LAPORAN PRAKTIKUM
MATA KULIAH PEMROGRAMAN BERBASIS OBJEK

Dosen Pengampu : Vit Zuraida, S.Kom, M.Kom

JOBSHEET - 4

Nama : **M. Zidna Billah Faza**

NIM : **2341760030**

Prodi : **D-IV Sistem Informasi Bisnis**

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

INHERITANCE, OVERLOADING, OVERRIDING

A. Percobaan 1 (extends)

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

```
public class Pegawai {

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

}
```

2) Buatlah subclass dari class Pegawai dengan nama Dosen, kemudian buat juga constructor tanpa parameter dengan baris kode berikut:

```
public class Dosen extends Pegawai {

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

}
```

3) Buatlah main class, misal InheritanceDemo.java, lakukan instansiasi objek baru bernama dosen1 dari class Dosen sebagai berikut:

```
public class DemoDosen {
    Run | Debug
    public static void main(String[] args) {
        Dosen dosen1 = new Dosen();
    }
}
```

4) Run programnya kemudian amati hasilnya.

```
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat
```

B. Pertanyaan Percobaan 1 (extends)

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

   Parent class adalah class Pegawai sedangkan child class adalah class Dosen

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

   Dengan kata kunci extends

3) Berdasarkan hasil yang ditampilkan oleh program, ada berapa constructor yang dieksekusi saat instansiasi objek dosen1? Constructor class mana yang lebih dulu dieksekusi?

Pada saat eksekusi program terdapat 2 constructor yang dieksekusi. Pertama adalah constructor pada parent class yaitu Pegawai lalu constructor pada child class yaitu Dosen.

C. Percobaan 2 (Pewarisan)

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(x:"Objek dari class Pegawai dibuat");
    }

    public String getInfo() {
        String info = "";

        info += "NIP    : " + nip + "\n";
        info += "Nama   : " + nama + "\n";
        info += "Gaji   : " + gaji + "\n";

        return info;
    }

}
```

2) Tambahkan pula atribut NIDN pada class Dosen

```
public class Dosen extends Pegawai {

    public String nidn;

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

}
```

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

```
public class DemoDosen {
    Run | Debug
    public static void main(String[] args) {
        Dosen dosen1 = new Dosen();

        System.out.println(x:"");

        dosen1.nama = "Yayan Ruhiyan";
        dosen1.nip = "34328970";
        dosen1.gaji = 3000000;
        dosen1.nidn = "19832432";

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

4) Run program kemudian amati hasilnya

```
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat

NIP     : 34328970
Nama    : Yayan Ruhiyan
Gaji    : 3000000.0
```

D. Pertanyaan Percobaan 2 (Pewarisan)

1) Pada percobaan 2 diatas, apakah program dapat berhasil dijalankan ataukah terjadi error?

   Program dapat berjalan dengan sesuai.

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?

   Karena secara default child class memiliki warisan atribut dari parent class

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

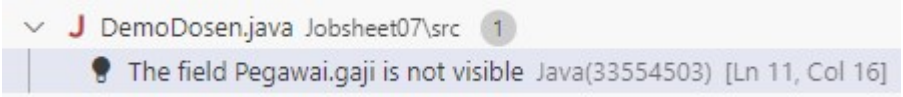
   Karena selain atribut yang diwariskan, method juga diwariskan ke child class dari parent class.

E. Percobaan 3 (Hak Akses)

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

```
public String nip;
public String nama;
private double gaji;
```

2) Run program kemudian amati hasilnya.



3) Ubah access level modifier atribut gaji menjadi protected kemudian pindah class Pegawai ke package baru, misalnya “testpackage”.

```
public String nip;
public String nama;
protected double gaji;
```

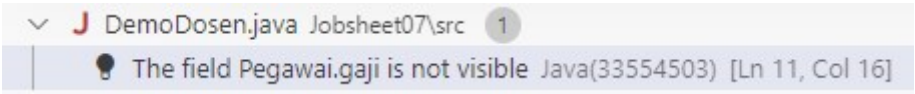
4) Import class Pegawai dari testpackage pada class Dosen.

```
import Jobsheet07.testpackage.Pegawai;
```

5) Akses atribut gaji pada class Dosen dengan coba mencetak atribut gaji pada constructor Dosen

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

6) Run program kemudian amati hasilnya



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

F. Pertanyaan Percobaan 3 (Hak Akses)

1) Pada langkah 1-2 di atas, terjadi error karena object dosen1 tidak dapat mengakses atributgaji. Padahal gaji merupakan atribut Pegawai yang merupakan parent class dari Dosen. Mengapa hal ini dapat terjadi?
Karena access modifier private hanya dapat diakses dari class yang sama.
2) Pada langkah 5-6, setelah class Pegawai berpindah ke package yang berbeda, class Dosen masih dapat mengakses atribut gaji. Mengapa?
Karena terdapat import yang memungkinkan untuk mengakses package testpackage.
3) Berdasarkan percobaan tersebut, bagaimana cara menentukan atribut dan method yang akan diwariskan oleh parent class ke child class?

Ketika akan membuat class apabila child class memiliki atribut yang sama dengan parent class maka access modifier public atau protected sedangkan ketika yang memiliki atribut tersebut hanya parent class maka private.

G. Percobaan 4 (Super - Atribut)

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;
}
```

2) Lakukan pemanggil method getAllInfo() oleh object dosen1 pada main function.

```
public class DemoDosen {
    Run | Debug
    public static void main(String[] args) {
        Dosen dosen1 = new Dosen();

        System.out.println(x:"");

        dosen1.nama = "Yayan Ruhiyan";
        dosen1.nip = "34328970";
        dosen1.gaji = 3000000;
        dosen1.nidn = "19832432";

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

3) Run program kemudian amati hasilnya

```
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat

NIP    : 34328970
Nama   : Yayan Ruhiyan
Gaji   : 3000000.0
NIDN   : 19832432
```

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;
}
```

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

```
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat

NIP    : 34328970
Nama   : Yayan Ruhiyan
Gaji   : 3000000.0
NIDN   : 19832432
```

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;
}
```

7) Run program kemudian bandingkan hasilnya dengan progam pada no 1 dan no 4.

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        nidn cannot be resolved or is not a field

        at Jobsheet07.src.Dosen.getAllInfo(Dosen.java:19)
        at Jobsheet07.src.DemoDosen.main(DemoDosen.java:14)
```

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;
}
```

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

```
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat

NIP     : 34328970
Nama    : Yayan Ruhiyan
Gaji    : 3000000.0
NIDN    : 19832432
```

H. Pertanyaan Percobaan 4 (Super - Atribut)

1) Apakah terdapat perbedaan output pada langkah 1, 4, dan 8? Mengapa?

Tidak terdapat perbedaan

- Langkah 1 : Mengakses atribut secara langsung, namun akan membingungkan mana atribut parent dan mana atribut child.
- Langkah 4 : Mengakses atribut dengan this, masih membingungkan karena tidak semua atribut berasal dari class child.
- Langkah 8 : Mengakses atribut dengan super dan this. Dengan membedakan atribut parent dengan super dan atribut child dengan this, maka tidak membingungkan.

2) Mengapa error terjadi pada program no 6?

Karena atribut nidn hanya ada pada class child. Yang mengebabkan error adalah menggunakan super padahal nidn ada pada class child.

I. Percobaan 5 (Super & Overriding)

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

```
public String getAllInfo() {
    String info = getInfo();

    info += "NIDN    : " + nidn + "\n";

    return info;
}
```

```
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat

NIP     : 34328970
Nama    : Yayan Ruhiyan
Gaji    : 3000000.0
NIDN    : 19832432
```

2) Lakukan modifikasi kembali pada method getAllInfo(). Run program kemudian amatihasilnya

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

    info += "NIDN    : " + nidn + "\n";

    return info;
}
```

```
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat

NIP     : 34328970
Nama    : Yayan Ruhiyan
Gaji    : 3000000.0
NIDN    : 19832432
```

3) Lakukan modifikasi kembali pada method getAllInfo(). Run program kemudian amatihasilnya

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

    info += "NIDN    : " + nidn + "\n";

    return info;
}
```

```
Objek dari class Pegawai dibuat
Objek dari class Dosen dibuat

NIP     : 34328970
Nama    : Yayan Ruhiyan
Gaji    : 3000000.0
NIDN    : 19832432
```

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(x:"Objek dari class Dosen dibuat");
    }

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

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

        info += "NIDN    : " + this.nidn + "\n";

        return info;
    }

}
```

J. Pertanyaan Percobaan 5 (Super & Overriding)

1) Apakah ada perbedaan method getInfo() yang diakses pada langkah 1, 2, dan 3?

Tidak ada perbedaan

- Langkah 1 : Mencari getInfo() di class dosen, apabila tidak ada maka akan mencari di parent class.
- Langkah 2 : Menggunakan this untuk mencari atribut pada class parent atau pada class child.
- Langkah 3 : Menggunakan super dari class parent, lalu menmbahkan atribut tambahan pada class child.

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

Terdapat perbedaan.

- super.getInfo() : Memanggil method getInfo() pada parent class.
- this.getInfo() : Melakukan override method getInfo(), maka outputnya adalah double.

```
NIDN   : 19832432
NIDN   : 19832432
```

3) Pada method manakah terjadi overriding? Jelaskan!

Terjadi overriding pada method getInfo() karena pada parent class ada dan child class ada.

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

Tidak, karena ketika sudah diberikan final maka method tersebut tidak bisa di override atau diubah.

K. Percobaan 6 (Overloading)

1) Tambahkan constructor baru untuk class Dosen sebagai berikut:

```
public Dosen(String nip, String nama, double gaji, String nidn) {
    System.out.println(x:"Obek dari class Dosen dibuat dengan constructor berparameter");
}
```

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

```
Dosen dosen2 = new Dosen(nip:"34328970", nama:"Yayan Ruhiyan", gaji:3000000, nidn:"19832432");
System.out.println(dosen2.getAllInfo());
```

L. Pertanyaan Percobaan 6 (Overloading)

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

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

public Dosen(String nip, String nama, double gaji, String nidn) {
    System.out.println(x:"Obek dari class Dosen dibuat dengan constructor berparameter");
}
```

Tidak teradi error karena constructor tersebut meiliki signature yang berbeda

2) Jelaskan apakah 2 constructor pada class Dosen memiliki signature yang sama?

Tidak, keduanya memiliki constructor yang berbeda, dosen() (tidak dengan parameter) dan dosen(String nip, String nama, double gaji, String nidn) (dengan parameter).

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

Dengan menggunakan konsep Overloading. Overloading terjadi ketika ada beberapa method atau constructor dalam satu class yang memiliki nama yang sama, tetapi parameter list-nya berbeda (baik dalam jumlah parameter, tipe data parameter, atau urutannya).

M. Percobaan 7 (Super - Constructor)

1) Modifikasi constructor berparameter 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;
}
```

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;
}
```

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;
}
```

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

```java
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;
    }

}
```

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

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

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

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

N. Pertanyaan Percobaan 7 (Super - Constructor)

1) Apakah terdapat perbedaan hasil pada langkah 1 dan 2? Jelaskan!

Tidak terdapat perbedaan, karena pada langkah 1 this berfokus pada objek yang sedang dibuat, sedangkan pada langkah 2 super menunjukkan bahwa atribut tersebut berasal dari class parent.

2) Apakah terdapat perbedaan hasil pada langkah 2 dan 3? Jelaskan!

Tidak terdapat perubahan, karena pada langkah 2 super menunjukkan bahwa atribut berasal dari class parent, sedangkan langkah 3 memanggil super() untuk memastikan constructor default dari class parent dipanggil terlebih dahulu.

3) Mengapa terjadi error pada langkah 4?

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        Implicit super constructor Pegawai() is undefined. Must explicitly invoke another constructor

        at Jobsheet07.src.Dosen.<init>(Dosen.java:7)
        at Jobsheet07.src.DemoDosen.main(DemoDosen.java:5)
```

Terjadi error karena ketika memanggil constructor Dosen(), Java mencoba memanggil constructor default dari class parent (Pegawai) yang mana constructor default sudah dihapus.

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

- Langkah 3 : Menggunakan super() untuk memanggil constructor default dari parent, dan kemudian menginisialisasi atribut parent (nip, nama, gaji) secara manual.
- Langkah 6 : Menggunakan super(nip, nama, gaji) untuk langsung memanggil constructor parent yang memiliki parameter, sehingga lebih ringkas dan efisien.

5) Mengapa terjadi error pada langkah 5?

Karena statement atau pemanggilan super() harus berada diawal baris.

O. 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 2 objek child class pada main class kemudian print info nya.

P. Jawaban Tugas

1) Program pada class Elektronik

```
package Jobsheet07.tugas;

public class Elektronik {

    public String model;
    public String nama;
    public String tahunRilis;

    public Elektronik() {

    }

    public Elektronik(String model, String nama, String tahunRilis) {
        this.model = model;
        this.nama = nama;
        this.tahunRilis = tahunRilis;
    }
    public String getModel() {
        return model;
    }
    public void setModel(String model) {
        this.model = model;
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public String getTahunRilis() {
        return tahunRilis;
    }
    public void setTahunRilis(String tahunRilis) {
        this.tahunRilis = tahunRilis;
    }

    public String getInfo() {
        String info = "";

        info += "Model          : " + model + "\n";
        info += "nama           : " + nama + "\n";
        info += "Tahun Rilis    : " + tahunRilis + "\n";

        return info;
    }

    public String getFeature() {
        return "Produk ini tidak memiliki fitur khusus";
    }

}
```

2) Program pada class Laptop

```
package Jobsheet07.tugas;

public class Laptop extends Elektronik {

    private String prosessor;
    private String ram;
    private String storage;

    public Laptop() {

    }

    public Laptop(String model, String nama, String tahunRilis, String prosessor, String ram, String storage) {
        super(model, nama, tahunRilis);
        this.prosessor = prosessor;
        this.ram = ram;
        this.storage = storage;
    }
    public String getProsessor() {
        return prosessor;
    }
    public void setProsessor(String prosessor) {
        this.prosessor = prosessor;
    }
    public String getRam() {
        return ram;
    }
    public void setRam(String ram) {
        this.ram = ram;
    }
    public String getStorage() {
        return storage;
    }
    public void setStorage(String storage) {
        this.storage = storage;
    }

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

        info += "Prosessor      : " + this.prosessor + "\n";
        info += "Ram            : " + this.ram + "\n";
        info += "Storage        : " + this.storage;

        return info;
    }

    public String getInfo() {
        String info = "";

        info += "Prosessor      : " + this.prosessor + "\n";
        info += "Ram            : " + this.ram + "\n";
        info += "Storage        : " + this.storage;

        return info;
    }

    public String getFeature() {
        return "Produk laptop " + this.nama + " memiliki fitur AI yang dapat memudahkan pekerjaan";
    }

}
```

3) Program pada class Kulkas

```
package Jobsheet07.tugas;

public class Kulkas extends Elektronik {

    private int jumlahPintu;
    private String watt;
    private String ukuran;

    public Kulkas() {

    }

    public Kulkas(String model, String nama, String tahunRilis, int jumlahPintu, String watt, String ukuran) {
        super(model, nama, tahunRilis);
        this.jumlahPintu = jumlahPintu;
        this.watt = watt;
        this.ukuran = ukuran;
    }

    public int getJumlahPintu() {
        return jumlahPintu;
    }

    public void setJumlahPintu(int jumlahPintu) {
        this.jumlahPintu = jumlahPintu;
    }

    public String getWatt() {
        return watt;
    }

    public void setWatt(String watt) {
        this.watt = watt;
    }

    public String getUkuran() {
        return ukuran;
    }

    public void setUkuran(String ukuran) {
        this.ukuran = ukuran;
    }

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

        info += "Jumlah Pintu   : " + this.jumlahPintu + "\n";
        info += "Watt           : " + this.watt + "\n";
        info += "Ukuran         : " + this.ukuran;

        return info;
    }

    public String getInfo() {
        String info = "";

        info += "Jumlah Pintu   : " + this.jumlahPintu + "\n";
        info += "Watt           : " + this.watt + "\n";
        info += "Ukuran         : " + this.ukuran;

        return info;
    }

    public String getFeature() {
        return "Produk kulkas " + this.nama + " memiliki fitur inverter yang hemat daya";
    }


}
```

4) Program pada class DemoElektronik

```
package Jobsheet07.tugas;

public class DemoElektronik {
    public static void main(String[] args) {

        header();
        System.out.println("           Elektronik Laptop      ");
        header();

        Laptop laptop1 = new Laptop("Chromebook", "Chromebook 4", "2024", "Intel Celeron N4020", "4 Gb", "32 Gb");
        System.out.println(laptop1.getAllInfo());
        header();

        Laptop laptop2 = new Laptop("Workstation", "Samsung Book 7", "2023", "AMD Ryzen 7 8845HS", "32 Gb", "1 Tb");
        System.out.println(laptop2.getAllInfo());
        System.out.println(laptop2.getFeature());
        header();

        System.out.println("           Elektronik Kulkas      ");
        header();

        Kulkas kulkas1 = new Kulkas("Multi-Door", "Samsung French-Door", "2022", 4, "130 Watt", "511 Liter");
        System.out.println(kulkas1.getAllInfo());
        header();

        Kulkas kulkas2 = new Kulkas("Top-Freezer", "Samsung RT22", "2021", 2, "100 Watt", "255 Liter");
        System.out.println(kulkas2.getAllInfo());
        System.out.println(kulkas2.getFeature());
        header();

    }

    public static void header() {
        int length = 40;
        for (int i = 0; i < length; i++) {
            System.out.print("=");
        }
        System.out.println("");
    }
}
```

5) Output

```
========================================
           Elektronik Laptop
========================================
Model           : Chromebook
nama            : Chromebook 4
Tahun Rilis     : 2024
Prosessor       : Intel Celeron N4020
Ram             : 4 Gb
Storage         : 32 Gb
========================================
Model           : Workstation
nama            : Samsung Book 7
Tahun Rilis     : 2023
Prosessor       : AMD Ryzen 7 8845HS
Ram             : 32 Gb
Storage         : 1 Tb
Produk laptop Samsung Book 7 memiliki fitur AI yang dapat memudahkan pekerjaan
========================================
           Elektronik Kulkas
========================================
Model           : Multi-Door
nama            : Samsung French-Door
Tahun Rilis     : 2022
Jumlah Pintu    : 4
Watt            : 130 Watt
Ukuran          : 511 Liter
========================================
Model           : Top-Freezer
nama            : Samsung RT22
Tahun Rilis     : 2021
Jumlah Pintu    : 2
Watt            : 100 Watt
Ukuran          : 255 Liter
Produk kulkas Samsung RT22 memiliki fitur inverter yang hemat daya
========================================
```