

**LAPORAN RESMI**  
**MODUL III**  
**INHERINTANCE & OVERRIDE**  
**PEMROGRAMAN BERBASIS OBJEK**



<b>NAMA</b>	<b>: ABAS PERMADANI</b>
<b>N.R.P</b>	<b>: 230441100107</b>
<b>DOSEN</b>	<b>: ACHMAD ZAIN NUR, S.KOM., M.T.</b>
<b>ASISTEN</b>	<b>: MUHAMMAD ROSYID MAULANA</b>
<b>TGL PRAKTIKUM</b>	<b>: 03 MEI 2024</b>

**Disetujui : 27 MEI 2024**  
**Asisten**

**MUHAMMAD ZAIN MAULANA**  
**20.04.411.00019**



**LABORATORIUM BISNIS INTELIJEN SISTEM**  
**PRODI SISTEM INFORMASI**  
**JURUSAN TEKNIK INFORMATIKA**  
**FAKULTAS TEKNIK**  
**UNIVERSITAS TRUNOJOYO MADURA**

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Dalam pemrograman berorientasi objek (OOP), "Inheritance" (Pewarisan) dan "Override" (Penimpaan) merupakan dua konsep kunci yang memungkinkan pemrogram untuk membuat software yang lebih modular, efisien, dan mudah untuk dikelola

**Inheritance (Pewarisan) Reuse Kode:** Pewarisan memungkinkan developers untuk menggunakan kembali kode yang sudah ada. Sebagai contoh, jika Anda memiliki class yang mendefinisikan karakteristik dasar dari semua kendaraan, Anda bisa mewariskan class ini untuk membuat class yang lebih spesifik seperti mobil atau sepeda motor tanpa harus menulis ulang semua kode yang sama. **Strukturisasi Kode:** Dengan pewarisan, kita bisa mengatur kode dalam hierarki yang jelas, di mana class dasar menyediakan fitur dasar, dan class turunan bisa menambah atau mengkustomisasi fitur tersebut. Hal ini memudahkan dalam pemahaman dan pengelolaan kode. **Abstraksi:** Pewarisan memungkinkan pemrogram untuk bekerja pada tingkat abstraksi yang lebih tinggi. Developer bisa menangani objek pada tingkat umum dalam hierarki, dan membiarkan detail implementasi pada subclass.

### **1.2 Tujuan**

- Mahasiswa mampu memahami konsep Inheritance atau pewarisan dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.
- Mahasiswa mampu memahami konsep override dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.

## BAB II

### DASAR TEORI

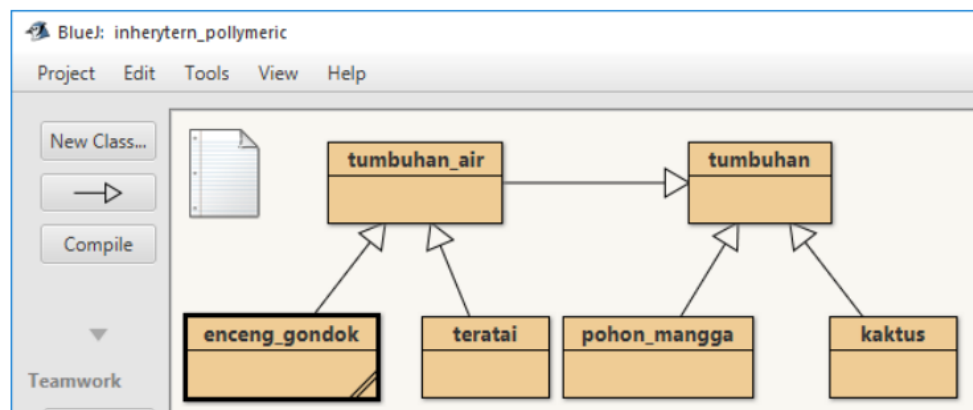
#### 2.1 Pengertian inheritance :

Inheritance adalah salah satu konsep dasar dalam java. Inheritance memiliki arti yaitu warisan jadi dalam java ada class yang akan mewariskan method atau fungsi dan ada yang akan menerima warisan itu. Pemberi warisan disebut superclass sedangkan untuk penerimanya disebut subclass, atau biasa juga disebut sebagai induk dan anak class.

#### 2.2 Pengertian override :

Override memiliki artian yaitu menimpa, dalam java method superclass dapat ditimpa oleh subclass dengan cara menuliskan lagi nama method yang sama pada subclass. Override biasanya dilakukan jika method superclassnya tidak memiliki makna yang sama atau isi dari method superclass harus dirubah.

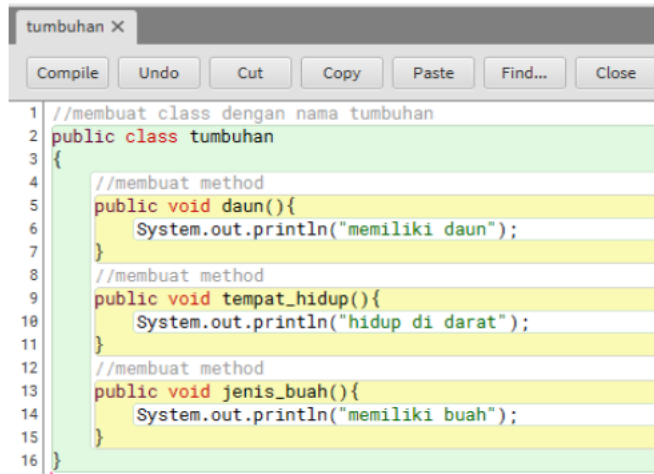
#### Contoh program :



Class program :

- Class tumbuhan sebagai superclass semua method awal dituliskan disini
- Class tumbuhan memiliki subclass yaitu pohon\_mangga, dan kaktus.
- Tumbuhan air memiliki subclass yaitu enceng\_gondok, dan teratai

Class tumbuhan :



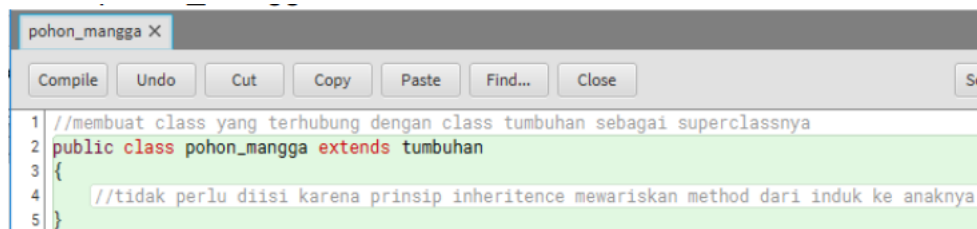
```

1 //membuat class dengan nama tumbuhan
2 public class tumbuhan
3 {
4     //membuat method
5     public void daun(){
6         System.out.println("memiliki daun");
7     }
8     //membuat method
9     public void tempat_hidup(){
10        System.out.println("hidup di darat");
11    }
12    //membuat method
13    public void jenis_buah(){
14        System.out.println("memiliki buah");
15    }
16 }

```

Class tumbuhan membuat 3 method yang nantinya akan diturunkan ke setiap subclassnya.

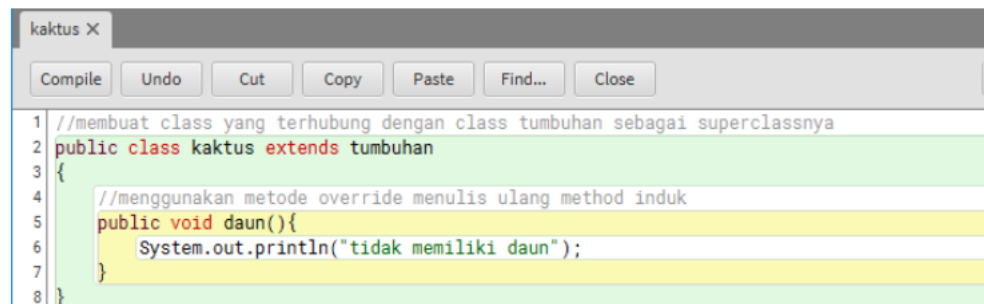
Class pohon\_mangga dan kaktus :



```

1 //membuat class yang terhubung dengan class tumbuhan sebagai superclassnya
2 public class pohon_mangga extends tumbuhan
3 {
4     //tidak perlu diisi karena prinsip inheritance mewariskan method dari induk ke anaknya
5 }

```



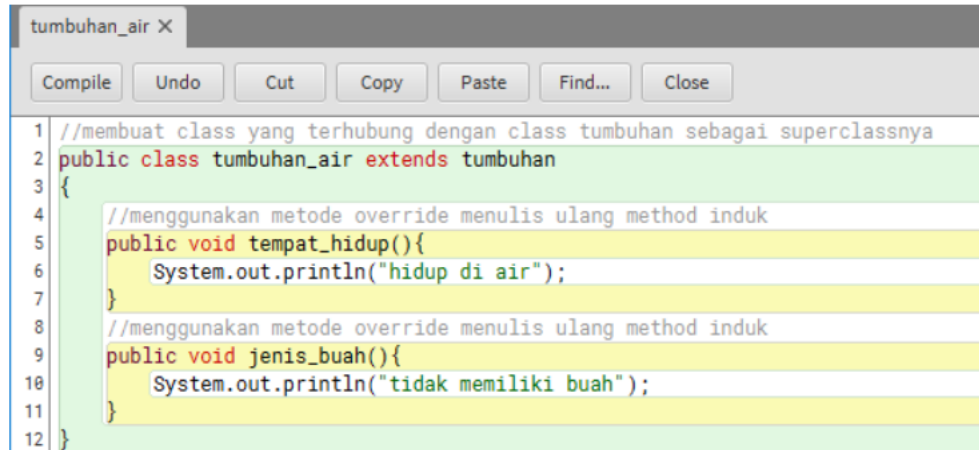
```

1 //membuat class yang terhubung dengan class tumbuhan sebagai superclassnya
2 public class kaktus extends tumbuhan
3 {
4     //menggunakan metode override menulis ulang method induk
5     public void daun(){
6         System.out.println("tidak memiliki daun");
7     }
8 }

```

Class pohon\_mangga menerima semua warisan dari class tumbuhan, sedangkan kaktus menimpa (override) method daun karena method daun pada class tumbuhan tidak memiliki makna yang sama dan harus diganti.


Class tumbuhan\_air :



```
1 //membuat class yang terhubung dengan class tumbuhan sebagai superclassnya
2 public class tumbuhan_air extends tumbuhan
3 {
4     //menggunakan metode override menulis ulang method induk
5     public void tempat_hidup(){
6         System.out.println("hidup di air");
7     }
8     //menggunakan metode override menulis ulang method induk
9     public void jenis_buah(){
10        System.out.println("tidak memiliki buah");
11    }
12 }
```

Class tumbuhan\_air menimpa class tumbuhan pada method tempat hidup dan jenis\_buah.

Class enceng\_gondok dan teratai :



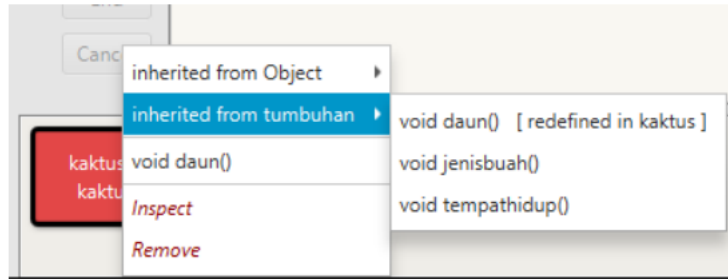
```
enceng_gondok X
1 //membuat class yang terhubung dengan class tumbuhan_air sebagai superclassnya
2 public class enceng_gondok extends tumbuhan_air
3 {
4     //tidak perlu diisi karena prinsip inheritance mewariskan method dari tumbuhan air ke class ini
5 }

teratai X
1 //membuat class yang terhubung dengan class tumbuhan_air sebagai superclassnya
2 public class teratai extends tumbuhan_air
3 {
4     //tidak perlu diisi karena prinsip inheritance mewariskan method dari tumbuhan air ke class ini
5 }
```

Class enceng\_gondok dan teratai terhubung dengan class tumbuhan air sehingga sudah tidak terpengaruh dengan class tumbuhan lagi, jadi method yang digunakan adalah method yang ditimpa di class tumbuhan\_air dan method warisan yang tidak ditulis ulang.

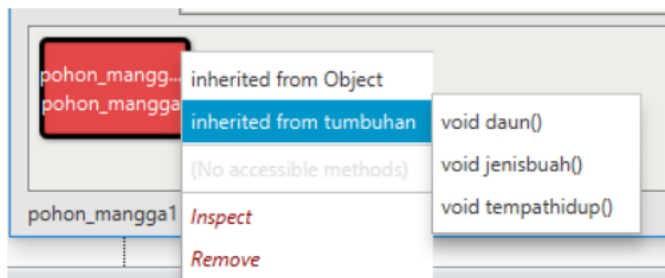
## Running program :

Class kaktus :



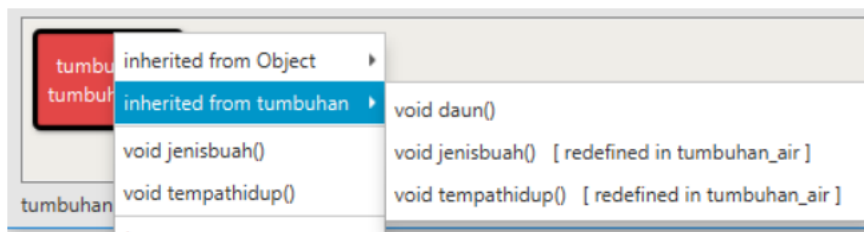
Perhatikan pada inherited from tumbuhan jadi diturunkan dari tumbuhan, pada method daun ada keterangan redefined in kaktus, jadi method daun telah ditimpa atau ditulis ulang pada class kaktus.

Class pohon\_mangga :



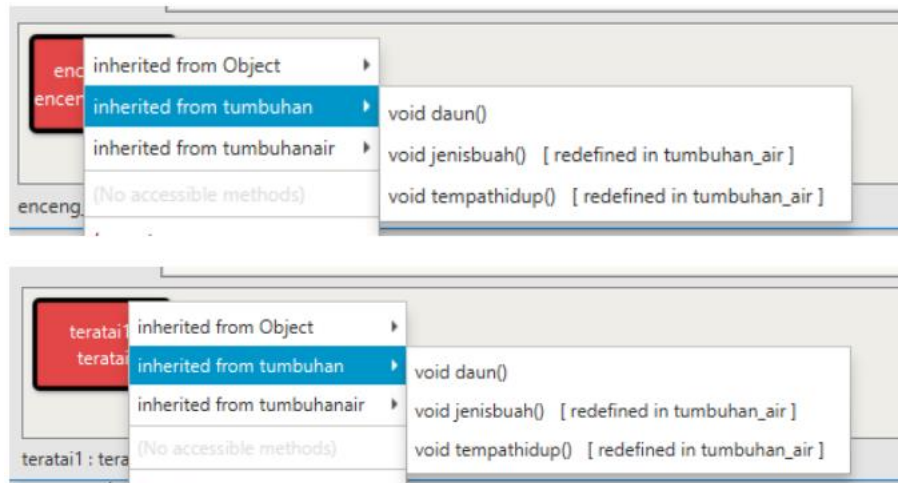
Pada inherited from tumbuhan pada semua methodnya tidak ada redefined berarti class ini mewarisi semua yang ada di class tumbuhan.

Class tumbuhan\_air :



Class tumbuhan\_air menimpa atau menulis ulang class tumbuhan pada bagian method jenisbuah dan tempathidup.

Class enceng\_gondok dan teratai :



Pada class `enceng_gondok` dan `teratai` dapat dilihat mereka menerima method dari class `tumbuhan air` karena pada method `jenisbuah` dan `tempathidup` telah ditulis ulang oleh class `tumbuhan air` sedangkan untuk `daun` masih warisan dari class `tumbuhan`.

## BAB III

### Tugas Pendahuluan

#### 3.1 soal

1. Saat ini industri mobile gaming merupakan industri yang sangat menjanjikan. Fakta ini tidak terbantahkan. Banyak pengembang game yang awalnya fokus di desktop kini merambah ke perangkat mobile. Faktor ini jugalah yang kemudian akhirnya membuat smartPhone terus berkembang dengan spesifikasi yang tinggi. sebagai seorang developer, jelaskan secara detail penggunaan konsep inheritance dan override dalam pembuatan (develop) sebuah game.

#### 3.2 Jawaban

- 1.- Inheritance (pewarisan): konsep di mana sebuah class dapat mewarisi properti dan metode dari class lainnya. Ini memungkinkan pengembang untuk mengelompokkan objek yang serupa dan menghindari duplikasi kode.
- Override (penggantian): konsep dimana class turunan (subclass) dapat mengganti implementasi metode yang diwarisi dari class dasar (superclass). Ini memungkinkan pengembang untuk menyesuaikan perilaku objek sesuai dengan kebutuhan khusus mereka.

Dengan menggunakan kedua konsep ini, pengembang game dapat mengorganisir kode mereka dengan lebih baik, mempercepat pengembangan, dan membuat game lebih mudah untuk dimodifikasi dan diperluas di masa mendatang.



## **BAB IV**

### **IMPLEMENTASI**

#### **4.1 Tugas praktikum**

Buatlah aplikasi perpustakaan sederhana sebagai berikut: Masukkan data buku sebanyak X kali

1. Judul : Judul buku.
2. Penulis : Nama penulis film tersebut.
3. Publisher : Perusahaan yang menerbitkan buku tersebut.
4. Kategori : SU = Semua Umur, D = Dewasa, R = Remaja, A = Anak-anak
5. Stok : Jumlah stok buku tersebut.
6. Tahun Terbit : Tahun buku tersebut diterbitkan

Judul, Penulis, Publisher, Kategori, Stok, dan Tahun Terbit di-looping sebanyak X kali.

Desainlah aplikasi perpustakaan tersebut dengan konsep inheritance dan tentukan parent class serta child class-nya. Setelah itu, implementasikan class-class yang telah didesain dengan membuat program sederhana yang memiliki fasilitas entri data buku dan melihat daftar buku yang telah di-entri-kan.

## 4.2 Source code

```
package perpustakaan;
import java.util.ArrayList;
import java.util.Scanner;

class Buku {
    String judul, penulis, penerbit, kategori;
    int stok;
    String tahunTerbit;

    public Buku(String judul, String penulis, String penerbit, String kategori, int stok, String tahunTerbit) {
        this.judul = judul;
        this.penulis = penulis;
        this.penerbit = penerbit;
        this.kategori = kategori;
        this.stok = stok;
        this.tahunTerbit = tahunTerbit;
    }

    public void displayInfo() {
        System.out.println("Judul: " + judul);
        System.out.println("Penulis: " + penulis);
        System.out.println("Penerbit: " + penerbit);
        System.out.println("Kategori: " + kategori);
        System.out.println("Stok: " + stok);
        System.out.println("Tahun Terbit: " + tahunTerbit);
    }
}
```

```
package perpustakaan;
/**
 *
 * @author RIZQY
 */
public class Anak extends Buku {
    public Anak (String penulis, String publisher, String kategori, int stok, String tahunT) {
        super(penulis,publisher,"A",kategori,stok,tahunT );
    }
}
```

```
package perpustakaan;
/**
 *
 * @author RIZQY
 */
public class SemuaU extends Buku {
    public SemuaU (String penulis, String publisher, String kategori, int stok, String tahunT) {
        super(penulis,publisher,"SUS",kategori,stok, tahunT );
    }
}
```

```

- */
package perpustakaan;
/**
 *
 * @author RIZQY
 */
public class Dewasa extends Buku {
    public Dewasa (String penulis, String publisher, String kategori, int stok, String tahunT) {
        super(penulis,publisher,"D",kategori,stok,tahunT );
    }
}

```

```

- */
package perpustakaan;
/**
 *
 * @author RIZQY
 */
public class Remaja extends Buku {
    public Remaja (String penulis, String publisher, String kategori, int stok, String tahunT) {
        super(penulis,publisher,"R",kategori,stok,tahunT );
    }
}

```

```

package perpustakaan;
/**
 * @author RIZQY
 */
public class Tampilan extends Buku {
    public Tampilan(String judul, String penulis, String publisher, String kategori,int stok, String tahunT) {
        super(judul, penulis, publisher, kategori, stok, tahunT);
    }

    @Override
    public void displayInfo() {
        // Implementasi detail informasi yang ingin ditampilkan
        System.out.println("Judul Buku: " + judul);
        System.out.println("Penulis: " + penulis);
        System.out.println("Publisher: " + publisher);
        System.out.println("Kategori: " + kategori);
        System.out.println("Stok: " + stok);
        System.out.println("Tahun Terbit: " + tahunT);
    }
}

```

```

System.out.print("Kategori (SU = Semua Umur, D = Dewasa, R = Remaja, A = Anak-anak) : ");
String kategoriInput = scanner.nextLine();
String kategori = null;
while (true) {
    if (kategoriInput.equalsIgnoreCase("SU")){
        kategori = "Semua umur";
        break;
    } else if (kategoriInput.equalsIgnoreCase("D")){
        kategori = "Dewasa";
        break;
    } else if (kategoriInput.equalsIgnoreCase("R")){
        kategori = "Remaja";
        break;
    } else if (kategoriInput.equalsIgnoreCase("A")){
        kategori = "Anak-anak";
        break;
    } else {
        System.out.println("Kategori tidak valid, masukkan lagi: ");
        kategoriInput = scanner.nextLine();
    }
}

System.out.print("Stok buku yang tersedia : ");
int stok = scanner.nextInt();
scanner.nextLine();

System.out.print("Tahun terbit buku : ");
String tahunT = scanner.nextLine();

Buku buku = new Buku (judul, penulis, penerbit, kategori, stok, tahunT);
daftarBuku.add(buku);
}

```

```

// Menampilkan daftar buku yang dipilih
System.out.println("\nDaftar buku yang telah dipilih:");
for (int i = 0; i < daftarBuku.size(); i++) {
    System.out.println("\nBuku ke-" + (i + 1));
    daftarBuku.get(i).displayInfo();
}

scanner.close();
}
}

```

### 4.3 Hasil output

```
Buku ke-1
Judul: spiderman
Penulis: rizqi
Penerbit: pt home
Kategori: Anak-anak
Stok: 3
Tahun Terbit: null

Buku ke-2
Judul: Ironman
Penulis: rizqi
Penerbit: pt jaya
Kategori: Remaja
Stok: 3
Tahun Terbit: null
BUILD SUCCESSFUL (total time: 1 minute 46 seconds)
```

- Penjelasan

Disini saya membuat program beberapa buku, yang terdiri dari judul, penulis, penerbit, kategori, stok, tahun terbit . Mengulangi entri data untuk setiap buku. Di dalam lingkaran : Input Detail Buku : Ini meminta pengguna untuk setiap judul buku, penulis, penerbit, kategori, stok, dan tahun penerbitan. Validasi Kategori : Memastikan kategori input valid (SU, D, R, A). Jika tidak, ia bertanya lagi. Membuat dan Menyimpan Objek Buku : Membuat Bukuobjek dengan detail yang disediakan dan menambahkannya ke daftarBukudaftar. Program ini dirancang untuk mengelola perpustakaan buku dengan memungkinkan pengguna memasukkan detail beberapa buku dan kemudian menampilkan detail tersebut. Ini menggunakan prinsip pemrograman berorientasi objek dengan merangkum detail buku di Bukukelas dan mengelola banyak buku menggunakan file ArrayList. Program ini menangani masukan pengguna dan memvalidasinya, khususnya untuk kategori buku, memastikan entri data yang kuat.

## **BAB V**

### **PENUTUP**

#### **5.1 Analisa**

**Inheritance** (Pewarisan) Definisi: Inheritance adalah sebuah mekanisme di dalam pemrograman berorientasi objek yang memungkinkan sebuah class untuk mewarisi atribut dan metode dari class lain. Class yang mewarisi disebut sebagai subclass (atau derived class, child class), sedangkan class yang diwarisi disebut superclass (atau base class, parent class). Keuntungan : Pengurangan Redundansi: Pewarisan memungkinkan penggunaan kembali kode yang sudah ada tanpa harus menulis ulang. Ini mengurangi redundansi kode dan memperbaiki efisiensi pengembangan. Pemeliharaan yang Mudah: Perubahan pada superclass akan otomatis terdistribusi ke subclass, kecuali jika fungsi tersebut di-override. Ini memudahkan pemeliharaan dan pengelolaan kode. Kekurangan : Ketergantungan Berlebihan: Terlalu banyak pewarisan dapat membuat struktur class menjadi rumit dan sulit untuk ditelusuri.

**Override** (Penimpaan) Definisi : Override adalah mekanisme di mana subclass memodifikasi implementasi dari metode yang diwarisi dari superclass. Subclass menggunakan nama yang sama untuk metode tersebut, tetapi mengubah perilaku yang didefinisikan dalam superclass. Keuntungan : Kustomisasi Perilaku: Override memungkinkan subclass untuk memodifikasi atau memperluas perilaku yang diwarisi dari superclass sesuai dengan kebutuhan yang lebih spesifik. Polimorfisme: Override adalah kunci dari polimorfisme, di mana metode yang sama dapat memiliki perilaku yang berbeda berdasarkan objek mana yang memanggil metode tersebut. Kekurangan : Kompleksitas Kode: Jika banyak metode di-override, hal ini bisa membuat kode sulit untuk dipahami dan dipelihara. Resiko Tersembunyi: Jika tidak dirancang dengan hati-hati, override dapat menyebabkan perilaku yang tidak diinginkan karena mengubah fungsi yang mungkin tidak sepenuhnya dipahami oleh pengembang subclass.

## 5.2 Kesimpulan

### **Inheritance (Pewarisan)**

- Pengurangan Redundansi: Mengizinkan penggunaan kembali kode, mengurangi kebutuhan untuk menulis ulang kode yang sama.
- Pemeliharaan yang Mudah: Perubahan pada superclass secara otomatis terdistribusi ke subclass, memudahkan pemeliharaan.
- Polimorfisme: Memungkinkan objek dari subclass diperlakukan sebagai objek dari superclass, memberikan fleksibilitas dalam pemanggilan metode.
- Ketergantungan Berlebihan: Bisa membuat struktur class rumit dan sulit untuk ditelusuri jika digunakan secara berlebihan.
- Kopling yang Kuat: Subclass yang terlalu bergantung pada perilaku superclass dapat menjadi masalah jika terjadi perubahan besar pada superclass.

### **Override (Penimpaan)**

- Kustomisasi Perilaku: Memungkinkan subclass untuk mengubah atau memperluas perilaku yang diwarisi dari superclass.
- Polimorfisme: Kunci dari polimorfisme, dimana metode yang sama bisa memiliki perilaku yang berbeda tergantung objek yang memanggilnya.
- Kompleksitas Kode: Membuat kode menjadi lebih sulit untuk dipahami jika terlalu banyak metode yang di-override.
- Resiko Tersembunyi: Bisa menyebabkan perilaku yang tidak diinginkan atau bug jika override tidak dilakukan dengan hati-hati.