

Jobsheet 4: Relasi Kelas
Object Oriented Programming



Arranged by :
Shofwah Kanaka Ebsa Anargya
2241720254 / 22
2I

INFORMATION TECHNOLOGY
D-IV INFORMATICS ENGINEERING
MALANG STATE POLYTECHNIC
2023

I. Percobaan 1

```
package ac.id.polinema.relasticlass.percobaan1;
public class Processor {
    private String merk;
    private double cache;
    public Processor() {
        // Constructor default
    }
    public Processor(String merk, double cache) {
        this.merk = merk;
        this.cache = cache;
    }
    public String getMerk() {
        return merk;
    }
    public void setMerk(String merk) {
        this.merk = merk;
    }
    public double getCache() {
        return cache;
    }
    public void setCache(double cache) {
        this.cache = cache;
    }
    public void info() {
        System.out.printf("Merk Processor = %s\n", merk);
        System.out.printf("Cache Memory = %.2f\n", cache);
    }
}
```

```
package ac.id.polinema.relasticlass.percobaan1;
public class Laptop {
    private String merk;
    private Processor proc;
    public Laptop() {
        // Constructor default
    }
    public Laptop(String merk, Processor proc) {
        this.merk = merk;
        this.proc = proc;
    }
    public String getMerk() {
        return merk;
    }
    public void setMerk(String merk) {
        this.merk = merk;
    }
    public Processor getProc() {
        return proc;
    }
    public void setProc(Processor proc) {
        this.proc = proc;
    }
    public void info() {
        System.out.println("Merk Laptop = " + merk);
        proc.info();
    }
}
```

```

package ac.id.polinema.relasticlass.percobaan1;

public class MainPercobaan1 {
    public static void main(String[] args) {
        Processor p = new Processor("Intel i5", 3);
        Laptop L = new Laptop("Thinkpad", p);

        L.info();

        Processor p1 = new Processor();
        p1.setMerk("Intel i5");
        p1.setCache(4);

        Laptop L1 = new Laptop();
        L1.setMerk("Thinkpad");
        L1.setProc(p1);

        L1.info();
    }
}

```

Compile kemudian run class MainPercobaan1, akan didapatkan hasil seperti berikut:

```

Run MainPercobaan1 x
/Library/Java/JavaVirtualMachines/jdk-19.jdk
↑
↓
⇌
⇓
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 3.00
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 4.00
Process finished with exit code 0

```

Pertanyaan

Berdasarkan percobaan 1, jawablah pertanyaan-pertanyaan yang terkait:

1. Di dalam *class* Processor dan *class* Laptop, terdapat method *setter* dan *getter* untuk masing-masing atributnya. Apakah gunanya *method setter* dan *getter* tersebut?
 Getter digunakan untuk mengambil nilai atribut,
 Setter digunakan untuk mengatur atau memperbarui nilai atribut.
2. Di dalam *class* Processor dan *class* Laptop, masing-masing terdapat konstruktor default dan konstruktor berparameter. Bagaimanakah beda penggunaan dari kedua jenis konstruktor tersebut?
 Konstruktor default tidak menerima.
 Konstruktor berparameter menerima argumen
3. Perhatikan *class* Laptop, di antara 2 atribut yang dimiliki (*merk* dan *proc*), atribut manakah yang bertipe *object*?
 yang merujuk ke objek dari kelas Processor.

4. Perhatikan *class* Laptop, pada baris manakah yang menunjukkan bahwa *class* Laptop memiliki relasi dengan *class* Processor?

Relasi antara *class* Laptop dan *class* Processor pada baris `private Processor proc` terdapat pada *class* Laptop, *class* Laptop memiliki atribut yang bertipe objek Processor yang menghubungkannya dengan *class* Processor.

5. Perhatikan pada *class* Laptop, Apakah guna dari sintaks `proc.info()`?

Sintaks `proc.info()` digunakan untuk memanggil metode `info()` dari objek Processor yang dimiliki oleh objek Laptop.

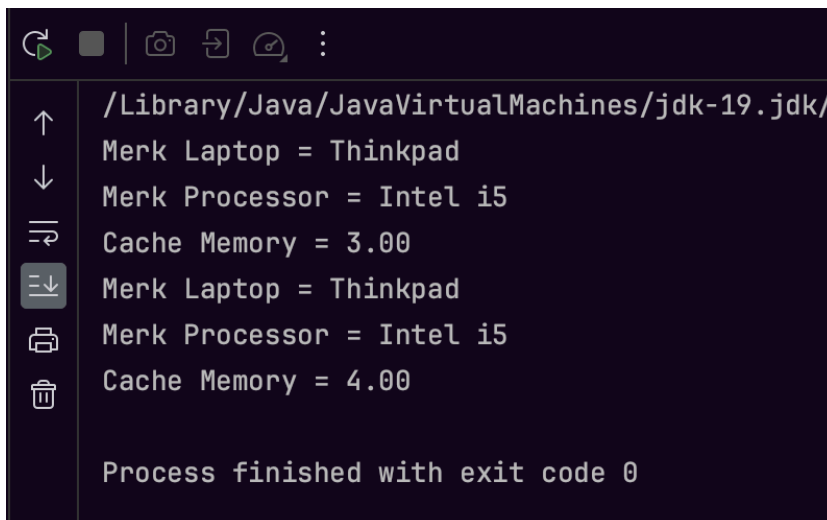
6. Pada *class* MainPercobaan1, terdapat baris kode:

`Laptop l = new Laptop("Thinkpad", p);`. Apakah `p` tersebut ?

Dan apakah yang terjadi jika baris kode tersebut diubah menjadi:

`Laptop l = new Laptop("Thinkpad", new Processor("Intel i5", 3));`

Bagaimanakah hasil program saat dijalankan, apakah ada perubahan ?



```
/Library/Java/JavaVirtualMachines/jdk-19.jdk/
↑
↓
⇌
⇓
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 3.00
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 4.00

Process finished with exit code 0
```

Pada baris `Laptop l = new Laptop("Thinkpad", p);`, `p` adalah objek dari *class* Processor. Jika diubah menjadi `Laptop l = new Laptop("Thinkpad", new Processor("Intel i5", 3));` maka objek Processor baru langsung sebagai argumen saat membuat objek Laptop. Hasil program saat dijalankan akan tetap sama.

Percobaan 2

```
package js4.percobaan2;

public class Mobil {
    private String merk;
    private int biaya;
    public Mobil() {
    }
    public String getMerk() {
        return merk;
    }
    public void setMerk(String merk) {
        this.merk = merk;
    }
    public int getBiaya() {
        return biaya;
    }
    public void setBiaya(int biaya) {
        this.biaya = biaya;
    }
    public int hitungBiayaMobil(int hari) {
        return biaya * hari;
    }
}
```

```
package js4.percobaan2;

public class Pelanggan {
    private String nama;
    private Mobil mobil;
    private Sopir sopir;
    private int hari;
    public Pelanggan() {
        // Constructor default
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public Mobil getMobil() {
        return mobil;
    }
    public void setMobil(Mobil mobil) {
        this.mobil = mobil;
    }
    public Sopir getSopir() {
        return sopir;
    }
    public void setSopir(Sopir sopir) {
        this.sopir = sopir;
    }
    public int getHari() {
        return hari;
    }
    public void setHari(int hari) {
        this.hari = hari;
    }
    public int hitungBiayaTotal() {
        return mobil.hitungBiayaMobil(hari) + sopir.hitungBiayaSopir(hari);
    }
}
```

```

package js4.percobaan2;

public class Sopir {
    private String nama;
    private int biaya;
    public Sopir() {
        // Constructor default
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public int getBiaya() {
        return biaya;
    }
    public void setBiaya(int biaya) {
        this.biaya = biaya;
    }
    public int hitungBiayaSopir(int hari) {
        return biaya * hari;
    }
}

```

```

package js4.percobaan2;

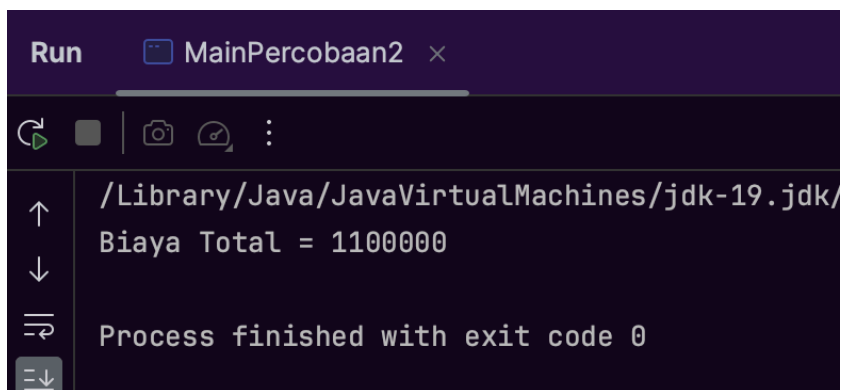
public class MainPercobaan2 {
    public static void main(String[] args) {
        Mobil m = new Mobil();
        m.setMerk("Avanza");
        m.setBiaya(350000);

        Sopir s = new Sopir();
        s.setNama("Adi");
        s.setBiaya(200000);

        Pelanggan p = new Pelanggan();
        p.setNama("Shofwah");
        p.setMobil(m);
        p.setSopir(s);
        p.setHari(2);

        System.out.println("Biaya Total = " + p.hitungBiayaTotal());
    }
}

```



```

Run MainPercobaan2
/Library/Java/JavaVirtualMachines/jdk-19.jdk/
Biaya Total = 1100000
Process finished with exit code 0

```

Pertanyaan

1. Perhatikan *class* Pelanggan. Pada baris program manakah yang menunjukkan bahwa *class* Pelanggan memiliki relasi dengan *class* Mobil dan *class* Sopir?

Relasi antara *class* Pelanggan, Mobil, dan Sopir terlihat pada baris berikut dalam *class* Pelanggan:

```
private Mobil mobil;
```

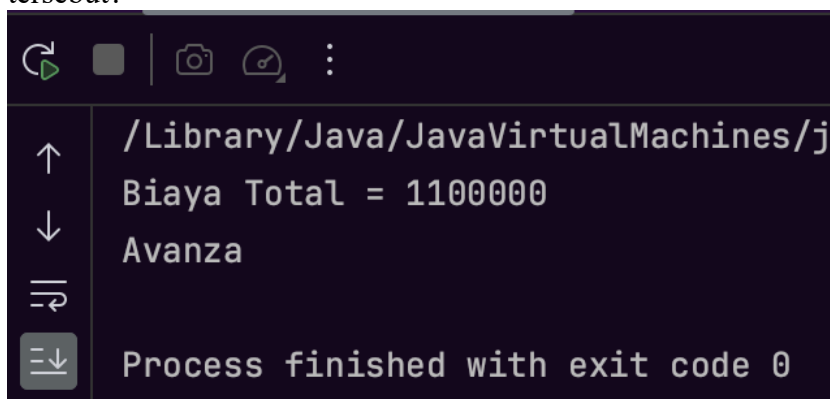
```
private Sopir sopir;
```

Ini menunjukkan bahwa class Pelanggan memiliki atribut yang bertipe Mobil dan Sopir, yang menunjukkan hubungan atau asosiasi dengan kelas tersebut.

- Perhatikan *method* hitungBiayaSopir pada class Sopir, serta method hitungBiayaMobil pada class Mobil. Mengapa menurut Anda *method* tersebut harus memiliki argument hari ?
Method hitungBiayaSopir pada class Sopir dan method hitungBiayaMobil pada class Mobil memerlukan argumen "hari" karena biaya perjalanan tergantung pada jumlah hari yang digunakan.
- Perhatikan kode dari *class* Pelanggan. Untuk apakah perintah mobil.hitungBiayaMobil(hari) dan sopir.hitungBiayaSopir(hari)?
Pada kode class Pelanggan, mobil.hitungBiayaMobil(hari) digunakan untuk menghitung biaya perjalanan menggunakan objek Mobil yang dimiliki oleh Pelanggan, dan sopir.hitungBiayaSopir(hari) digunakan untuk menghitung biaya sopir menggunakan objek Sopir yang dimiliki oleh Pelanggan. Kedua method tersebut memerlukan jumlah "hari" sebagai argumen untuk menghitung biaya berdasarkan lama perjalanan.
- Perhatikan *class* MainPercobaan2. Untuk apakah sintaks p.setMobil(m) dan p.setSopir(s)?
Sintaks p.setMobil(m) digunakan untuk mengatur objek Mobil yang dimiliki oleh Pelanggan p, dan p.setSopir(s) digunakan untuk mengatur objek Sopir yang dimiliki oleh Pelanggan p. Dengan kata lain, ini digunakan untuk menghubungkan Pelanggan dengan Mobil dan Sopir yang akan digunakan.
- Perhatikan class MainPercobaan2. Untuk apakah proses p.hitungBiayaTotal() tersebut ?
Proses p.hitungBiayaTotal() digunakan untuk menghitung biaya total perjalanan Pelanggan p. menggabungkan biaya perjalanan Mobil dan biaya sopir yang telah dihitung sebelumnya.
- Perhatikan class MainPercobaan2, coba tambahkan pada baris terakhir dari *method* main dan amati perubahan saat di-run!

```
System.out.println(p.getMobil().getMerk());
```

Jadi untuk apakah sintaks p.getMobil().getMerk() yang ada di dalam *method* main tersebut?



```
/Library/Java/JavaVirtualMachines/jdk-11.0.2/bin/java -Djava.class.path=. MainPercobaan2
Biaya Total = 1100000
Avanza
Process finished with exit code 0
```

Sintaks p.getMobil().getMerk() dalam method main digunakan untuk mengambil merk dari objek Mobil yang dimiliki oleh objek Pelanggan (p) dan kemudian mencetak merk mobil

tersebut ke layar. Hasil cetakan dari pernyataan ini akan menampilkan merk mobil yang dimiliki oleh pelanggan.

Percobaan 3

```
package ac.id.polinema.relasiclass.percobaan3;

public class Pegawai {
    private String nip;
    private String nama;
    public Pegawai(String nip, String nama) {
        this.nip = nip;
        this.nama = nama;
    }
    public String getNip() {
        return nip;
    }
    public void setNip(String nip) {
        this.nip = nip;
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public String info() {
        String info = "";
        info += "Nip: " + this.nip + "\n";
        info += "Nama: " + this.nama + "\n";
        return info;
    }
}
```

```

package ac.id.polinema.relasticlass.percobaan3;

public class KeretaApi {
    private String nama;
    private String kelas;
    private Pegawai masinis;
    private Pegawai asisten;
    public KeretaApi(String nama, String kelas, Pegawai masinis) {
        this.nama = nama;
        this.kelas = kelas;
        this.masinis = masinis;
    }
    public KeretaApi(String nama, String kelas, Pegawai masinis, Pegawai
asisten) {
        this.nama = nama;
        this.kelas = kelas;
        this.masinis = masinis;
        this.asisten = asisten;
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public String getKelas() {
        return kelas;
    }
    public void setKelas(String kelas) {
        this.kelas = kelas;
    }
    public Pegawai getMasinis() {
        return masinis;
    }
    public void setMasinis(Pegawai masinis) {
        this.masinis = masinis;
    }
    public Pegawai getAsisten() {
        return asisten;
    }
    public void setAsisten(Pegawai asisten) {
        this.asisten = asisten;
    }
    public String info() {
        String info = "";
        info += "Nama: " + this.nama + "\n";
        info += "Kelas: " + this.kelas + "\n";
        info += "Masinis:\n" + this.masinis.info() + "\n";
        if (asisten != null) {
            info += "Asisten:\n" + this.asisten.info() + "\n";
        }
        return info;
    }
}

```

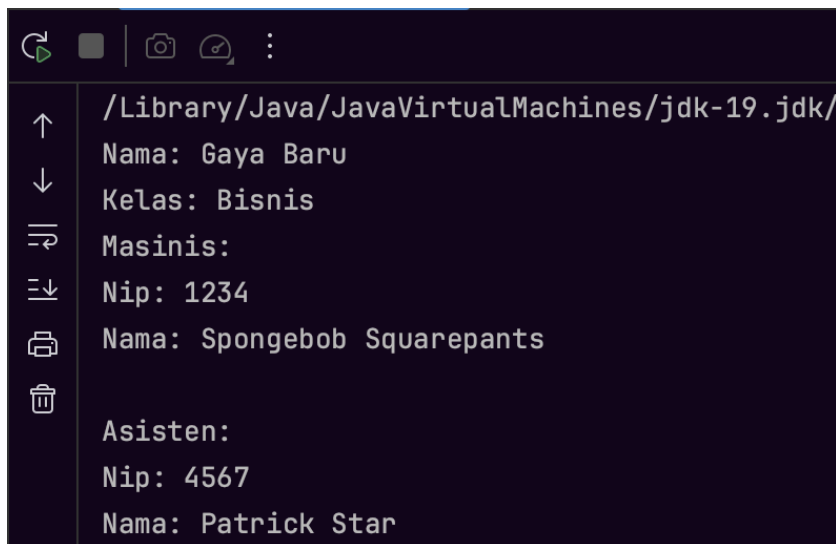
```

package ac.id.polinema.relasticlass.percobaan3;

public class MainPercobaan3 {
    public static void main(String[] args) {
        Pegawai masinis = new Pegawai("1234", "Spongebob Squarepants");
        Pegawai asisten = new Pegawai("4567", "Patrick Star");
        KeretaApi keretaApi = new KeretaApi("Gaya Baru", "Bisnis",
masinis, asisten);

        System.out.println(keretaApi.info());
    }
}

```



```
/Library/Java/JavaVirtualMachines/jdk-19.jdk/
Nama: Gaya Baru
Kelas: Bisnis
Masinis:
Nip: 1234
Nama: Spongebob Squarepants
Asisten:
Nip: 4567
Nama: Patrick Star
```

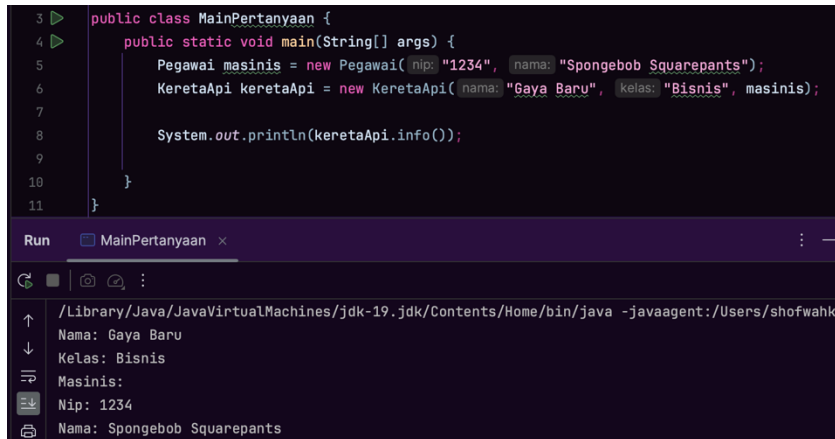
Pertanyaan

1. Di dalam *method* `info()` pada *class* `KeretaApi`, baris `this.masinis.info()` dan `this.asisten.info()` digunakan untuk apa ?

Baris `this.masinis.info()` dan `this.asisten.info()` dalam *method* `info()` pada *class* `KeretaApi` digunakan untuk mengambil informasi tentang masinis dan asisten yang merupakan objek dari *class* `Pegawai`. Untuk mengakses informasi seperti Nip dan Nama dari staf-staf tersebut dan memasukkannya ke dalam informasi kereta api yang akan dicetak.

Buatlah *main* program baru dengan nama *class* MainPertanyaan pada *package* yang sama. Tambahkan kode berikut pada *method* main()!

```
Pegawai masinis = new Pegawai("1234", "Spongebob  
Squarepants");  
KeretaApi keretaApi = new KeretaApi("Gaya Baru", "Bisnis", masinis);  
  
System.out.println(keretaApi.info());
```



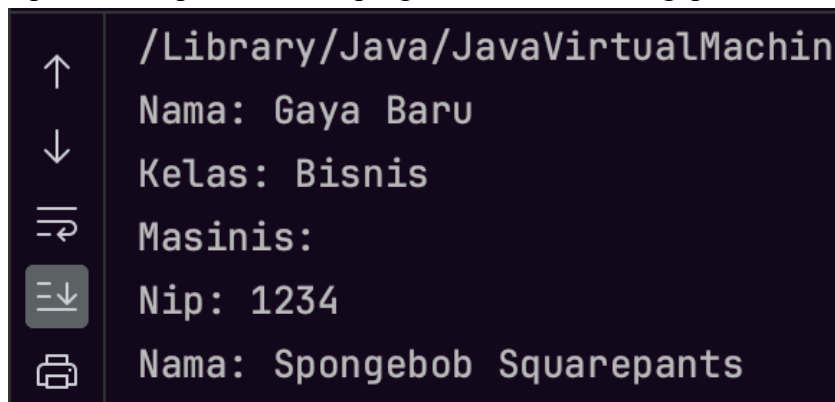
The screenshot shows an IDE with a Java file named MainPertanyaan.java. The code defines a class MainPertanyaan with a main method. Inside the main method, a Pegawai object named masinis is created with nip "1234" and name "Spongebob Squarepants". Then, a KeretaApi object named keretaApi is created with name "Gaya Baru", kelas "Bisnis", and the masinis object. Finally, the info() method of keretaApi is called, which prints the details of the KeretaApi and its associated Pegawai.

```
3 public class MainPertanyaan {  
4     public static void main(String[] args) {  
5         Pegawai masinis = new Pegawai( nip: "1234", nama: "Spongebob Squarepants");  
6         KeretaApi keretaApi = new KeretaApi( nama: "Gaya Baru", kelas: "Bisnis", masinis);  
7  
8         System.out.println(keretaApi.info());  
9     }  
10 }  
11 }
```

The Run window shows the output of the program:

```
Run MainPertanyaan x  
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java -javaagent:/Users/shofwahk  
Nama: Gaya Baru  
Kelas: Bisnis  
Masinis:  
Nip: 1234  
Nama: Spongebob Squarepants
```

2. Apa hasil output dari *main* program tersebut ? Mengapa hal tersebut dapat terjadi ?



The screenshot shows a terminal window with the output of the program. The output is as follows:

```
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java -javaagent:/Users/shofwahk  
Nama: Gaya Baru  
Kelas: Bisnis  
Masinis:  
Nip: 1234  
Nama: Spongebob Squarepants
```

Hal ini terjadi karena program menciptakan objek Pegawai (masinis) dan objek KeretaApi, kemudian mencetak informasi tentang kereta api beserta masinisnya.

3. Perbaiki *class* KeretaApisehingga program dapat berjalan !

```
2
3 ▶ public class MainPertanyaan {
4 ▶     public static void main(String[] args) {
5         Pegawai masinis = new Pegawai( nip: "1234", nama: "Spongebob Squarepants");
6         Pegawai asisten = new Pegawai( nip: "4567", nama: "Patrick Star");
7         KeretaApi keretaApi = new KeretaApi( nama: "Gaya Baru", kelas: "Bisnis", masinis, as
8
9         System.out.println(keretaApi.info());
10
11     }
```

Run MainPertanyaan x

↶ ■ | 📄 🔍 ⋮

↑ /Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java -javaagent:/Users/shofwahk
↓ Nama: Gaya Baru
⇅ Kelas: Bisnis
≡ Masinis:
📄 Nip: 1234
🗑 Nama: Spongebob Squarepants
Asisten:
Nip: 4567
Nama: Patrick Star

Percobaan 4

```
package ac.id.polinema.relasiclass.percobaan4;

public class Penumpang {
    private String ktp;
    private String nama;

    public Penumpang(String ktp, String nama) {
        this.ktp = ktp;
        this.nama = nama;
    }

    public void setKtp(String ktp) {
        this.ktp = ktp;
    }

    public String getKtp() {
        return ktp;
    }

    public void setName(String nama) {
        this.nama = nama;
    }

    public String getName() {
        return nama;
    }

    public String info() {
        return "Ktp: " + ktp + "\nNama: " + nama;
    }
}
```

```
package ac.id.polinema.relasticlass.percobaan4;
public class Kursi {
    private String nomor;
    private Penumpang penumpang;

    public Kursi(String nomor) {
        this.nomor = nomor;
    }

    public void setNomor(String nomor) {
        this.nomor = nomor;
    }

    public String getNomor() {
        return nomor;
    }

    public void setPenumpang(Penumpang penumpang) {
        this.penumpang = penumpang;
    }

    public Penumpang getPenumpang() {
        return penumpang;
    }

    public String info() {
        String info = "Nomor: " + nomor + "\n";
        if (penumpang != null) {
            info += "Penumpang: " + penumpang.info() + "\n";
        }
        return info;
    }
}
```

```

package ac.id.polinema.relasticlass.percobaan4;

public class Gerbong {
    private String kode;
    private Kursi[] arrayKursi;

    public Gerbong(String kode, int jumlah) {
        this.kode = kode;
        this.arrayKursi = new Kursi[jumlah];
        initKursi();
    }

    private void initKursi() {
        for (int i = 0; i < arrayKursi.length; i++) {
            this.arrayKursi[i] = new Kursi(String.valueOf(i + 1));
        }
    }

    public void setKode(String kode) {
        this.kode = kode;
    }

    public String getKode() {
        return kode;
    }

    public void setPenumpang(Penumpang penumpang, int nomor) {
        arrayKursi[nomor - 1].setPenumpang(penumpang);
    }

    public Kursi[] getArrayKursi() {
        return arrayKursi;
    }

    public String info() {
        String info = "Kode: " + kode + "\n";
        for (Kursi kursi : arrayKursi) {
            info += kursi.info();
        }
        return info;
    }
}

```

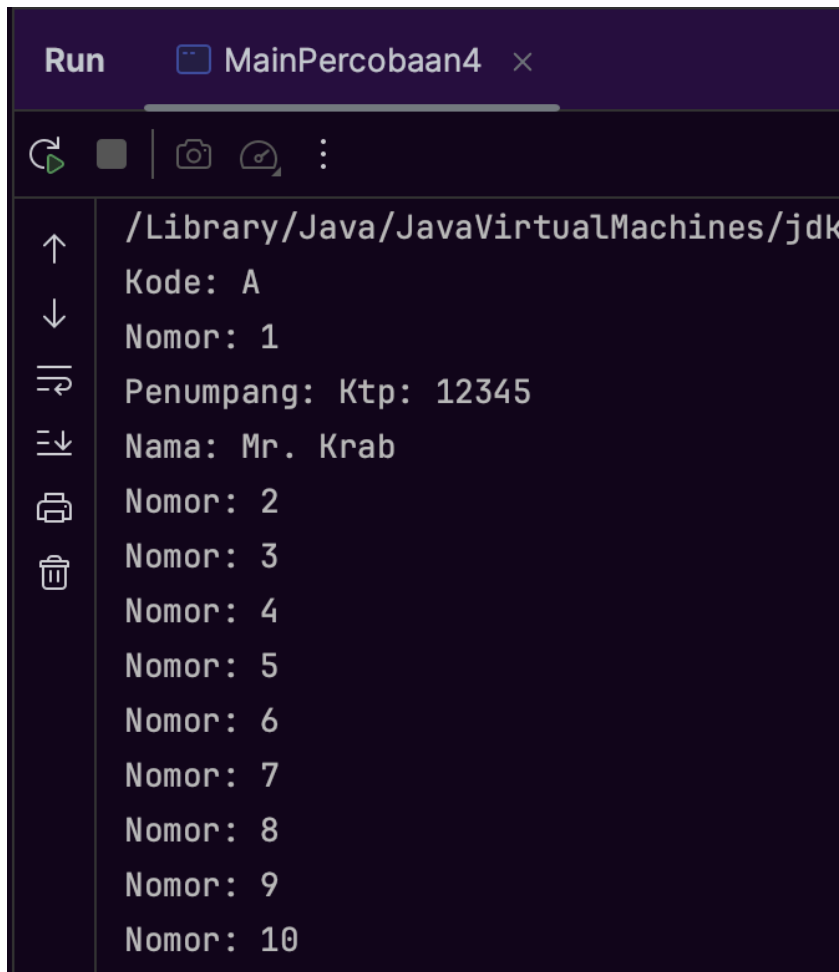
```

package ac.id.polinema.relasticlass.percobaan4;

public class MainPercobaan4 {
    public static void main(String[] args) {
        Penumpang p = new Penumpang("12345", "Mr. Krab");
        Gerbong gerbong = new Gerbong("A", 10);
        gerbong.setPenumpang(p, 1);

        System.out.println(gerbong.info());
    }
}

```

```
Run MainPercobaan4 x
/Library/Java/JavaVirtualMachines/jdk
Kode: A
Nomor: 1
Penumpang: Ktp: 12345
Nama: Mr. Krab
Nomor: 2
Nomor: 3
Nomor: 4
Nomor: 5
Nomor: 6
Nomor: 7
Nomor: 8
Nomor: 9
Nomor: 10
```

Pertanyaan

1. Pada *main* program dalam *class* MainPercobaan4, berapakah jumlah kursi dalam Gerbong A ?
Jumlah kursi dalam Gerbong A adalah 10.
2. Perhatikan potongan kode pada *method* info() dalam *class* Kursi. Apa maksud kode tersebut ?

```
...
if (this.penumpang != null) {
    info += "Penumpang: " + penumpang.info() + "\n";
}
...
```

Kode ini digunakan untuk memeriksa apakah ada penumpang di kursi tersebut sebelum mencetak informasi penumpang dalam kursi. Jika ada penumpang di kursi tersebut (penumpang tidak sama dengan null), maka informasi penumpang akan ditampilkan. Jika tidak ada penumpang, maka tidak akan ditampilkan informasi penumpang.

3. Mengapa pada *method* `setPenumpang()` dalam *class* `Gerbong`, nilai nomor dikurangi dengan angka 1 ?
- Nilai nomor dikurangi 1 karena indeks array dimulai dari 0, sedangkan nomor kursi dimulai dari 1. Sehingga, untuk mengakses array `Kursi` dengan nomor kursi yang benar, harus mengurangi nomor dengan 1.
4. Instansiasi objek baru `budi` dengan tipe `Penumpang`, kemudian masukkan objek baru tersebut pada `gerbong` dengan `gerbong.setPenumpang(budi, 1)`. Apakah yang terjadi ?
- Modifikasi program sehingga tidak diperkenankan untuk menduduki kursi yang sudah adapenumpang lain !

```
3 public class MainPercobaan4 {
4     public static void main(String[] args) {
5         Penumpang MrKrab = new Penumpang("12345", "Mr.Krab");
6         Penumpang Plankton = new Penumpang("12346", "Plankton");
7
8         Gerbong gerbongA = new Gerbong("A", 10);
9
10        gerbongA.setPenumpang(MrKrab, 1);
11        gerbongA.setPenumpang(Plankton, 1); // Kursi nomor 1 sudah ditempati
12
13        System.out.println(gerbongA.info());
14    }
15 }
```

Run MainPercobaan4

/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java -javaagent:/U
Kursi nomor 1 sudah ditempati.
Kode: A
Nomor: 1
Penumpang: Ktp: 12345
Nama: Mr.Krab
Nomor: 2
Nomor: 3
Nomor: 4
Nomor: 5
Nomor: 6
Nomor: 7
Nomor: 8
Nomor: 9
Nomor: 10

```
3 public class MainPercobaan4 {
4     public static void main(String[] args) {
5         Penumpang MrKrab = new Penumpang("12345", "Mr.Krab");
6         Penumpang Plankton= new Penumpang("12346", "Plankton");
7
8         Gerbong gerbongA = new Gerbong("A", 10);
9
10        gerbongA.setPenumpang(MrKrab, 1);
11        gerbongA.setPenumpang(Plankton, 1); // Kursi nomor 1 sudah ditempati
12
13        System.out.println(gerbongA.info());
14    }
15 }
```

Run MainPercobaan4 x

/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java -javaagent:/U
Kursi nomor 1 sudah ditempati.
Kode: A
Nomor: 1
Penumpang: Ktp: 12345
Nama: Mr.Krab
Nomor: 2
Nomor: 3
Nomor: 4
Nomor: 5
Nomor: 6
Nomor: 7
Nomor: 8
Nomor: 9
Nomor: 10

II. Tugas

Berdasarkan latihan di pertemuan teori, rancang dengan *class* diagram, kemudian implementasikan ke dalam program! Studi kasus harus mewakili relasi *class* dari percobaan-percobaan yang telah dilakukan pada materi ini, setidaknya melibatkan minimal 4 *class* (*class* yang berisi *main* tidak dihitung).

```
package ac.id.polinema.relasticlass.Tugas;

public class Mahasiswa {
    private String nim;
    private String nama;
    private MataKuliah[] mataKuliah;
    private int jumlahMataKuliah;

    public Mahasiswa(String nim, String nama, int maxJumlahMataKuliah) {
        this.nim = nim;
        this.nama = nama;
        this.mataKuliah = new MataKuliah[maxJumlahMataKuliah];
        this.jumlahMataKuliah = 0;
    }

    public void ambilMataKuliah(MataKuliah mk) {
        if (jumlahMataKuliah < mataKuliah.length) {
            mataKuliah[jumlahMataKuliah] = mk;
            jumlahMataKuliah++;
        } else {
            System.out.println("Maksimum jumlah mata kuliah tercapai.");
        }
    }

    public MataKuliah[] getMataKuliah() {
        return mataKuliah;
    }

    public String getNIM() {
        return nim;
    }

    public String getNama() {
        return nama;
    }

    public String info() {
        return "NIM: " + nim + "\nNama: " + nama;
    }
}
```

```
package ac.id.polinema.relasticlass.Tugas;

public class Dosen {
    private String nip;
    private String nama;
    private MataKuliah[] mataKuliahAjar;
    private int jumlahMataKuliah;

    public Dosen(String nip, String nama, int maxJumlahMataKuliah) {
        this.nip = nip;
        this.nama = nama;
        this.mataKuliahAjar = new MataKuliah[maxJumlahMataKuliah];
        this.jumlahMataKuliah = 0;
    }

    public void tambahMataKuliah(MataKuliah mk) {
        if (jumlahMataKuliah < mataKuliahAjar.length) {
            mataKuliahAjar[jumlahMataKuliah] = mk;
            jumlahMataKuliah++;
        } else {
            System.out.println("Maksimum jumlah mata kuliah ajar
tercapai.");
        }
    }

    public MataKuliah[] getMataKuliahAjar() {
        return mataKuliahAjar;
    }

    public String getNIP() {

        return nip;
    }

    public String getNama() {

        return nama;
    }

    public String info() {
        String infoDosen = "NIP: " + nip + "\nNama Dosen: " + nama +
"\nMata Kuliah yang Diajar:\n";
        for (int i = 0; i < jumlahMataKuliah; i++) {
            infoDosen += mataKuliahAjar[i].info() + "\n";
        }
        return infoDosen;
    }
}
```

```
package ac.id.polinema.relasticlass.Tugas;

public class MataKuliah {
    private String kode;
    private String namaMataKuliah;

    public MataKuliah(String kode, String namaMataKuliah) {
        this.kode = kode;
        this.namaMataKuliah = namaMataKuliah;
    }

    public String getKode() {
        return kode;
    }

    public String getNama() {
        return namaMataKuliah;
    }

    public String info() {
        return "Kode MK: " + kode + "\nNama MK: " + namaMataKuliah ;
    }
}
```

```
package ac.id.polinema.relasticlass.Tugas;

public class Pendaftaran {
    private Mahasiswa[] mahasiswa;
    private int jumlahMahasiswa;

    public Pendaftaran(int maxJumlahMahasiswa) {
        mahasiswa = new Mahasiswa[maxJumlahMahasiswa];
        jumlahMahasiswa = 0;
    }

    public void daftarMahasiswa(Mahasiswa mahasiswa) {
        if (jumlahMahasiswa < mahasiswa.length) {
            mahasiswa[jumlahMahasiswa] = mahasiswa;
            jumlahMahasiswa++;
        } else {
            System.out.println("Maksimum jumlah mahasiswa tercapai.");
        }
    }

    public Mahasiswa[] getMahasiswa() {
        return mahasiswa;
    }
}
```

```

package ac.id.polinema.relasticlass.Tugas;

public class MainPercobaan {
    public static void main(String[] args) {
        // Membuat objek MataKuliah
        MataKuliah mk1 = new MataKuliah("MK101", "Pemrograman Java");
        MataKuliah mk2 = new MataKuliah("MK102", "Basis Data");
        MataKuliah mk3 = new MataKuliah("MK103", "Pemrograman Web");

        // Membuat objek Mahasiswa
        Mahasiswa mhs1 = new Mahasiswa("12345", "Shofwah", 3);
        mhs1.ambilMataKuliah(mk1);
        mhs1.ambilMataKuliah(mk2);

        // Membuat objek Dosen
        Dosen dosen1 = new Dosen("67890", "Kanaka", 2);
        dosen1.tambahMataKuliah(mk1);
        dosen1.tambahMataKuliah(mk3);

        // Membuat objek Pendaftaran
        Pendaftaran pendaftaran = new Pendaftaran(2);
        pendaftaran.daftarMahasiswa(mhs1);

        // Menampilkan informasi mahasiswa, mata kuliah, dosen, dan
        pendaftaran
        System.out.println("Informasi Mahasiswa:");
        System.out.println(mhs1.info());

        System.out.println("\nInformasi Dosen:");
        System.out.println(dosen1.info());

        System.out.println("\nInformasi Pendaftaran:");
        for (Mahasiswa mahasiswa : pendaftaran.getMahasiswa()) {
            if (mahasiswa != null) {
                System.out.println(mahasiswa.info());
            }
        }
    }
}

```

```
Run MainPercobaan x
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/H
Informasi Mahasiswa:
NIM: 12345
Nama: Shofwah

Informasi Dosen:
NIP: 67890
Nama Dosen: Kanaka
Mata Kuliah yang Diajar:
Kode MK: MK101
Nama MK: Pemrograman Java
Kode MK: MK103
Nama MK: Pemrograman Web

Informasi Pendaftaran:
NIM: 12345
Nama: Shofwah

Process finished with exit code 0
```