LAPORAN PRAKTIKUM
PERTEMUAN 3 MODUL II
Pewarisan Tunggal (Inheritance)

Alif Alpian Sahrul Muharom (20102007)

Dosen :

Agus Priyanto S.kom, M.kom

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2022

I. TUJUAN

a. Mengerti dan memahami tentang konsep pewarisan tunggal (Inheritance).
b. Mampu mewujudkan bentuk pewarisan tunggal dalam C++ dan Java.

II. TOOL

1. Apache NetBeans IDE 13
2. Java SE Development Kit 18

III. DASAR TEORI

a) Inheritance pada Java

Di Java, istilah inheritance mengacu pada adopsi semua properti non-privat dan metode dari satu kelas (superclass) oleh kelas lain (subclass). Inheritance adalah cara membuat salinan kelas yang sudah ada sebagai titik awal untuk kelas yang lain. Selain istilah 'subclass', kelas inheritance juga disebut kelas turunan.

Untuk lebih jelasnya, jika subclass dibuat menggunakan superclass dan subclass tetap tidak berubah, kedua kelas tersebut akan identik. Tetapi kebanyakan subclass tidak tetap tidak berubah. Karena subclass masih merupakan kelas, itu dapat diubah untuk menyertakan properti dan metode baru. Subclass yang telah selesai bahkan dapat digunakan sebagai superclass untuk membuat subclass tambahan. Tidak ada batasan efektif untuk jumlah level warisan.

Metode dan properti subkelas dapat digunakan seperti superkelasnya. Mereka juga bisa diganti. Overriding adalah proses mengganti (atau menambah) kode asli dengan kode baru agar sesuai dengan tujuan saat ini. *Method signature* yang diganti di subclass tetap sama dengan superclass tetapi konten metode akan diubah untuk memenuhi tujuan metode dalam bentuk barunya.

b) Overriding

Overriding atau sering pula disebut dengan *redefinisi* adalah kemampuan suatu kelas turunan untuk memodifikasi (mendefinisikan kembali) data dan method dari kelas induknya. Proses ini akan mengubah data method dari keduanya, kelas induk dan kelas turunannya. Alasan mengapa dilakukan overriding antara lain jika akan dilakukan perubahan secara menyeluruh, baik jumlah maupun tipe parameter maupun behaviour pemrosesan datanya. Overriding dapat juga dilakukan jika akan dilakukan perubahan hanya untuk menambahkan behaviour khusus yang dimiliki hanya oleh kelas turunan tersebut. Yang perlu diperhatikan dalam melakukan overriding adalah modifier penentu aksesbilitas data dan methodnya yakni *private, public atau protected*.

1. Public

Mengijinkan kelas dan sub kelas dari package manapun untuk mengaksesnya.

2. Private

Membatasi akses hanya untuk kelas itu sendiri dan objek yang diinstans darinya.

3. Protected

Akses hanya diberikan kepada kelas itu sendiri dan sub kelas yang diturunkan darinya.

IV. GUIDED

1. Guided 1 (projectSpeak)

Dog.java

Source code

```
package com.Alpiann.pertemuan5.guided.projectSpeak;

/**
 *
 * @author Lenovo
 */
public class Dog extends
  Mammal { public void
  speak() {
    System.out.println("Arf! Arf!");
  }
}
```

Duck.java

Source code

```
package com.Alpiann.pertemuan5.guided.projectSpeak;

/**
 *
 * @author Lenovo
 */
public class Duck
  extends Mammal{ public
  void speak() {
    System.out.println("Quack! Quack!");
  }
}
```

Horse.java

Source code

```
package com.Alpiann.pertemuan5.guided.projectSpeak;

/**
 *
 * @author Lenovo
 */
public class Horse
  extends Mammal{ public
  void speak() {
    System.out.println("Whinny! Whinny!");
  }
}
```

MikeWallace.java
Source code

```
package com.Alpiann.pertemuan5.guided.projectSpeak;

/**
 *
 * @author Lenovo
 */
public class MikeWallace extends
  Mammal { public void speak() {
    System.out.println("Can I ask you a few question about "
        + "your 1087 tax statement");
  }
}
```

MorleySafer.java
Source code

```
package com.Alpiann.pertemuan5.guided.projectSpeak;

/**
 *
 * @author Lenovo
 */
public class MorleySafer
  extends Mammal{ public void
  speak() {
    System.out.println("Can I ask you a few question about "
        + "your 1087 tax statement");
  }
}
```

Owl.java
Source code

```
package com.Alpiann.pertemuan5.guided.projectSpeak;

/**
 *
 * @author Lenovo
 */
public class Owl
  extends Mammal{
  public void speak() {
    System.out.println("Whoo! Whoo!");
  }

  //jika menggunakan @override, maka saat
  //main dijalankan, method ini yang akan berjalan(bukan yang
  ada
  //pada class Mammal
```

```
  @Override
  public void
    sleep() {
    System.out.prin
    tln(" ");
  }
}
```

Penjelasan:
Kelas-kelas di atas merupakan turunan dari induk kelas Mammal. Setiap kelas memiliki method speak( ) yang akan dijalankan di badan main. Namun, khusus kelas Owl.java terdapat @Override, dimana command tersebut memberikan perintah yang dapat menghiraukan method sleep( ) dari kelas induk.

Mammal.java
Source code

```
package com.Alpiann.pertemuan5.guided.projectSpeak;

/**
 *
 * @author Lenovo
 */
public class
  Mammal {
  protected
  String name;

  public void sleep() {
    System.out.println("ZZZZ
    ZZZZZ ZZZZ");
  }
}
```

Penjelasan:
Mammal.java merupakan induk kelas dari kelas-kelas dog, owl, duck, dll. Protected merupakan atribut yang hanya dapat digunakan antar kelas induk-turunan. Method sleep () akan dijalankan pada badan main.

Main.java
Source code

```
package com.Alpiann.pertemuan5.guided.projectSpeak;

/**
 *
 * @author Lenovo
 */
public class Main {
  public static void
    main(String[] args) {
```

```
        Dog guffy = new Dog();
        Duck donald = new Duck();
        MorleySafer morley = new MorleySafer();

        Owl woodsy = new Owl();

        guffy.name = "Guffy";
        donald.name =
        "Donald"; morley.name
        = "Morley Safer";
        woodsy.name =
        "Woodsy";

        System.out.println("First we'll get
        the dog to speak:"); guffy.speak();
        System.out.println();

        System.out.println("Now, the duck
        will speak:"); donald.speak();
        System.out.println();

        System.out.println("Now it's Morley's
        turn to speak:"); morley.speak();
        System.out.println();

        System.out.println("Finally, the owl
        will speak:"); woodsy.speak();
        System.out.println();

        System.out.println("Time  for  all
        four to sleep:"); guffy.sleep();
        donald.sl
        eep();
        morley.sl
        eep();
        woodsy.sl
        eep();
      }
    }
```

Penjelasan:
Karena kelas main, maka Main.java harus memiliki object dimana object akan digunakan untuk memanggil method-method dari semua kelas yang ada pada suatu package.

2. Guided 2 (projectCircle)

Circle.java

Source code

```
package com.Alpiann.pertemuan5.guided.projectPoint;

/**
 *
 * @author Lenovo
 */
public class Circle
  extends Point{ private
  float radius;
  public Circle (float r,
    float a, float b){
    super(a, b);
    radius = r;
    System.out.println("Konstruktor Circle dijalankan");
  }

  public void cetakPoint(){
    super.cetakPoint();
    System.out.println("Radius :"
    + radius);
  }
}
```

Penjelasan:

Circle.java merupakan turunan dari kelas induk Point.java. Perbedaan dari atribut antara Circle.java dan Point.java yaitu pada turunan kelas terdapat atribut float r untuk menyimpan nilai dari object pada Main.java (6.5f dan 5). Super pada method cetakPoint berfungsi untuk memanggil method cetakPoint dari kelas induk.

Point.java

Source code

```
package com.Alpiann.pertemuan5.guided.projectPoint;

/**
 *
 * @author Lenovo
 */
public class
  Point {
  protected
  float x,y;

  public Point (float a, float b){
    System.out.println("Konstruktor Point
    dijalankan"); x = a;
    y = b;
  }
```

```
  public void cetakPoint(){
    System.out.println("Point :
    ["+x+", "+y+"]");
  }
}
```

Penjelasan:
Di dalam *constructor*, atribut float x dan y di deklarasikan ulang menjadi float a dan float b. Jadi, saat digunakan lagi pada Point.java, maka penggunaannya harus float a dan b. Method cetakPoint ( ) digunakan untuk menyimpan nilai pada object di kelas Main.

Main.java
Source code

```
package com.Alpiann.pertemuan5.guided.projectPoint;

/**
 *
 * @author Lenovo
 */
public class Main {
  public static void main(String[] args) {
    Circle circle1 = new
    Circle(6.5f, 8.2f, 1.9f);
    circle1.cetakPoint();

    System.out.println("");

    Circle circle2 = new
    Circle(5,5,6);
    circle2.cetakPoint();
  }
}
```

Penjelasan:
Pada kelas Main.java program dijalankan. Nilai object yang pertama pada float a dan b adalah 8.2f dan 1.9f dengan radius 6.5f. Pada object yang kedua, nilai float a dan b adalah 5 dan 6 dengan radius 5.

V. UNGUIDED

Satpam.java
Source code

```
package com.Alpiann.pertemuan5.unguided.Asuransi;

/**
 *
 * @author Lenovo
 */
public class Satpam extends
  Pegawai { private int
  gajiPokok;
```

```
  private int jamLembur;
```

```
gajiPokok;
  private int jamLembur;

  public int
    getGajiPokok() {
    return gajiPokok;
  }

  public int
    getJamLembur() {
    return jamLembur;
  }

      public void setSatpam(String nama,
          String NIP, String alamat, int
          tahunMasuk, int gajiPokok, int
          jamLembur) {
        this.nama
        = nama;
        this.NIP =
        NIP;
        this.alama
        t =
        alamat;
        this.tahunMasuk =
        tahunMasuk;
        this.gajiPokok =
        gajiPokok;
        this.jamLembur =
        jamLembur;
      }

      public int HitungGajiAkhir() {
        int gajiAkhir = gajiPokok + (10000 *
        jamLembur); return gajiAkhir;
      }

      public void cetakSatpam(){
         System.out.println("\n—
        Data
         Satpam--");
         System.out.println("Nama:
      "      + this.nama);
         System.out.println("NIP: "
      +    this.NIP);
         System.out.println("Alamat
      :    " + this.alamat);
```

```
        System.out.println("Tahun Masuk: " +
        this.tahunMasuk);
        System.out.println("Gaji Pokok: " +
        this.gajiPokok);
        System.out.println("Jumlah Jam Lembur: "
        + this.jamLembur);
        System.out.println("Gaji Akhir: " +
        HitungGajiAkhir());
      }
          }
```

Sales.java

Source code

```
package com.Alpiann.pertemuan5.unguided.Asuransi;

/**
 *
 * @author Lenovo
 */
public class Sales extends
  Pegawai{ private int
  gajiPokok;
  private int jumlahPelanggan;

  public int
    getGajiPokok() {
    return gajiPokok;
  }

  public int
    getJumlahPelanggan() {
    return jumlahPelanggan;
  }
  public void setSales(String nama, String
      NIP, String alamat, int tahunMasuk, int
      gajiPokok, int jumlahPelanggan) {
    this.nama =
    nama; this.NIP =
    NIP; this.alamat
    = alamat;
    this.tahunMasuk = tahunMasuk;
    this.gajiPokok = gajiPokok;
    this.jumlahPelanggan =
    jumlahPelanggan;
  }

  public int HitungGajiAkhir() {
    int gajiAkhir = gajiPokok + (50000 *
    jumlahPelanggan); return gajiAkhir;
  }
```

```
  public void cetakSales() {
    System.out.println("\n--Data
    Sales--");
    System.out.println("Nama: " +
    this.nama);
    System.out.println("NIP: " +
    this.NIP);
    System.out.println("Alamat: " +
    this.alamat);
    System.out.println("Tahun Masuk: " +
    this.tahunMasuk); System.out.println("Gaji
    Pokok: " + this.gajiPokok);
    System.out.println("Jumlah Pelanggan yang
    Direkrut: "
        + this.jumlahPelanggan);
    System.out.println("Gaji Akhir: " + HitungGajiAkhir());
  }
  }
```

Manajer.java

Source code

```
package com.Alpiann.pertemuan5.unguided.Asuransi;

/**
 *
 * @author Lenovo
 */
public class Manajer extends
  Pegawai{ private int
  gajiPokok;
  private String divisi;
  private int tunjanganJabatan;

  public int
    getGajiPokok() {
    return gajiPokok;
  }
  public String
    getDivisi() {
    return divisi;
  }

  public int
    getTunjanganJabatan() {
    return tunjanganJabatan;
  }

  public void setManajer(String nama, String NIP, String
  alamat,
    int tahunMasuk, int gajiPokok, int
    tunjanganJabatan, String divisi) { this.nama =
```

```
        nama;
        this.NIP = NIP;
        this.alamat =
        alamat;
        this.tahunMasuk = tahunMasuk;
        this.gajiPokok = gajiPokok;
        this.tunjanganJabatan =
        tunjanganJabatan; this.divisi =
        divisi;
      }
      public int HitungGajiAkhir() {
        int gajiAkhir = gajiPokok +
        tunjanganJabatan; return gajiAkhir;
      }
      System.out.println("Alamat: " +
        this.alamat);
        System.out.println("Tahun Masuk: " +
        this.tahunMasuk); System.out.println("Gaji Pokok:
        " + this.gajiPokok); System.out.println("Tunjangan
        Jabatan: " + this.tunjanganJabatan);
        System.out.println("Divisi: " + this.divisi);
        System.out.println("Gaji Akhir: " + HitungGajiAkhir());
      }
        }
```

Penjelasan:

Kelas Satpam.java, Sales.java, dan Manajer.java merupakan turunan dari kelas Pegawai.java yang dimana mereka mewarisi atribut dari kelas induk. Perbedaan dari ketiganya adalah pada kelas Satpam.java terdapat perhitungan gaji akhir di mana method tersebut menggunakan atribut int jamLembur. Pada kelas Sales, perhitungannya menggunakan atribut int jumlahPelanggan. Sedangkan pada Manajer.kelas menggunakan int tunjanganJabatan. Hanya terdapat satu method setter pada kelas turunan di mana method tersebut akan dipanggil di dalam kelas Main.java.

Pegawai.java

Source code

```
    package com.Alpiann.pertemuan5.unguided.Asuransi;

    /**
     *
     * @author Lenovo
     */
    public class Pegawai
      { protected String
      NIP; protected
      String nama;
      protected String
      alamat; protected
```

```
    int tahunMasuk;
      protected int
      gajiAkhir;
    }
```

Penjelasan:
Kelas induk berfungsi untuk menyimpan atribut-atribut yang sama yang diperlukan di dalam kelas turunan.

Main.java
Source code

```
package com.Alpiann.pertemuan5.unguided.Asuransi;

/**
 *
 * @author Lenovo
 */
public class Main {
  public static void
    main(String[] args) {
    Satpam S = new Satpam();
    Sales T = new Sales();
    Manajer M = new
    Manajer();
    S.setSatpam("Rendra","0042","Jl. Itik 15",2000,300000,5);
    T.setSales("Wibisana","0185","Jl. Ayam 78",2006,500000,10);
    M.setManajer("Adi","0005","Jl. Angsa 56" ,1999 ,1500000,
        450000, "Keuangan");
    System.out.println("\n\n==DISPLAY DATA
    KARYAWAN=="); S.cetakSatpam();
    T.cetakSales()
    ;
    M.cetakManajer
    ();
  }
}
```

Penjelasan:
Terdapat 3 object pada kelas Main.java, yaitu new Satpam(), new Sales(), dan new Manajer(). Object tersebut lalu digunakan untuk memanggil method setter dari ketiga kelas induk yang diisi dengan parameter sesuai dengan yang dibutukan kelas masing-masing. Setelah itu, output dari setiap turunan kelas dijalankan.

V. KESIMPULAN

Inheritance mengacu pada adopsi semua properti *non-private* dan metode dari satu kelas (superclass) oleh kelas lain (subclass). Inheritance adalah cara membuat salinan kelas yang sudah ada sebagai titik awal untuk kelas yang lain. Metode dan properti subkelas dapat digunakan seperti superkelasnya. Mereka juga bisa diganti.

Overriding adalah proses mengganti (atau menambah) kode asli dengan kode baru agar sesuai dengan tujuan saat ini. *Method signature* yang diganti di subclass tetap sama dengan superclass tetapi konten metode akan diubah untuk memenuhi tujuan metode dalam bentuk barunya. Overriding dapat juga dilakukan jika akan dilakukan perubahan hanya untuk menambahkan behaviour khusus yang dimiliki hanya oleh kelas turunan tersebut. Yang perlu diperhatikan dalam melakukan overriding adalah modifier penentu aksesbilitas data dan methodnya yakni *private, public atau protected*.