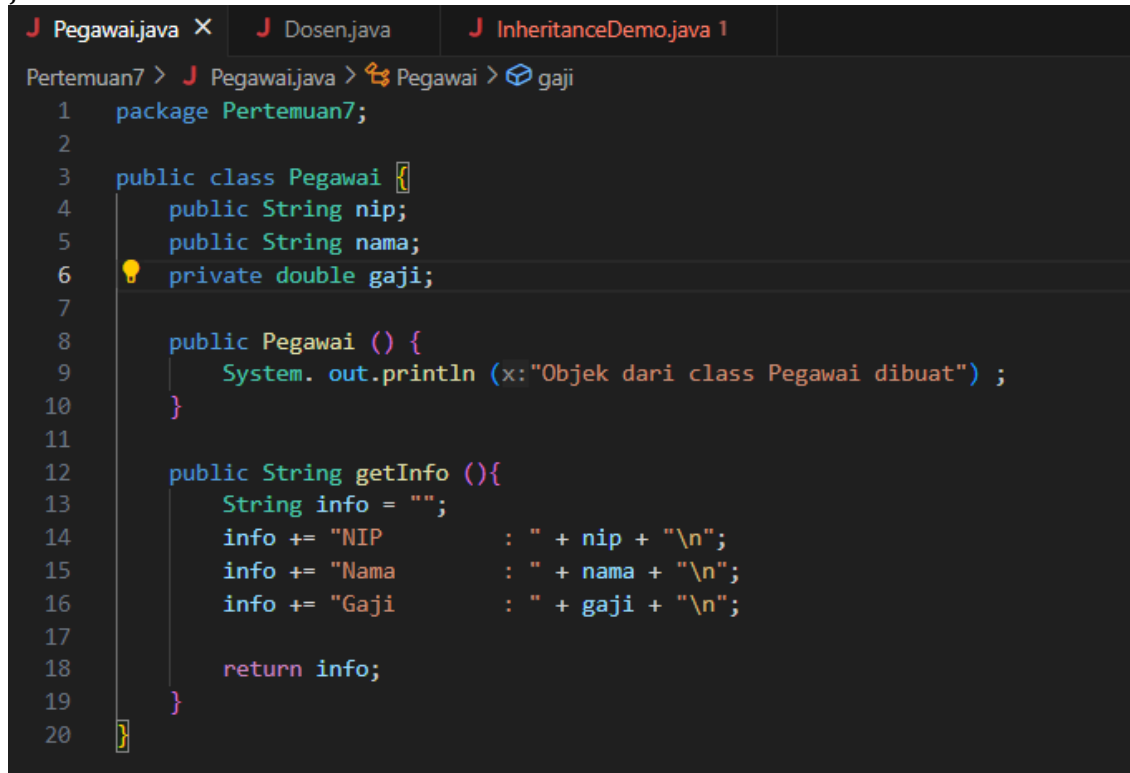
5. PERCOBAAN 3 (Hak akses)

A. TAHAPAN PERCOBAAN

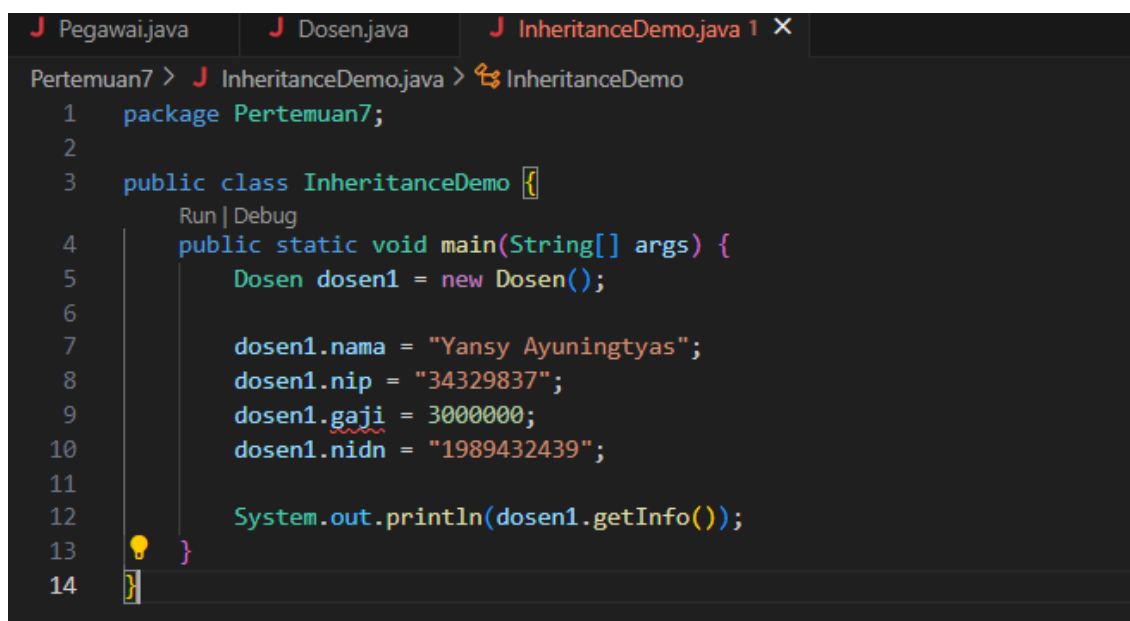
1. Modifikasi access level modifier pada atribut gaji menjadi private pada class Pegawai.java

```
public class Pegawai {  
    public String nip;  
    public String nama;  
    private double gaji;  
}
```

Jawab:



```
J Pegawai.java X J Dosen.java J InheritanceDemo.java 1  
Pertemuan7 > J Pegawai.java > Pegawai > gaji  
1 package Pertemuan7;  
2  
3 public class Pegawai {  
4     public String nip;  
5     public String nama;  
6     private double gaji;  
7  
8     public Pegawai () {  
9         System.out.println(x: \"Objek dari class Pegawai dibuat\");  
10    }  
11  
12    public String getInfo () {  
13        String info = \"\";  
14        info += \"NIP : \" + nip + \"\\n\";  
15        info += \"Nama : \" + nama + \"\\n\";  
16        info += \"Gaji : \" + gaji + \"\\n\";  
17  
18        return info;  
19    }  
20 }
```

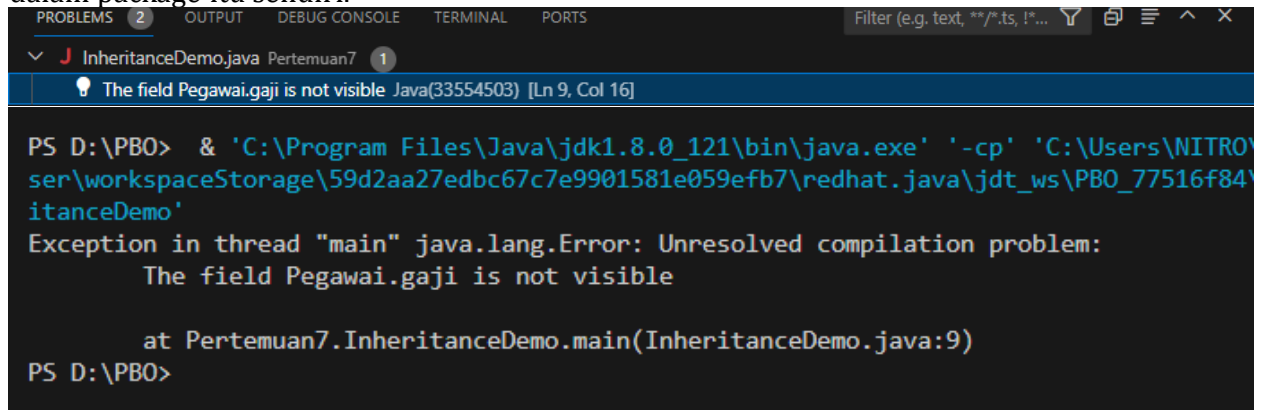


```
J Pegawai.java J Dosen.java J InheritanceDemo.java 1 X  
Pertemuan7 > J InheritanceDemo.java > InheritanceDemo  
1 package Pertemuan7;  
2  
3 public class InheritanceDemo {  
4     Run | Debug  
5     public static void main(String[] args) {  
6         Dosen dosen1 = new Dosen();  
7  
8         dosen1.nama = \"Yansy Ayuningtyas\";  
9         dosen1.nip = \"34329837\";  
10        dosen1.gaji = 3000000;  
11        dosen1.nidn = \"1989432439\";  
12  
13        System.out.println(dosen1.getInfo());  
14    }  
15 }
```

2. Run program kemudian amati hasilnya.

Jawab: Maka akan terjadi error di bagian atribut gaji, karena pada class Pegawai gaji dimodifikasi menjadi private, yang dimana hal ini tidak bisa dilakukan pewarisan kepada class Dosen, karena bersifat private hanya dapat diakses class itu sendiri dan

dalam package itu sendiri.



```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
Filter (e.g. text, **/*.ts, !*...

InheritanceDemo.java Pertemuan7 1
The field Pegawai.gaji is not visible Java(33554503) [Ln 9, Col 16]

PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp' 'C:\Users\NITRO\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\InheritanceDemo'
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
    The field Pegawai.gaji is not visible

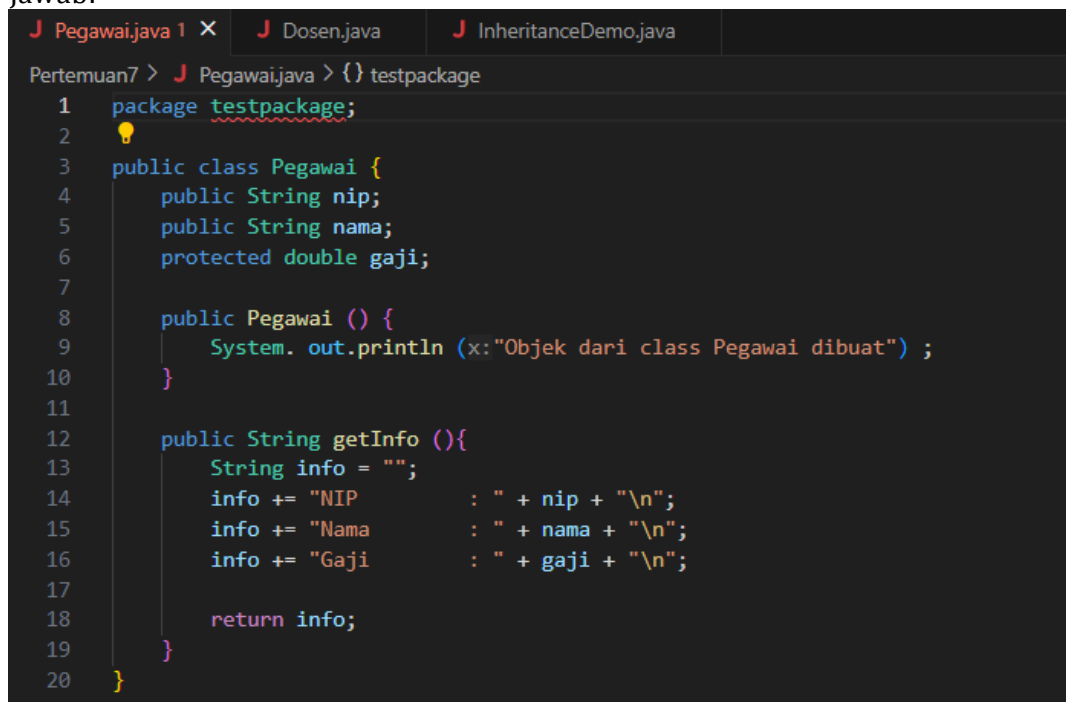
    at Pertemuan7.InheritanceDemo.main(InheritanceDemo.java:9)
PS D:\PBO>
```

- Ubah access level modifier atribut gaji menjadi protected kemudian pindah class Pegawai kepackage baru, misalnya "testpackage".

```
package testpackage;

public class Pegawai {
    public String nip;
    public String nama;
    protected double gaji;
}
```

Jawab:



```
Pegawai.java 1 X Dosen.java InheritanceDemo.java
Pertemuan7 > J Pegawai.java > {} testpackage

1 package testpackage;
2
3 public class Pegawai {
4     public String nip;
5     public String nama;
6     protected double gaji;
7
8     public Pegawai () {
9         System.out.println(x:"Objek dari class Pegawai dibuat") ;
10    }
11
12    public String getInfo (){
13        String info = "";
14        info += "NIP      : " + nip + "\n";
15        info += "Nama     : " + nama + "\n";
16        info += "Gaji      : " + gaji + "\n";
17
18        return info;
19    }
20 }
```

- Import class Pegawai dari testpackage pada class Dosen.

```
package inheritance;
import testpackage.Pegawai;
```

Jawab:

```

J Pegawai.java 1      J Dosen.java 1 X      J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen
1  package Pertemuan7;
2  import testpackage.Pegawai;
3
4  public class Dosen extends Pegawai {
5      public String nidn;
6
7      public Dosen () {
8          System.out.println (x:"Objek dari class Dosen dibuat") ;
9      }
10 }

```

5. Akses atribut gaji pada class Dosen dengan coba mencetak atribut gaji pada constructorDosen

```

public Dosen() {
    System.out.println(gaji);
    System.out.println("Objek dari class Dosen dibuat");
}

```

Jawab:

```

J Pegawai.java 1      J Dosen.java 1 X      J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen
1  package Pertemuan7;
2  import testpackage.Pegawai;
3
4  public class Dosen extends Pegawai {
5      public String nidn;
6
7      public Dosen () {
8          System.out.println(gaji);
9          System.out.println (x:"Objek dari class Dosen dibuat") ;
10 }
11 }

```

6. Ubah kembali access level modifier menjadi public dan kembalikan class Pegawai ke package semula.

Jawab:

```

J Pegawai.java X      J Dosen.java      J InheritanceDemo.java
Pertemuan7 > J Pegawai.java > Pegawai > gaji
1  package Pertemuan7;
2
3  public class Pegawai {
4      public String nip;
5      public String nama;
6      public double gaji;
7
8      public Pegawai () {
9          System.out.println (x:"Objek dari class Pegawai dibuat") ;
10 }
11
12 public String getInfo () {
13     String info = "";
14     info += "NIP" : " + nip + "\n";
15     info += "Nama" : " + nama + "\n";
16     info += "Gaji" : " + gaji + "\n";
17
18     return info;
19 }
20 }

```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe'
ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redhat.jav
itanceDemo'
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
PS D:\PBO>
```

B. PERTANYAAN

1. Pada langkah 1 di atas, terjadi error karena object dosen1 tidak dapat mengakses atribut gaji. Padahal gaji merupakan atribut Pegawai yang merupakan parent class dari Dosen. Mengapa hal ini dapat terjadi?

Jawab: Karena pada class Pegawai gaji dimodifikasi menjadi private, yang dimana hal ini tidak bisa dilakukan pewarisan kepada class Dosen, karena bersifat private hanya dapat diakses class itu sendiri dan dalam package itu sendiri. Sehingga akan terjadi error.

2. Pada langkah 5, setelah class Pegawai berpindah ke package yang berbeda, class Dosen masih dapat mengakses atribut gaji. Mengapa?

Jawab: Karena atribut gaji pada class Pegawai diberikan access level modifier protected, sehingga atribut tersebut dapat diakses oleh class dalam package yang sama.

3. Berdasarkan percobaan tersebut, bagaimana menentukan atribut dan method yang akan diwariskan oleh parent class ke child class?

Jawab: Dengan menentukan hak akses pada parent class. Jika didalam parent class terdapat atribut atau method yang semisalkan access level modifiernya private maka tidak dapat diwariskan ke child class. Sedangkan jika ingin mewariskan semua atribut atau method yang ada diparent class maka lebih baiknya bersifat public agar dapat diwariskan ke child class.

6. PERCOBAAN 4 (Super - atribut)

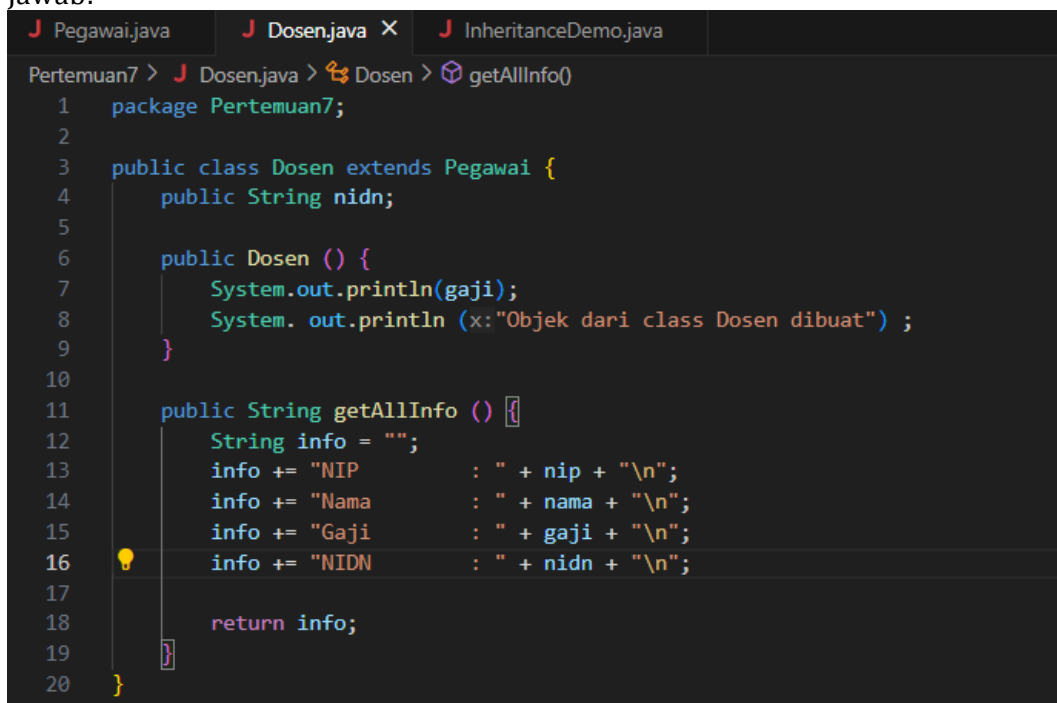
A. TAHAPAN PERCOBAAN

1. Butlah method `getAllInfo()` pada class `Dosen`

```
public String getAllInfo() {
    String info = "";
    info += "NIP          : " + nip + "\n";
    info += "Nama          : " + nama + "\n";
    info += "Gaji           : " + gaji + "\n";
    info += "NIDN            : " + nidn + "\n";

    return info;
}
```

Jawab:



The screenshot shows an IDE with three tabs: `Pegawai.java`, `Dosen.java` (active), and `InheritanceDemo.java`. The `Dosen.java` file is open, showing the following code:

```
Pertemuan7 > J Dosen.java > Dosen > getAllInfo()
1  package Pertemuan7;
2
3  public class Dosen extends Pegawai {
4      public String nidn;
5
6      public Dosen () {
7          System.out.println(gaji);
8          System.out.println (x:"Objek dari class Dosen dibuat") ;
9      }
10
11     public String getAllInfo () {
12         String info = "";
13         info += "NIP          : " + nip + "\n";
14         info += "Nama          : " + nama + "\n";
15         info += "Gaji           : " + gaji + "\n";
16         info += "NIDN            : " + nidn + "\n";
17
18         return info;
19     }
20 }
```

2. Lakukan pemanggilan method `getAllInfo()` oleh object `dosen1` pada class `InheritanceDemo.java`

```
public static void main(String[] args) {
    Dosen dosen1 = new Dosen();

    dosen1.nama = "Yansy Ayuningtyas";
    dosen1.nip = "34329837";
    dosen1.gaji = 3000000;
    dosen1.nidn = "1989432439";

    System.out.println(dosen1.getAllInfo());
}
```

Jawab:

```
J Pegawai.java J Dosen.java J InheritanceDemo.java X
Pertemuan7 > J InheritanceDemo.java > InheritanceDemo > main(String[])
1 package Pertemuan7;
2
3 public class InheritanceDemo {
4     Run | Debug
5     public static void main(String[] args) {
6         Dosen dosen1 = new Dosen();
7
8         dosen1.nama = "Yansy Ayuningtyas";
9         dosen1.nip = "34329837";
10        dosen1.gaji = 3000000;
11        dosen1.nidn = "1989432439";
12
13        System.out.println(dosen1.getAllInfo());
14    }
}
```

3. Run program kemudian amati hasilnya

Jawab:

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe'
ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redhat
itanceDemo'
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

PS D:\PBO>
```

4. Lakukan modifikasi method getAllInfo() pada class Dosen

```
public String getAllInfo() {
    String info = "";
    info += "NIP      : " + this.nip + "\n";
    info += "Nama     : " + this.nama + "\n";
    info += "Gaji     : " + this.gaji + "\n";
    info += "NIDN     : " + this.nidn + "\n";

    return info;
}
```

Jawab:

```

J Pegawai.java    J Dosen.java X    J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > getAllInfo()
1  package Pertemuan7;
2
3  public class Dosen extends Pegawai {
4      public String nidn;
5
6      public Dosen () {
7          System.out.println(gaji);
8          System.out.println (x:"Objek dari class Dosen dibuat") ;
9      }
10
11     public String getAllInfo () {
12         String info = "";
13         info += "NIP          : " + this.nip + "\n";
14         info += "Nama          : " + this.nama + "\n";
15         info += "Gaji          : " + this.gaji + "\n";
16         info += "NIDN         : " + this.nidn + "\n";
17
18         return info;
19     }
20 }

```

5. Run program kemudian bandingkan hasilnya dengan langkah no 2.

Jawab: Tidak ada perbedaan karena kedua kode tersebut meskipun berbeda kode yang dilangkah no 2 tidak ada this dan kode dilangkah no 4 terdapat this akan tetap menghasilkan output yang sama ketika memanggil method getAllInfo(). Penggunaan this ini berguna untuk kejelasan bahwa atribut tersebut merupakan atribut instance dari class Dosen atau kata kunci this digunakan untuk merujuk pada class Dosen itu sendiri jika atribut di class Dosen itu ada yang tidak ada maka akan mencari di parent classnya, sedangkan pada langkah 2 tidak menggunakan kata kunci this ini berarti merujuk pada atribut terdekat, dimana pada langkah no 2 mencari atribut terdekat di class itu jika tidak ada baru mencari diparent classnya. Sehingga hal ini tidak mempengaruhi cara atribut tersebut diakses atau output yang dihasilkan dengan menggunakan this dan tanpa this.

```

PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe'
ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redhat
itanceDemo'
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP          : 34329837
Nama          : Yansy Ayuningtyas
Gaji          : 3000000.0
NIDN         : 1989432439

PS D:\PBO>

```

6. Lakukan modifikasi method getAllInfo() pada class Dosen kembali

```

public String getAllInfo() {
    String info = "";
    info += "NIP          : " + super.nip + "\n";
    info += "Nama          : " + super.nama + "\n";
    info += "Gaji          : " + super.gaji + "\n";
    info += "NIDN         : " + super.nidn + "\n";

    return info;
}

```

Jawab:


```

J Pegawai.java    J Dosen.java 1 X    J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > getAllInfo()
1  package Pertemuan7;
2
3  public class Dosen extends Pegawai {
4      public String nidn;
5
6      public Dosen () {
7          System.out.println(gaji);
8          System.out.println (x:"Objek dari class Dosen dibuat") ;
9      }
10
11     public String getAllInfo () {
12         String info = "";
13         info += "NIP          : " + super.nip + "\n";
14         info += "Nama          : " + super.nama + "\n";
15         info += "Gaji          : " + super.gaji + "\n";
16         info += "NIDN          : " + super.nidn + "\n";
17
18         return info;
19     }
20 }

```

7. Run program kemudian bandingkan hasilnya dengan program pada no 1 dan no 4.
 Jawab: Maka hasilnya untuk program 1 dan 4 tidak mengalami error karena dilangkah 1 dan 4 menggunakan this dan juga tidak menggunakan this. Sehingga tidak mempengaruhi cara atribut tersebut diakses. Namun pada percobaan langkah ke 6 program menjadi error dikarenakan semua diganti menggunakan super disemua atribut, sedangkan atribut yang seharusnya dapat menggunakan kata kunci super itu merupakan atribut yang diwariskan dari class Pegawai, sedangkan atribut NIDN bukan atribut yang diwariskan dari class Pegawai. Karena atribut tersebut milik class Dosen, sehingga jika menggunakan kata kunci super akan error.

```

PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp' 'C:\Users\NITRO\ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\itanceDemo'
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
    nidn cannot be resolved or is not a field

    at Pertemuan7.Dosen.getAllInfo(Dosen.java:16)
    at Pertemuan7.InheritanceDemo.main(InheritanceDemo.java:12)
PS D:\PBO>

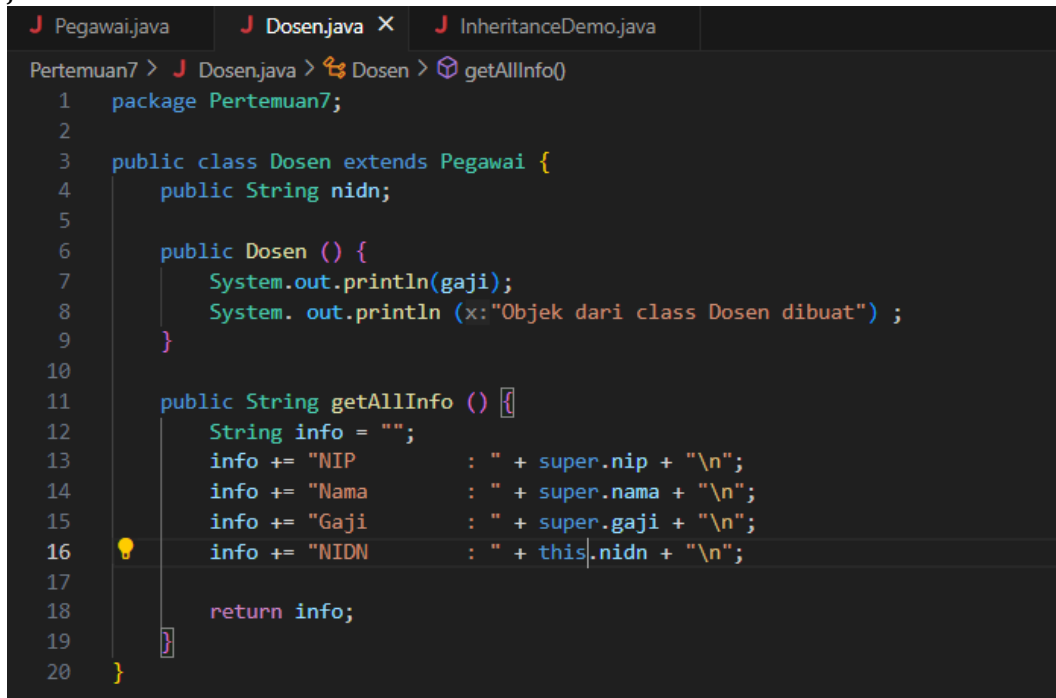
```

8. Lakukan modifikasi method `getAllInfo()` pada class Dosen kembali

```
public String getAllInfo() {
    String info = "";
    info += "NIP      : " + super.nip + "\n";
    info += "Nama      : " + super.nama + "\n";
    info += "Gaji       : " + super.gaji + "\n";
    info += "NIDN      : " + this.nidn + "\n";

    return info;
}
```

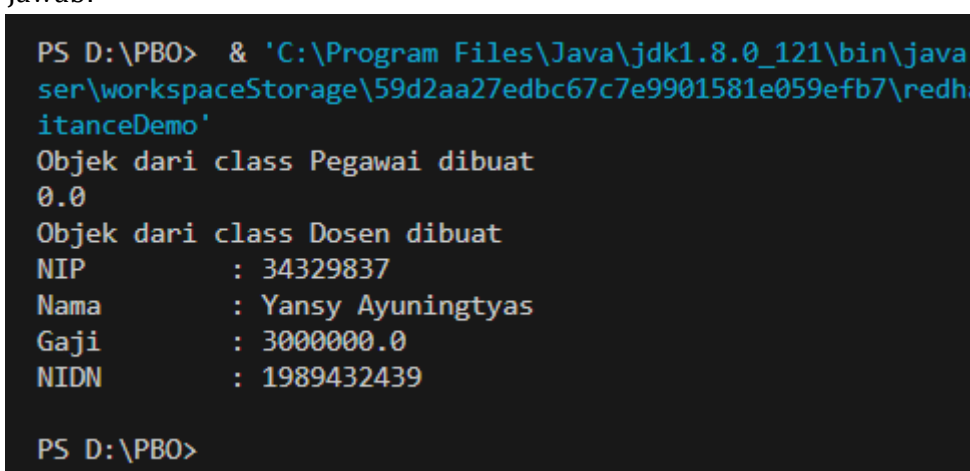
Jawab:



```
J Pegawai.java  J Dosen.java X  J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > getAllInfo()
1  package Pertemuan7;
2
3  public class Dosen extends Pegawai {
4      public String nidn;
5
6      public Dosen () {
7          System.out.println(gaji);
8          System.out.println (x:"Objek dari class Dosen dibuat") ;
9      }
10
11     public String getAllInfo () {
12         String info = "";
13         info += "NIP      : " + super.nip + "\n";
14         info += "Nama      : " + super.nama + "\n";
15         info += "Gaji       : " + super.gaji + "\n";
16         info += "NIDN      : " + this.nidn + "\n";
17
18         return info;
19     }
20 }
```

9. Run program kemudian bandingkan hasilnya dengan program pada no 2 dan no 4.

Jawab:



```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java
ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redh
itanceDemo'
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP      : 34329837
Nama      : Yansy Ayuningtyas
Gaji      : 3000000.0
NIDN     : 1989432439

PS D:\PBO>
```

B. PERTANYAAN

1. Apakah terdapat perbedaan hasil nama, nip, dan gaji yang ditampilkan pada program 1,4,dan 8? Mengapa?

Jawab: Tidak, karena program 1 tidak menggunakan kata kunci apapun sehingga merujuk pada atribut terdekat yaitu pertama mencari di class itu sendiri jika tidak ada baru mencari diparent classnya. Sedangkan untuk program 4 menggunakan `this` yang artinya merujuk pada class Dosen itu sendiri yang telah diwariskan oleh parent class. Sedangkan untuk program 8 ini terdapat kata kunci `super` untuk merujuk pada atribut

parent class dan kata kunci this untuk merujuk ke atribut terdekat yaitu dalam class itu sendiri (NIDN).

2. Mengapa error terjadi pada program no 6?

Jawab: Karena penggunaan kata kunci super digunakan disemua atribut. Sedangkan atribut NIDN merupakan atribut child class atau class Dosen itu sendiri. Sehingga tidak dapat menggunakan kata kunci super karena kata kunci super itu langsung merujuk pada parent class.

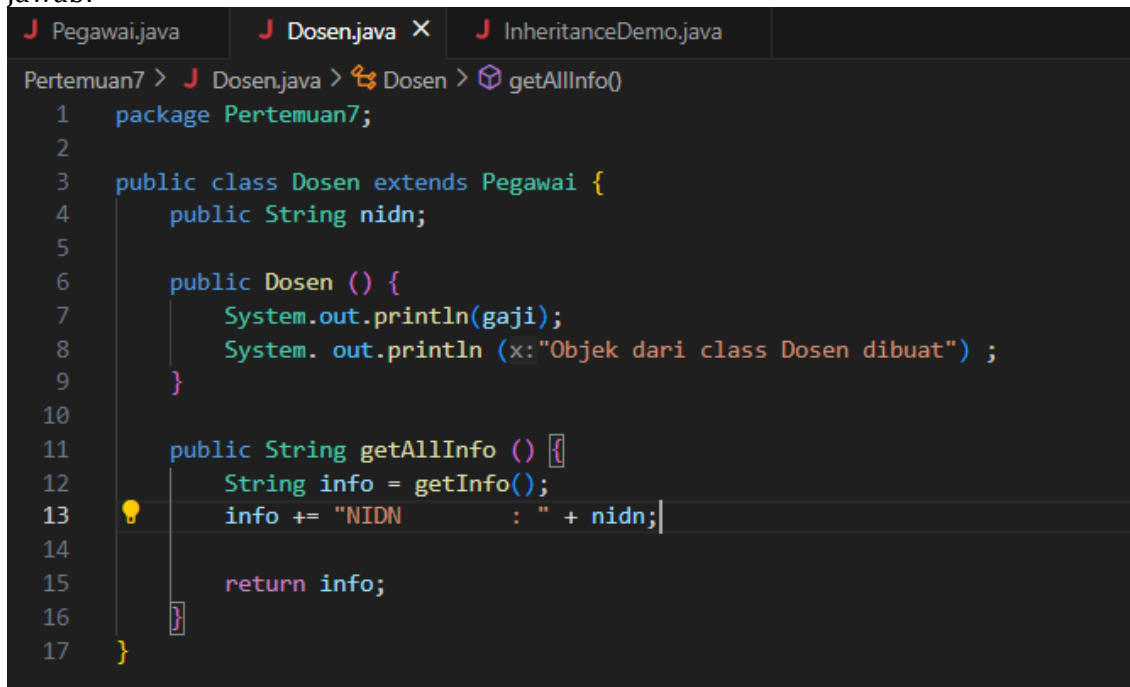
7. PERCOBAAN 5 (super & overriding)

A. TAHAPAN PERCOBAAN

1. Lakukan modifikasi kembali pada method `getAllInfo()`. Run program kemudian amati hasilnya

```
public String getAllInfo() {  
    String info = getInfo();  
    info += "NIDN      : " + nidn;  
  
    return info;  
}
```

Jawab:



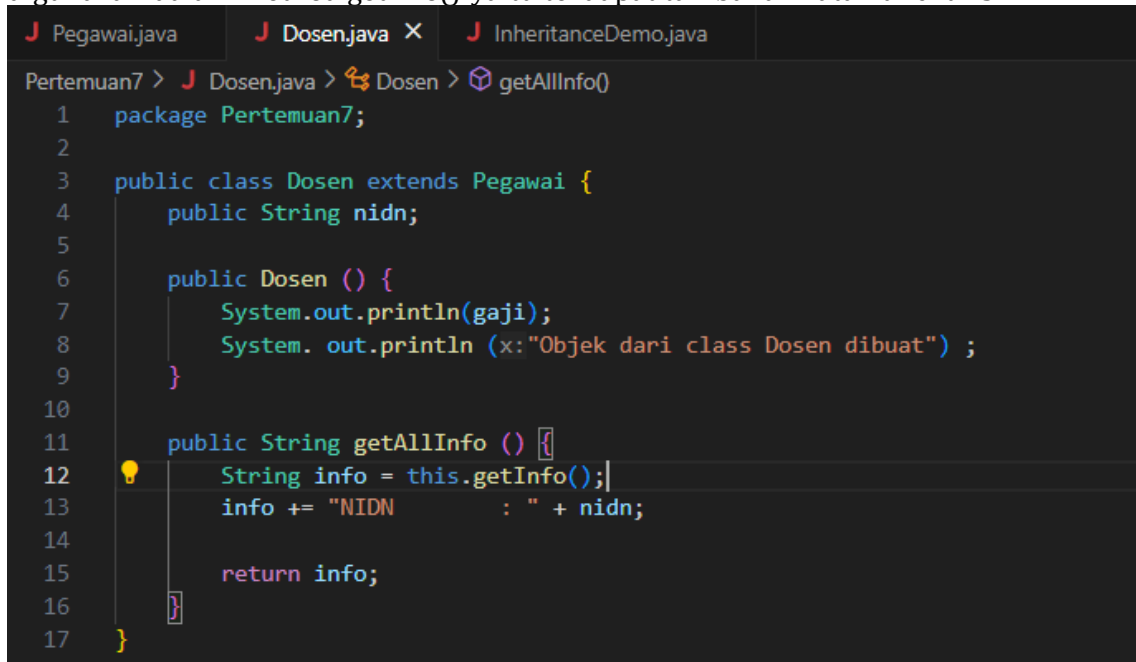
```
Pertemuan7 > J Dosen.java > Dosen > getAllInfo()  
1  package Pertemuan7;  
2  
3  public class Dosen extends Pegawai {  
4      public String nidn;  
5  
6      public Dosen () {  
7          System.out.println(gaji);  
8          System.out.println (x:"Objek dari class Dosen dibuat") ;  
9      }  
10  
11     public String getAllInfo () {  
12         String info = getInfo();  
13         info += "NIDN      : " + nidn;  
14  
15         return info;  
16     }  
17 }
```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe'  
ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redhat.  
itanceDemo'  
Objek dari class Pegawai dibuat  
0.0  
Objek dari class Dosen dibuat  
NIP      : 34329837  
Nama     : Yansy Ayuningtyas  
Gaji     : 3000000.0  
NIDN     : 1989432439  
PS D:\PBO>
```

2. Lakukan modifikasi kembali pada method `getAllInfo()`. Run program kemudian amati hasilnya

```
public String getAllInfo() {  
    String info = this.getInfo();  
    info += "NIDN          : " + nidn;  
  
    return info;  
}
```

Jawab: Output yang dihasilkan tetap sama, yang membedakan hanya kode yang digunakan dalam method `getInfo()` yaitu terdapat tambahan kata kunci `this`.



```
J Pegawai.java  J Dosen.java X  J InheritanceDemo.java  
Pertemuan7 > J Dosen.java > Dosen > getAllInfo()  
1  package Pertemuan7;  
2  
3  public class Dosen extends Pegawai {  
4      public String nidn;  
5  
6      public Dosen () {  
7          System.out.println(gaji);  
8          System.out.println(x:"Objek dari class Dosen dibuat") ;  
9      }  
10  
11     public String getAllInfo () {  
12         String info = this.getInfo();  
13         info += "NIDN          : " + nidn;  
14  
15         return info;  
16     }  
17 }
```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' %*  
ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\re  
itanceDemo'  
Objek dari class Pegawai dibuat  
0.0  
Objek dari class Dosen dibuat  
NIP          : 34329837  
Nama         : Yansy Ayuningtyas  
Gaji         : 3000000.0  
NIDN         : 1989432439  
PS D:\PBO>
```

3. Lakukan modifikasi kembali pada method `getAllInfo()`. Run program kemudian amati hasilnya

```
public String getAllInfo() {  
    String info = super.getInfo();  
    info += "NIDN          : " + nidn;  
  
    return info;  
}
```

Jawab: Hasilnya tetap sama, yang membedakan yaitu kata kunci `super`.

```
J Pegawai.java    J Dosen.java X    J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > getAllInfo()
1  package Pertemuan7;
2
3  public class Dosen extends Pegawai {
4      public String nidn;
5
6      public Dosen () {
7          System.out.println(gaji);
8          System.out.println (x:"Objek dari class Dosen dibuat") ;
9      }
10
11     public String getAllInfo () {
12         String info = super.getInfo();
13         info += "NIDN      : " + nidn;
14
15         return info;
16     }
17 }
```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' -Xmx1024m -Xms128m -Duser.dir=D:\PBO -classpath D:\PBO\src InheritanceDemo
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439
PS D:\PBO>
```

4. Tambahkan method `getInfo()` pada class `Dosen` dan modifikasi method `getAllInfo()` sebagai berikut

```
public class Dosen extends Pegawai {
    public String nidn;

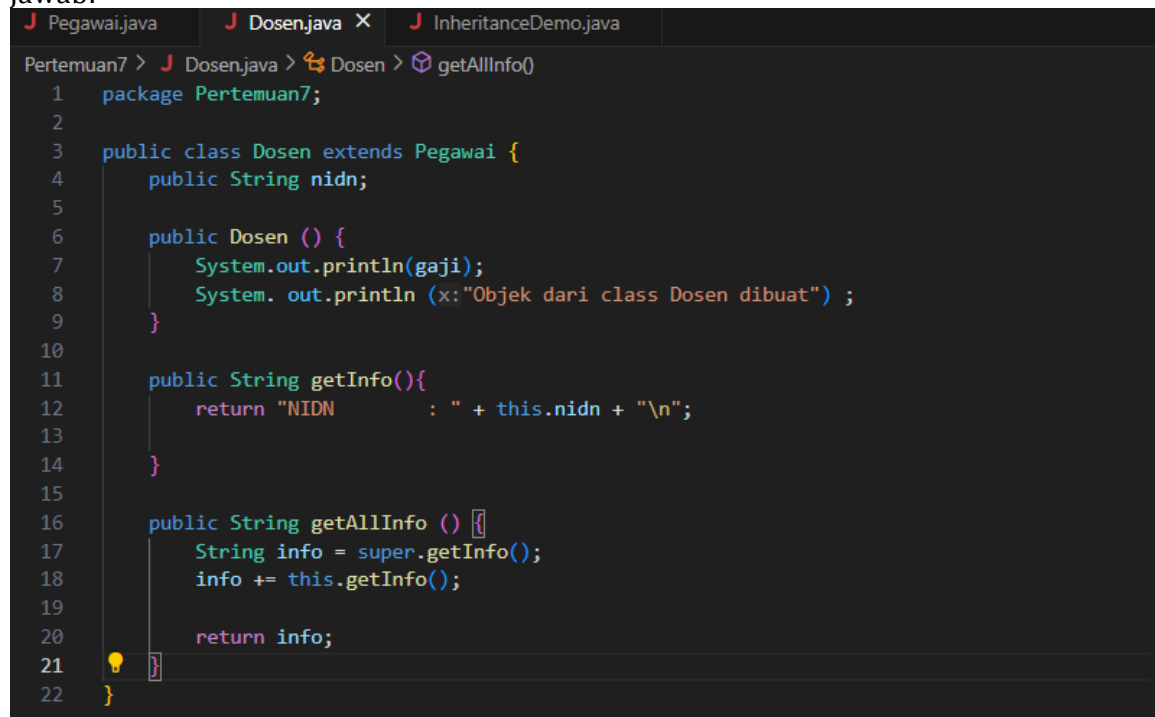
    public Dosen() {
        System.out.println("Objek dari class Dosen dibuat");
    }

    public String getInfo(){
        return "NIDN      : " + this.nidn + "\n";
    }

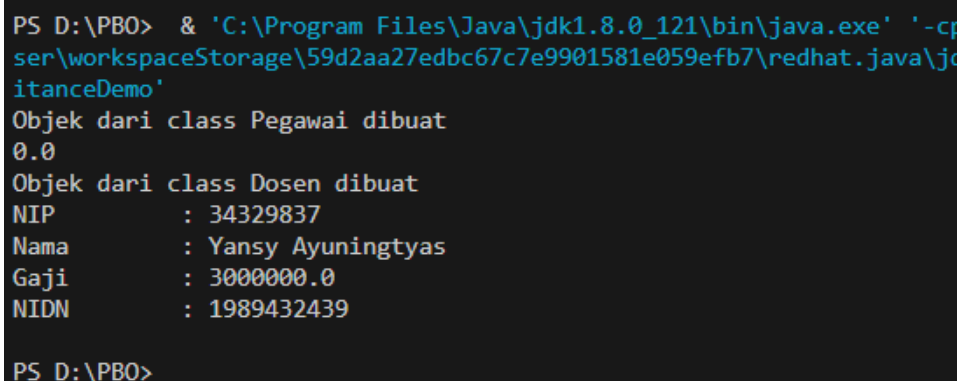
    public String getAllInfo(){
        String info = super.getInfo();
        info += this.getInfo();

        return info;
    }
}
```

Jawab:



```
J Pegawai.java  J Dosen.java X  J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > getAllInfo()
1  package Pertemuan7;
2
3  public class Dosen extends Pegawai {
4      public String nidn;
5
6      public Dosen () {
7          System.out.println(gaji);
8          System.out.println (x:"Objek dari class Dosen dibuat") ;
9      }
10
11     public String getInfo(){
12         return "NIDN      : " + this.nidn + "\n";
13     }
14
15     public String getAllInfo () {
16         String info = super.getInfo();
17         info += this.getInfo();
18
19         return info;
20     }
21 }
22 }
```



```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp
ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redhat.java\j
itanceDemo'
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

PS D:\PBO>
```

B. PERTANYAAN

1. Apakah ada perbedaan method `getInfo()` yang diakses pada langkah 1, 2, dan 3?
Jawab: Tidak ada karena menghasilkan output yang sama, karena method `getInfo()` tersebut telah ditambahkan kedalam method `getAllInfo()` didalam class `Dosen`

sehingga baik tidak menggunakan kata kunci, maupun menggunakan `this`, maupun menggunakan `super` akan tetap menghasilkan yang sama. Karena telah diwariskan oleh parent class. Sedangkan untuk penjelasan pada kode disetiap langkah yaitu

- Langkah 1, method `getAllInfo()` mengakses method `getInfo()` tanpa menggunakan kata kunci apapun. Itu berarti menginisiasi method `getInfo()` dengan method yang ada di parent class dan pada parent class ada method `getInfo()` juga maka akan langsung diwariskan pada child class
- Langkah 2, method `getAllInfo()` menggunakan `this` untuk mengakses method `getInfo()`. Dimana hal itu merujuk pada atribut di child class, jika di child class tidak ada `getInfo()` maka mencari terdekat yaitu di parent class.
- Langkah 3, method `getAllInfo()` menggunakan `super` untuk mengakses method `getInfo()`. Dimana hal itu langsung merujuk pada `getInfo()` yang ada di parent class.

2. Apakah ada perbedaan method `super.getInfo()` dan `this.getInfo()` yang dipanggil dalam method `getAllInfo()` pada langkah 4? Jelaskan!

Jawab: Pada method `super.getInfo()` yang dipanggil adalah method yang ada di parent classnya. Hal ini digunakan ketika method tersebut telah dioverride di class Dosen. Sedangkan untuk `this.getInfo()` yang dipanggil adalah method yang berada pada class Dosen itu sendiri.

3. Pada method manakah terjadi overriding? Jelaskan!

Jawab: Overriding terjadi pada method `getInfo()`. Ini karena class Dosen memiliki implementasi yang berbeda untuk method `getInfo()` daripada yang ada di parent class Pegawai. Ketika sebuah method dalam subclass memiliki nama, parameter, dan tipe kembalian yang sama dengan method di superclassnya, maka method tersebut dianggap overriding. Dalam hal ini, method `getInfo()` di class Dosen telah menggantikan method `getInfo()` yang ada di class Pegawai.

4. Tambahkan keyword `final` pada method `getInfo()` di class Pegawai. Apakah program dapat dicompile? Mengapa?

Jawab: Tidak, karena ketika sebuah method telah dinyatakan sebagai `final` artinya method tersebut tidak dapat di-override oleh child classnya.

```
J Pegawai.java X J Dosen.java 1 J InheritanceDemo.java
Pertemuan7 > J Pegawai.java > Pegawai > getInfo()
1 package Pertemuan7;
2
3 public class Pegawai {
4     public String nip;
5     public String nama;
6     public double gaji;
7
8     public Pegawai () {
9         System.out.println (x:"Objek dari class Pegawai dibuat") ;
10    }
11
12    public Pegawai(String nip, String nama, double gaji){
13        this.nip = nip;
14        this.nama = nama;
15        this.gaji = gaji;
16    }
17
18    public final String getInfo (){
19        String info = "";
20        info += "NIP      : " + nip + "\n";
21        info += "Nama      : " + nama + "\n";
22        info += "Gaji      : " + gaji + "\n";
23
24        return info;
25    }
26 }
```

```
J Pegawai.java J Dosen.java 1 X J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > getInfo()
1 package Pertemuan7;
2
3 public class Dosen extends Pegawai {
4     public String nidn;
5
6     public Dosen () {
7         System.out.println(gaji);
8         System.out.println (x:"Objek dari class Dosen dibuat") ;
9     }
10
11     public String getInfo(){
12         return "NIDN      : " + this.nidn + "\n";
13     }
14
15
16     public String getAllInfo () {
17         String info = super.getInfo();
18         info += "NIDN      : " + nidn;
19
20         return info;
21     }
22 }
```



```

PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp' 'C:\Users\WITRO\AppData
ser\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\bin' '
itanceDemo'
Exception in thread "main" java.lang.VerifyError: class Pertemuan7.Dosen overrides final met
ava/lang/String;
    at java.lang.ClassLoader.defineClass1(Native Method)
    at java.lang.ClassLoader.defineClass(ClassLoader.java:763)
    at java.security.SecureClassLoader.defineClass(SecureClassLoader.java:142)
    at java.net.URLClassLoader.defineClass(URLClassLoader.java:467)
    at java.net.URLClassLoader.access$100(URLClassLoader.java:73)
    at java.net.URLClassLoader$1.run(URLClassLoader.java:368)
    at java.net.URLClassLoader$1.run(URLClassLoader.java:362)
    at java.security.AccessController.doPrivileged(Native Method)
    at java.net.URLClassLoader.findClass(URLClassLoader.java:361)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:424)
    at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:331)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:357)
    at Pertemuan7.InheritanceDemo.main(InheritanceDemo.java:5)
PS D:\PBO>

```

8. PERCOBAAN 6 (overloading)

A. TAHAPAN PERCOBAAN

1. Tambahkan constructor baru untuk class Dosen sebagai berikut

```

public Dosen(String nip, String nama, double gaji, String nidn){
    System.out.print("Objek dari class Dosen dibuat dengan constructor berparameter");
}

```

Jawab:

```

J Pegawai.java  J Dosen.java X  J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > Dosen(String, String, double, String)
1 package Pertemuan7;
2
3 public class Dosen extends Pegawai {
4     public String nidn;
5
6     public Dosen(String nip, String nama, double gaji, String nidn){
7         System.out.println(x:"Objek dari class Dosen dibuat dengan constructor berparameter");
8     }
9
10    public Dosen () {
11        System.out.println(gaji);
12        System.out.println (x:"Objek dari class Dosen dibuat") ;
13    }
14
15    public String getInfo(){
16        return "NIDN      : " + this.nidn + "\n";
17    }
18
19
20    public String getAllInfo () {
21        String info = super.getInfo();
22        info += this.getInfo();
23
24        return info;
25    }
26 }

```

2. Modifikasi class InheritanceDemo untuk menginstansiasi object baru dengan nama dosen2 dengan constructor yang berparameter. Run program kemudian amati hasilnya.

```

public static void main(String[] args) {
    Dosen dosen2 = new Dosen("34329837", "Yansy Ayuningtyas", 3000000, "1989432439");
    System.out.println(dosen2.getAllInfo());
}

```

Jawab:

```
J Pegawai.java J Dosen.java J InheritanceDemo.java X
Pertemuan7 > J InheritanceDemo.java > InheritanceDemo > main(String[])
1 package Pertemuan7;
2
3 public class InheritanceDemo {
4     Run | Debug
5     public static void main(String[] args) {
6         Dosen dosen1 = new Dosen();
7
8         dosen1.nama = "Yansy Ayuningtyas";
9         dosen1.nip = "34329837";
10        dosen1.gaji = 3000000;
11        dosen1.nidn = "1989432439";
12
13        System.out.println(dosen1.getAllInfo());
14
15        Dosen dosen2 = new Dosen(nip:"34329837", nama:"Yansy Ayuningtyas", gaji:3000000, nidn:"1989432439");
16        System.out.println(dosen2.getAllInfo());
17    }
```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp' 'C:\Users\NITRO\AppData\Local\Temp\7edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\bin' 'Pertemuan7.InheritanceDemo'
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat dengan constructor berparameter
NIP      : null
Nama     : null
Gaji     : 0.0
NIDN     : null

PS D:\PBO>
```

B. PERTANYAAN

1. Bagaimana hasil nilai nip, nama, gaji, dan nidn yang ditampilkan pada langkah 2? Mengapa demikian?

Jawab: Output yang dikeluarkan nip, nama, gaji dan nidn adalah null. Karena nilai parameter tidak dimasukkan ke atribut nip, nama, gaji dan nidn.

2. Jelaskan apakah constructor tanpa parameter dan constructor class Dosen yang dibuat padalangkah 1 memiliki signature yang sama?

Jawab: Tidak sama, karena constructor tanpa parameter dan constructor class Dosen memiliki signature yang berbeda. Dimana memiliki jumlah parameter yang berbeda, yang satunya tidak ada parameter, yang satunya ada.

3. Konsep apa dalam OOP yang membolehkan suatu class memiliki constructor atau method dengan nama yang sama dan signature yang berbeda pada satu class?

Jawab: Overloading.

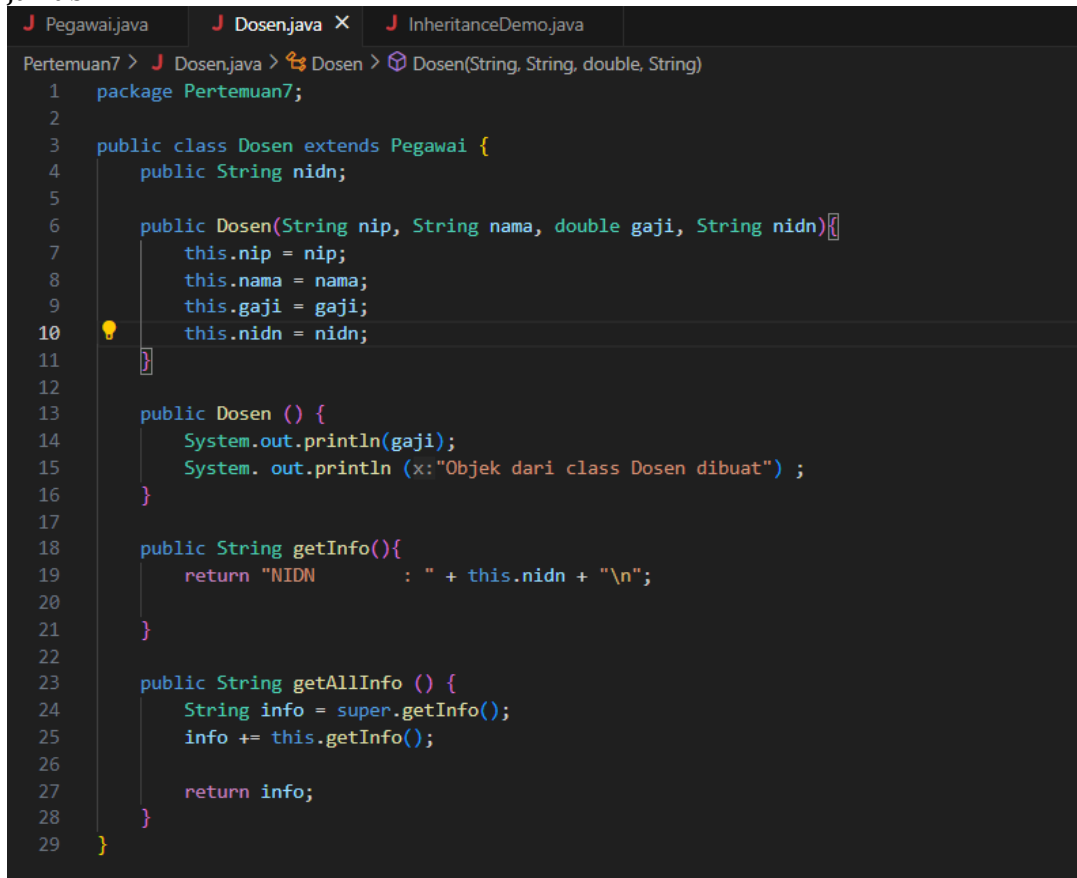
9. PERCOBAAN 7 (super - constructor)

A. TAHAPAN PERCOBAAN

1. Modifikasi constructor pada class Dosen sebagai berikut. Run program kemudian amati hasilnya.

```
public Dosen(String nip, String nama, double gaji, String nidn){
    this.nip = nip;
    this.nama = nama;
    this.gaji = gaji;
    this.nidn = nidn;
}
```

Jawab:



```
1 package Pertemuan7;
2
3 public class Dosen extends Pegawai {
4     public String nidn;
5
6     public Dosen(String nip, String nama, double gaji, String nidn){
7         this.nip = nip;
8         this.nama = nama;
9         this.gaji = gaji;
10        this.nidn = nidn;
11    }
12
13    public Dosen () {
14        System.out.println(gaji);
15        System.out.println (x:"Objek dari class Dosen dibuat") ;
16    }
17
18    public String getInfo(){
19        return "NIDN      : " + this.nidn + "\n";
20    }
21
22    public String getAllInfo () {
23        String info = super.getInfo();
24        info += this.getInfo();
25
26        return info;
27    }
28 }
29
```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp'
7edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\bin' 'Pertem
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

Objek dari class Pegawai dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

PS D:\PBO>
```

2. Modifikasi constructor pada class Dosen sebagai berikut. Run program kemudian amati hasilnya.

```
public Dosen(String nip, String nama, double gaji, String nidn){
    super.nip = nip;
    super.nama = nama;
    super.gaji = gaji;
    this.nidn = nidn;
}
```

Jawab:

```
J Pegawai.java X Dosen.java X InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > Dosen(String, String, double, String)
1 package Pertemuan7;
2
3 public class Dosen extends Pegawai {
4     public String nidn;
5
6     public Dosen(String nip, String nama, double gaji, String nidn){
7         super.nip = nip;
8         super.nama = nama;
9         super.gaji = gaji;
10        this.nidn = nidn;
11    }
12
13    public Dosen () {
14        System.out.println(gaji);
15        System.out.println (x:"Objek dari class Dosen dibuat") ;
16    }
17
18    public String getInfo(){
19        return "NIDN : " + this.nidn + "\n";
20    }
21
22    public String getAllInfo () {
23        String info = super.getInfo();
24        info += this.getInfo();
25
26        return info;
27    }
28
29 }
```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp' 'C:\Users\NI
7edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\bin' 'Pertemuan7.Inherit
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

Objek dari class Pegawai dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

PS D:\PBO>
```

3. Modifikasi constructor pada class Dosen sebagai berikut. Run program kemudian amati hasilnya.

```
public Dosen(String nip, String nama, double gaji, String nidn){
    super();
    super.nip = nip;
    super.nama = nama;
    super.gaji = gaji;
    this.nidn = nidn;
}
```

Jawab:

```
Pertemuan7 > J Dosen.java X InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > Dosen(String, String, double, String)
1 package Pertemuan7;
2
3 public class Dosen extends Pegawai {
4     public String nidn;
5
6     public Dosen(String nip, String nama, double gaji, String nidn){
7         super();
8         super.nip = nip;
9         super.nama = nama;
10        super.gaji = gaji;
11        this.nidn = nidn;
12    }
13
14    public Dosen () {
15        System.out.println(gaji);
16        System.out.println (x:"Objek dari class Dosen dibuat" );
17    }
18
19    public String getInfo(){
20        return "NIDN      : " + this.nidn + "\n";
21    }
22
23    public String getAllInfo () {
24        String info = super.getInfo();
25        info += this.getInfo();
26
27        return info;
28    }
29 }
30 }
```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp' 'C:\7edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\bin' 'Pertemuan7
Objek dari class Pegawai dibuat
0.0
Objek dari class Dosen dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

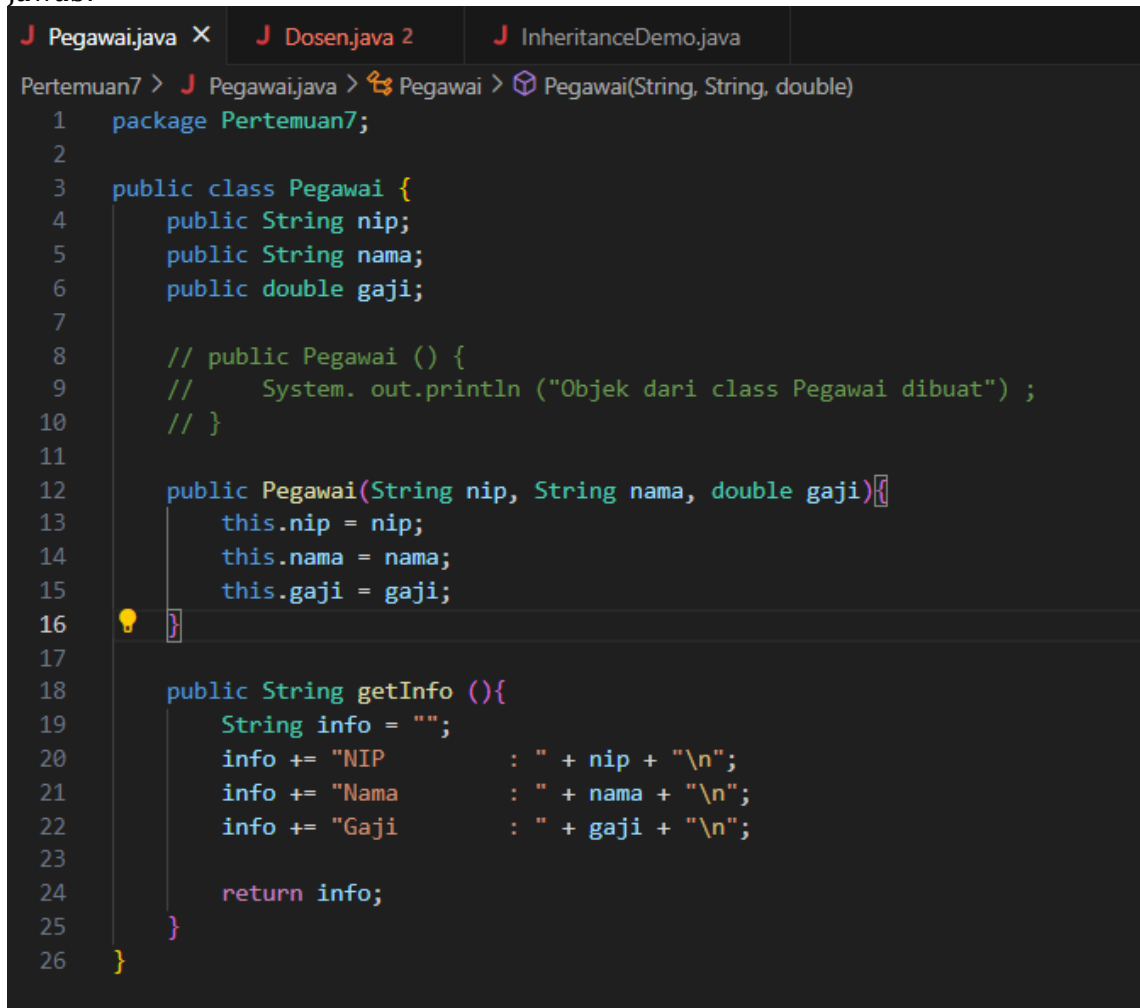
Objek dari class Pegawai dibuat
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

PS D:\PBO>
```

4. Hapus/comment constructor tanpa parameter dari class Pegawai. Tambahkan constructor baru untuk class Pegawai sebagai berikut. Run program kemudian amati hasilnya.

```
public class Pegawai {  
    public String nip;  
    public String nama;  
    public double gaji;  
  
    //    public Pegawai() {  
    //        System.out.println("Objek dari class Pegawai dibuat");  
    //    }  
  
    public Pegawai(String nip, String nama, double gaji) {  
        this.nip = nip;  
        this.nama = nama;  
        this.gaji = gaji;  
    }  
  
    public String getInfo(){  
        String info = "";  
        info += "NIP        : " + nip + "\n";  
        info += "Nama        : " + nama + "\n";  
        info += "Gaji        : " + gaji + "\n";  
  
        return info;  
    }  
}
```

Jawab:



```
J Pegawai.java X  J Dosen.java 2  J InheritanceDemo.java  
Pertemuan7 > J Pegawai.java > Pegawai > Pegawai(String, String, double)  
1  package Pertemuan7;  
2  
3  public class Pegawai {  
4      public String nip;  
5      public String nama;  
6      public double gaji;  
7  
8      // public Pegawai () {  
9      //     System.out.println ("Objek dari class Pegawai dibuat") ;  
10     // }  
11  
12     public Pegawai(String nip, String nama, double gaji){  
13         this.nip = nip;  
14         this.nama = nama;  
15         this.gaji = gaji;  
16     }  
17  
18     public String getInfo (){  
19         String info = "";  
20         info += "NIP        : " + nip + "\n";  
21         info += "Nama        : " + nama + "\n";  
22         info += "Gaji        : " + gaji + "\n";  
23  
24         return info;  
25     }  
26 }
```

```
Pertemuan7 > J Dosen.java > Dosen > getInfo()
1 package Pertemuan7;
2
3 public class Dosen extends Pegawai {
4     public String nidn;
5
6     public Dosen(String nip, String nama, double gaji, String nidn){
7         super.nip = nip;
8         super.nama = nama;
9         super.gaji = gaji;
10        this.nidn = nidn;
11    }
12
13    // public Dosen () {
14    //     System.out.println(gaji);
15    //     System.out.println("Objek dari class Dosen dibuat");
16    // }
17
18    public String getInfo(){
19        return "NIDN : " + this.nidn + "\n";
20    }
21
22    public String getAllInfo () {
23        String info = super.getInfo();
24        info += this.getInfo();
25
26        return info;
27    }
28 }
29 }
```

```
Pertemuan7 > J InheritanceDemo.java > InheritanceDemo
1 package Pertemuan7;
2
3 public class InheritanceDemo {
4     public static void main(String[] args) {
5         // Dosen dosen1 = new Dosen();
6
7         // dosen1.nama = "Yansy Ayuningtyas";
8         // dosen1.nip = "34329837";
9         // dosen1.gaji = 3000000;
10        // dosen1.nidn = "1989432439";
11
12        // System.out.println(dosen1.getAllInfo());
13
14        Dosen dosen2 = new Dosen(nip:"34329837", nama:"Yansy Ayuningtyas", gaji:3000000, nidn:"1989432439");
15        System.out.println(dosen2.getAllInfo());
16    }
17 }
```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp' 'C:\Users\NITRO\AppData\Roaming\Code\workspaceStorage\59d2aa27edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\bin' 'Pertemuan7.InheritanceDemo'
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
    Implicit super constructor Pegawai() is undefined. Must explicitly invoke another constructor

    at Pertemuan7.Dosen.<init>(Dosen.java:6)
    at Pertemuan7.InheritanceDemo.main(InheritanceDemo.java:14)
PS D:\PBO>
```

5. Modifikasi constructor pada class Dosen sebagai berikut. Run program kemudian amati hasilnya.

```
public Dosen(String nip, String nama, double gaji, String nidn){
    this.nidn = nidn;
    super(nip, nama, gaji);
}
```

Jawab:

```
J Pegawai.java J Dosen.java 2 X J InheritanceDemo.java
Pertemuan7 > J Dosen.java > Dosen > Dosen(String, String, double, String)
1 package Pertemuan7;
2
3 public class Dosen extends Pegawai {
4     public String nidn;
5
6     public Dosen(String nip, String nama, double gaji, String nidn){
7         this.nidn = nidn;
8         super(nip, nama, gaji);
9     }
10
11     // public Dosen () {
12     //     System.out.println(gaji);
13     //     System.out.println ("Objek dari class Dosen dibuat") ;
14     // }
15
16     public String getInfo(){
17         return "NIDN : " + this.nidn + "\n";
18     }
19
20
21     public String getAllInfo () {
22         String info = super.getInfo();
23         info += this.getInfo();
24
25         return info;
26     }
27 }
```

```
J Pegawai.java J Dosen.java 2 J InheritanceDemo.java X
Pertemuan7 > J InheritanceDemo.java > InheritanceDemo
1 package Pertemuan7;
2
3 public class InheritanceDemo {
4     public static void main(String[] args) {
5         // Dosen dosen1 = new Dosen();
6
7         // dosen1.nama = "Yansy Ayuningtyas";
8         // dosen1.nip = "34329837";
9         // dosen1.gaji = 3000000;
10        // dosen1.nidn = "1989432439";
11
12        // System.out.println(dosen1.getAllInfo());
13
14        Dosen dosen2 = new Dosen(nip:"34329837", nama:"Yansy Ayuningtyas", gaji:3000000, nidn:"1989432439");
15        System.out.println(dosen2.getAllInfo());
16    }
17 }
```

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp' 'C:\Users\NITRO\AppData\Roaming\Code\2aa27edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\bin' 'Pertemuan7.InheritanceDemo'
Exception in thread "main" java.lang.Error: Unresolved compilation problems:
    Implicit super constructor Pegawai() is undefined. Must explicitly invoke another constructor
    Constructor call must be the first statement in a constructor

    at Pertemuan7.Dosen.<init>(Dosen.java:6)
    at Pertemuan7.InheritanceDemo.main(InheritanceDemo.java:14)
PS D:\PBO>
```

6. Modifikasi constructor pada class Dosen sebagai berikut. Run program kemudian amati hasilnya.


```
public Dosen(String nip, String nama, double gaji, String nidn){
    super(nip, nama, gaji);
    this.nidn = nidn;
}
```

Jawab:

```

1 package Pertemuan7;
2
3 public class Dosen extends Pegawai {
4     public String nidn;
5
6     public Dosen(String nip, String nama, double gaji, String nidn){
7         super(nip, nama, gaji);
8         this.nidn = nidn;
9     }
10
11     // public Dosen () {
12     //     System.out.println(gaji);
13     //     System.out.println("Objek dari class Dosen dibuat") ;
14     // }
15
16     public String getInfo(){
17         return "NIDN : " + this.nidn + "\n";
18     }
19
20
21     public String getAllInfo () {
22         String info = super.getInfo();
23         info += this.getInfo();
24
25         return info;
26     }
27 }

```

```

PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe'
2aa27edbc67c7e9901581e059efb7\redhat.java\jdt_ws\PBO_77516f84\bin
NIP      : 34329837
Nama     : Yansy Ayuningtyas
Gaji     : 3000000.0
NIDN     : 1989432439

PS D:\PBO>

```

B. PERTANYAAN

1. Apakah terdapat perbedaan hasil pada langkah 1 dan 2? Jelaskan!
Jawab: Tidak, karena semua menampilkan atau output yang sama. Dimana pada langkah 1 menggunakan kata kunci this untuk memanggil atribut parent class dan juga atribut punya child class itu yaitu NIDN. Sedangkan langkah 2 menggunakan kata kunci kombinasi super yang akan langsung merujuk pada parent class, dan NIDNnya menggunakan kata kunci this dimana menggunakan atribut class itu sendiri.
2. Apakah terdapat perbedaan hasil pada langkah 2 dan 3? Jelaskan!
Jawab: Tidak, karena sama-sama menginisialisasi atribut parent class akan tetapi dilangkah 3 pemanggilan method constructor yaitu merujuk pada constructor parent class yaitu super().
3. Mengapa terjadi error pada langkah 4?
Jawab: Karena tidak ada constructor tanpa parameter yang tersedia di class Pegawai, sehingga class Dosen tidak dapat menemukan constructor default untuk parent class-

nya saat mencoba membuat objek dari class Dosen.

4. Apa perbedaan `super()` yang dipanggil pada langkah 3 dan 6?

Jawab: Pada langkah 3 `super` yang dipanggil yaitu constructor Pegawai tidak menggunakan parameter atau tanpa parameter, sedangkan `super` pada langkah 6 memanggil constructor Pegawai menggunakan parameter.

5. Mengapa terjadi error pada langkah 5?

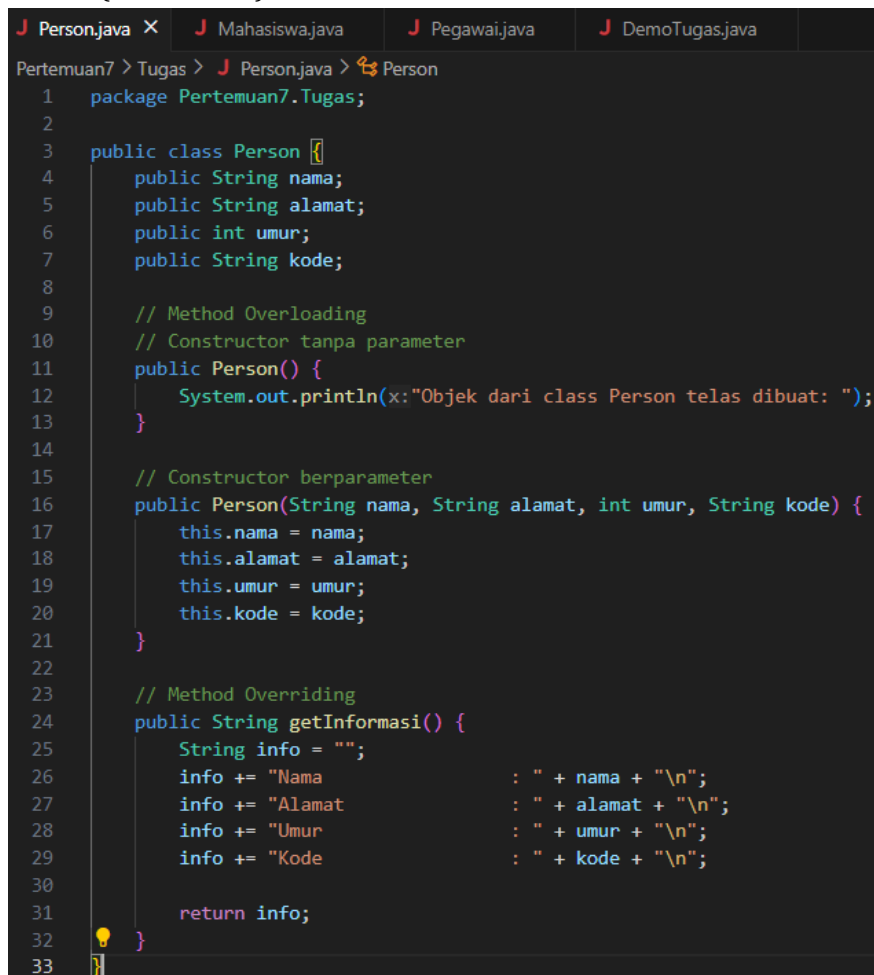
Jawab: Karena tata letak pemanggilan `super` harus selalu berada pada baris pertama dalam constructor child class sebelum statement lainnya, oleh karena itu pada langkah 5 akan terjadi error.

10. TUGAS

1. Tentukan sebuah class yang merupakan turunan dari class yang lain.
2. Buat 3 atribut pada parent class kemudian tambahkan minimal 1 atribut pada child class.
3. Lakukan method overloading dengan membuat 2 constructor yaitu constructor tanpa parameter dan constructor berparameter pada masing-masing class. Panggil constructor `super()` berparameter untuk membuat object dari parent class pada constructor child class.
4. Lakukan method overriding dengan membuat method dengan nama dan signature yang sama pada parent class dan child class.
5. Lakukan instansiasi objek child class pada main class kemudian print info nya.

Jawab:

- Person (Parent class)



```
Person.java X  Mahasiswa.java  Pegawai.java  DemoTugas.java
Pertemuan7 > Tugas > Person.java > Person
1  package Pertemuan7.Tugas;
2
3  public class Person {
4      public String nama;
5      public String alamat;
6      public int umur;
7      public String kode;
8
9      // Method Overloading
10     // Constructor tanpa parameter
11     public Person() {
12         System.out.println("Objek dari class Person telah dibuat: ");
13     }
14
15     // Constructor berparameter
16     public Person(String nama, String alamat, int umur, String kode) {
17         this.nama = nama;
18         this.alamat = alamat;
19         this.umur = umur;
20         this.kode = kode;
21     }
22
23     // Method Overriding
24     public String getInformasi() {
25         String info = "";
26         info += "Nama          : " + nama + "\n";
27         info += "Alamat         : " + alamat + "\n";
28         info += "Umur           : " + umur + "\n";
29         info += "Kode            : " + kode + "\n";
30
31         return info;
32     }
33 }
```

- Mahasiswa (child class)

```
J Person.java J Mahasiswajava X J Pegawai.java J DemoTugas.java
Pertemuan7 > Tugas > J Mahasiswajava > Mahasiswa
1 package Pertemuan7.Tugas;
2
3 class Mahasiswa extends Person {
4     public String programStudi;
5     public String jurusan;
6
7     // Method Overloading
8     // Constructor tanpa parameter
9     public Mahasiswa() {
10         System.out.println(x:"Objek dari class Mahasiswa telah dibuat: ");
11     }
12
13     // Constructor berparameter
14     public Mahasiswa(String nama, String alamat, int umur, String kode, String programStudi, String jurusan) {
15         super(nama, alamat, umur, kode);
16         this.programStudi = programStudi;
17         this.jurusan = jurusan;
18     }
19
20     public String getInformasi(){
21         String info = "";
22         info += "Program Studi      : " + programStudi + "\n";
23         info += "Jurusan          : " + jurusan + "\n";
24
25         return info;
26     }
27
28     // Method Overriding
29     public String getAllInformasi() {
30         String info = super.getInformasi();
31         info += this.getInformasi();
32
33         return info;
34     }
35 }
```

- Pegawai (child class)

```
J Person.java J Mahasiswajava J Pegawai.java X J DemoTugas.java
Pertemuan7 > Tugas > J Pegawai.java > Pegawai
1 package Pertemuan7.Tugas;
2
3 class Pegawai extends Person {
4     public int gaji;
5
6     // Method Overloading
7     // Constructor tanpa parameter
8     public Pegawai() {
9         System.out.println(x:"Objek dari class Pegawai telah dibuat: ");
10    }
11
12    // Constructor berparameter
13    public Pegawai(String nama, String alamat, int umur, String kode, int gaji) {
14        super(nama, alamat, umur, kode);
15        this.gaji = gaji;
16    }
17
18
19    public String getInformasi(){
20        return "Gaji          : " + this.gaji + "\n";
21    }
22
23
24    // Method Overriding
25    public String getAllInformasi() {
26        String info = super.getInformasi();
27        info += this.getInformasi();
28
29        return info;
30    }
31 }
```

- DemoTugas

```
Person.java Mahasiswajava Pegawajava DemoTugas.java X
Pertemuan7 > Tugas > DemoTugas.java > DemoTugas
1 package Pertemuan7.Tugas;
2
3 public class DemoTugas {
4     Run/Debug
5     public static void main(String[] args) {
6         // Membuat objek Mahasiswa tanpa parameter
7         Mahasiswa mahasiswa1 = new Mahasiswa();
8         mahasiswa1.nama = "Mutia Rengganis";
9         mahasiswa1.alamat = "Dsn. Tapen, Ds. Tapen, Kec. Kudu, Kab. Jombang";
10        mahasiswa1.umur = 20;
11        mahasiswa1.kode = "2241760086";
12        mahasiswa1.programStudi = "D - IV Sisten Informasi Bisnis";
13        mahasiswa1.jurusan = "Teknologi Informasi";
14        System.out.println("=====");
15        // Menampilkan informasi mahasiswa
16        System.out.println("Informasi Mahasiswa");
17        System.out.println("=====");
18        System.out.println(mahasiswa1.getAllInformasi());
19
20        // Membuat objek Mahasiswa
21        Mahasiswa mahasiswa2 = new Mahasiswa("Oddis Nur Alifathur Razaq", alamat:"Dsn. Mojoyanti, Ds. Jatibanjat, Kec. Ploso, Kab. Jombang", umur:19, kode:"2241760015", programStudi:"D - IV Sisten Informasi Bisnis", jurusan:"Teknologi Informasi");
22        System.out.println("=====");
23        // Menampilkan informasi mahasiswa
24        System.out.println("Informasi Mahasiswa");
25        System.out.println("=====");
26        System.out.println(mahasiswa2.getAllInformasi());
27
28        // Membuat objek Pegawai tanpa parameter
29        Pegawai pegawai1 = new Pegawai();
30        pegawai1.nama = "Arya Bima Putra Dewangga";
31        pegawai1.alamat = "Dsn. Cipir Dondong, Ds. Banjardowo, Kec. Kabuh, Kab. Jombang";
32        pegawai1.umur = 20;
33        pegawai1.kode = "20030312201501";
34        pegawai1.gaji = 20000000;
35        System.out.println("=====");
36        // Menampilkan informasi pegawai
37        System.out.println("Informasi Pegawai");
38        System.out.println("=====");
39        System.out.println(pegawai1.getAllInformasi());
40
41        // Membuat objek Pegawai
42        Pegawai pegawai2 = new Pegawai("Dedy Setia Nanda", alamat:"Dsn. Mojoyanti, Ds. Jatibanjat, Kec. Ploso, Kab. Jombang", umur:26, kode:"19960312201501", gaji:19000000);
43        System.out.println("=====");
44        System.out.println(pegawai2.getAllInformasi());
45    }
46 }
```

- Hasil Run

```
PS D:\PBO> & 'C:\Program Files\Java\jdk1.8.0_121\bin\java.exe' '-cp' 'C:\Users\WITRO\AppData\Roaming\Code\User\workspaceStorage\59d2aa27edbc67
Objek dari class Person telah dibuat:
Objek dari class Mahasiswa telah dibuat:

=====

Informasi Mahasiswa

=====

Nama          : Mutia Rengganis
Alamat        : Dsn. Tapen, Ds. Tapen, Kec. Kudu, Kab. Jombang
Umur          : 20
Kode          : 2241760086
Program Studi : D - IV Sistem Informasi Bisnis
Jurusan       : Teknologi Informasi

=====

Nama          : Oddis Nur Alifathur Razaq
Alamat        : Dsn. Mojoyanti, Ds. Jatibanjat, Kec. Ploso, Kab. Jombang
Umur          : 19
Kode          : 2241760015
Program Studi : D - IV Sistem Informasi Bisnis
Jurusan       : Teknologi Informasi

=====

Objek dari class Person telah dibuat:
Objek dari class Pegawai telah dibuat:

=====

Informasi Pegawai

=====

Nama          : Arya Bima Putra Dewangga
Alamat        : Dsn. Cipir Dondong, Ds. Banjardowo, Kec. Kabuh, Kab. Jombang
Umur          : 20
Kode          : 20030312201501
Gaji          : 20000000

=====

Nama          : Dedy Setia Nanda
Alamat        : Dsn. Mojoyanti, Ds. Jatibanjat, Kec. Ploso, Kab. Jombang
Umur          : 26
Kode          : 19960312201501
Gaji          : 19000000

=====

PS D:\PBO>
```