



OBJECT ORIENTED PROGRAM

Jobsheet 4 – Class Relation



AYU JOVITA WIDYADHARI
2241720219/04/21

1st Practicum

Processor class

```
package js4.relasiclass.percobaan1;

public class Processor {
    private String merk;
    private double cache;

    public Processor() {
    }

    public Processor(String merk, double cache) {
        this.merk = merk;
        this.cache = cache;
    }

    public void setMerk(String merk) {
        this.merk = merk;
    }

    public String getMerk() {
        return merk;
    }

    public void setCache(double cache) {
        this.cache = cache;
    }

    public double getCache() {
        return cache;
    }

    public void info() {
        System.out.printf(format: "Merk Processor = %s\n", args: merk);
        System.out.printf(format: "Cache Memory   = %s\n", args: merk);
    }
}
```

Laptop class

```

package js4.relasticlass.percobaan1;
public class Laptop {
    private String merk;
    private Processor proc;

    public Laptop() {
    }

    public Laptop(String merk, Processor proc) {
        this.merk = merk;
        this.proc = proc;
    }

    public void setMerk(String merk) {
        this.merk = merk;
    }

    public String getMerk() {
        return merk;
    }

    public void setProc(Processor p) {
        this.proc = p;
    }

    public Processor getProc() {
        return proc;
    }

    public void info() {
        System.out.println("Merk Laptop = " + merk);
        proc.info();
    }
}

```

Main Class

```

package js4.relasticlass.percobaan1;
public class MainPercobaan1 {
    public static void main(String[] args) {
        Processor p = new Processor(merk: "Intel i5", cache: 3);
        Laptop L = new Laptop(merk: "Thinkpad", proc: p);
        L.info();

        Processor p1 = new Processor();
        p1.setMerk(merk: "Intel i5");
        p1.setCache(cache: 4);
        Laptop L1 = new Laptop();
        L1.setMerk(merk: "Thinkpad");
        L1.setProc(p: p1);
        L1.info();
    }
}

```

Result

```
Output - js4..relasiclass.percobaan1 (run)
run:
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = Intel i5
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = Intel i5
BUILD SUCCESSFUL (total time: 0 seconds)
```

Questions

1. What are the uses of setter and getter methods in class Processor and class Laptop?

- **Setter Methods:** A class's private attributes can be set using setter methods. Setter methods like `setMerk()` and `setCache()` are used in the context of the Processor class and the Laptop class, respectively, to set the brand and cache attribute values in the former and the brand and proc attribute values in the latter.
- **Getter Methods:** Private attribute values are retrieved from a class using getter methods. Getter methods like `getMerk()`, `getCache()`, and `getProc()` are used in the context of the Processor class and Laptop class to retrieve the brand and cache attribute values from the Processor class and the brand and Processor object attribute values from the Laptop class, respectively.

2. What are the different uses of default constructors and parameterized constructors?

- **Default Constructor:** A default constructor is a parameterless constructor that Java automatically creates if you don't provide a custom constructor in the class. It accepts no arguments and can be used to create objects without assigning initial values to class attributes.
- **Parameterized Constructors:** A parameterized constructor is a constructor that accepts one or more parameters. In the context of the Processor class and Laptop class, parameterized constructors are used to initialize the values of the class attributes when the object is created. By using a parameterized constructor, you can initialize the values of the class attributes right from the start when the object is created.

3. Which attribute is of type object in class Laptop?

The proc attribute of class Laptop is of type object. In class Laptop, proc is an object of class Processor.

4. Which line indicates that class Laptop is related to class Processor?

Class Laptop's private Processor proc; line demonstrates its connection to class Processor. The link between the Laptop class and the Processor class is shown by the proc property in the Laptop class, which is an object of type Processor.

5. What is the use of the `proc.info()` syntax in the Laptop class?

The `proc.info()` syntax is used to call the `info()` method of the Processor object stored in the `proc` attribute of the Laptop class. This way, information about the Processor object associated with the Laptop object can be printed to the console.

6. What does `p` mean in the Laptop code line `l = new Laptop("Thinkpad", p);`?

In the context of that line of code, `p` is an object of the Processor class that has been previously declared with the values "Intel i5" for the brand attribute and 3 for the cache attribute.

What happens if the line of code is changed to:

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

If that line of code is changed, a new Processor object will be created directly as an argument when creating the Laptop object. The result remains the same, that is, a Laptop object with the brand name "Thinkpad" and a Processor object with the brand name "Intel i5" and cache 3 will be created and printed using the `info()` method.

2nd Practicum

Mobil class

```
package js4.relasticlass.percobaan2;
public class Mobil {
    private String merk;
    private int biaya;

    public Mobil() {
    }

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

    public String getNama() {
        return merk;
    }

    public void setBiaya(int biaya) {
        this.biaya = biaya;
    }

    public int getBiaya() {
        return biaya;
    }

    public int hitungBiayaMobil(int hari) {
        return biaya * hari;
    }
}
```

Sopir class

```
package js4.relasticlass.percobaan2;

public class Sopir {
    private String driver;
    private int biaya;

    public Sopir() {
    }

    public void setNama(String driver) {
        this.driver = driver;
    }

    public String getNama() {
        return driver;
    }

    public void setBiaya(int biaya) {
        this.biaya = biaya;
    }

    public int getBiaya() {
        return biaya;
    }

    public int hitungBiayaSopir(int hari) {
        return hari * biaya;
    }
}
```

Pelanggan class

```
package js4.relasticlass.percobaan2;
public class Pelanggan {
    private String nama;
    private Mobil mobil;
    private Sopir sopir;
    private int hari;

    public Pelanggan() {
    }

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

    public String getNama() {
        return nama;
    }

    public void setMobil(Mobil mobil) {
        this.mobil = mobil;
    }

    public Mobil getMobil() {
        return mobil;
    }

    public void setSopir(Sopir sopir) {
        this.sopir = sopir;
    }

    public Sopir getSopir() {
        return sopir;
    }

    public void setHari(int hari) {
        this.hari = hari;
    }
}
```

```
public int hitungBiayaTotal() {
    return mobil.hitungBiayaMobil(hari) + sopir.hitungBiayaSopir(hari);
}
```

Percobaan 2 main class

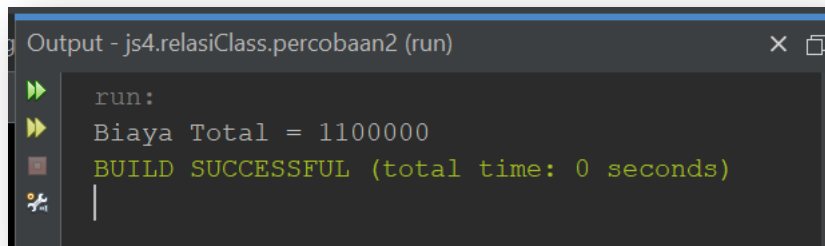
```
package js4.relasticlass.percobaan2;
public class mainPercobaan2 {
    public static void main(String[] args) {
        Mobil m = new Mobil();
        m.setNama(nama: "Avanza");
        m.setBiaya(biaya: 350000);

        Sopir s = new Sopir();
        s.setNama(driver: "John Doe");
        s.setBiaya(biaya: 200000);

        Pelanggan p = new Pelanggan();
        p.setNama(nama: "Jane Doe");
        p.setMobil(mobil: m);
        p.setSopir(sopir: s);
        p.setHari(hari: 2);

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

Result



```
Output - js4.relatiClass.percobaan2 (run)
run:
Biaya Total = 1100000
BUILD SUCCESSFUL (total time: 0 seconds)
```

Questions

1. ****Relasi dengan class Mobil dan class Sopir di class Pelanggan:****

- Relasi dengan class Mobil terdapat pada baris ``private Mobil mobil;``, yang menunjukkan bahwa class Pelanggan memiliki atribut objek dari class Mobil.
- Relasi dengan class Sopir terdapat pada baris ``private Sopir sopir;``, yang menunjukkan bahwa class Pelanggan memiliki atribut objek dari class Sopir.

2. ****Mengapa method `hitungBiayaSopir` dan `hitungBiayaMobil` memiliki argumen `hari`?**

- Kedua method tersebut membutuhkan argumen ``hari`` karena biaya sopir dan biaya sewa mobil dihitung per hari. Dengan menyertakan argumen ``hari``, method dapat menghitung biaya berdasarkan jumlah hari yang diberikan.

3. ****Fungsi dari ``mobil.hitungBiayaMobil(hari)`` dan ``sopir.hitungBiayaSopir(hari)``:**

- ``mobil.hitungBiayaMobil(hari)`` digunakan untuk menghitung biaya sewa mobil selama ``hari`` hari berdasarkan biaya per hari yang telah ditetapkan untuk mobil tersebut.
- ``sopir.hitungBiayaSopir(hari)`` digunakan untuk menghitung biaya sopir selama ``hari`` hari berdasarkan biaya per hari yang telah ditetapkan untuk sopir tersebut.

4. ****Sintaks ``p.setMobil(m)`` dan ``p.setSopir(s)``:**

- ``p.setMobil(m)`` digunakan untuk mengatur objek mobil (``m``) ke dalam atribut ``mobil`` dari objek pelanggan (``p``).
- ``p.setSopir(s)`` digunakan untuk mengatur objek sopir (``s``) ke dalam atribut ``sopir`` dari objek pelanggan (``p``).

5. ****Proses ``p.hitungBiayaTotal()``:**

- Proses ini menghitung biaya total yang harus dibayar oleh pelanggan. ``p.hitungBiayaTotal()`` memanggil method ``hitungBiayaMobil(hari)`` dari objek ``mobil`` dan ``hitungBiayaSopir(hari)`` dari objek ``sopir``, lalu menjumlahkan hasil perhitungan tersebut.

6. **Sintaks `p.getMobil().getMerk()`.**

- Sintaks ini mengambil nilai atribut `merk` dari objek `mobil` yang dimiliki oleh objek pelanggan (`p`). Dalam konteks method `main`, sintaks tersebut mencetak merk mobil yang disewa oleh pelanggan (`p`) ke konsol.

3rd Practicum

Pegawai class

```
5 package js4.relasticlass.percobaan3;
6 public class Pegawai {
7     private String nip;
8     private String nama;
9
10    public Pegawai(String nip, String nama){
11        this.nip = nip;
12        this.nama = nama;
13    }
14    public void setNip(String nip){
15        this.nip = nip;
16    }
17    public String getNip(){
18        return nip;
19    }
20    public void setName(String nama){
21        this.nama = nama;
22    }
23    public String getNama(){
24        return nama;
25    }
26    public String info(){
27        String info = "";
28        info += "\nNIP : " + this.nip + "\n";
29        info += "Name: " + this.nama + "\n";
30        return info;
31    }
32 }
```

KeretaApi class

```
5 package js4.relasticlass.percobaan3;
6 public class KeretaApi {
7     private String nama;
8     private String kelas;
9     private Pegawai masinis;
10    private Pegawai asisten;
11
12    public KeretaApi(String nama, String kelas, Pegawai masinis) {
13        this.nama = nama;
14        this.kelas = kelas;
15        this.masinis = masinis;
16    }
17
18    public KeretaApi(String nama, String kelas, Pegawai masinis, Pegawai asisten) {
19        this.nama = nama;
20        this.kelas = kelas;
21        this.masinis = masinis;
22        this.asisten = asisten;
23    }
24
25    public void setNama(String nama) {
26        this.nama = nama;
27    }
28
29    public String getNama() {
30        return nama;
31    }
32
```

```
33    public void setKelas(String kelas) {
34        this.kelas = kelas;
35    }
36
37    public String getKelas() {
38        return kelas;
39    }
40
41    public void setMasinis(Pegawai masinis) {
42        this.masinis = masinis;
43    }
44
45    public Pegawai getMasinis() {
46        return masinis;
47    }
48
49    public void setAsisten(Pegawai asisten) {
50        this.asisten = asisten;
51    }
52
53    public Pegawai getAsisten() {
54        return asisten;
55    }
56
```

```

56
57     public String info() {
58         String info = "";
59         info += "Nama Kereta: " + this.nama + "\n";
60         info += "Kelas      : " + this.kelas + "\n";
61         info += "Masinis      " + this.masinis.info() + "\n";
62         if (this.asisten != null) {
63             info += "Asisten    " + this.asisten.info() + "\n";
64         }
65         return info;
66     }
67 }

```

Main Class

```

5  package js4.relasticlass.percobaan3;
6  public class MainPercobaan3 {
7      public static void main(String[] args) {
8          Pegawai masinis = new Pegawai(nip:"1234", nama:"Spongebob Squarepants");
9          Pegawai asisten = new Pegawai(nip:"4567", nama:"Patrick Star");
10         KeretaApi keretaApi = new KeretaApi(nama:"Gaya Baru", kelas:"Bisnis", masinis, asisten);
11         System.out.println(x: keretaApi.info());
12     }
13 }
14

```

Result

```

Output - js4.relasticlass.percobaan3 (run)
run:
Nama Kereta: Gaya Baru
Kelas      : Bisnis
Masinis
NIP : 1234
Name: Spongebob Squarepants

Asisten
NIP : 4567
Name: Patrick Star

BUILD SUCCESSFUL (total time: 0 seconds)

```

Questions

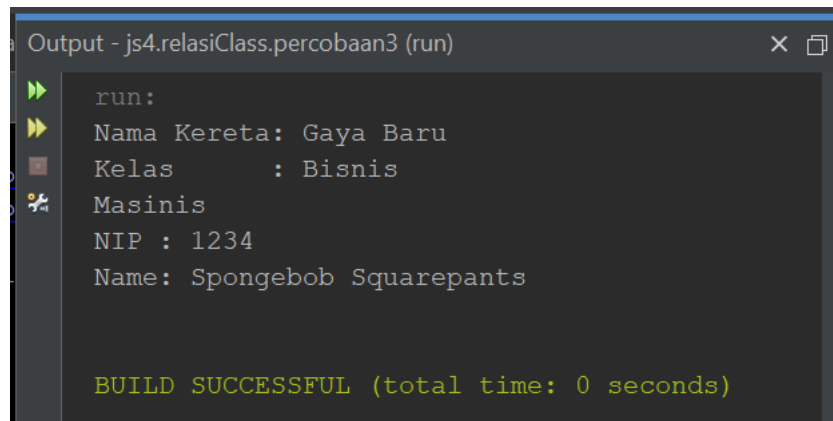
1. Dalam method **info()** pada class **KeretaApi**, baris **this.masinis.info()** digunakan untuk mendapatkan informasi (NIP dan nama) tentang masinis kereta api, sedangkan **this.asisten.info()** digunakan untuk mendapatkan informasi (NIP dan nama) tentang asisten masinis. Dengan menggunakan metode **info()** dari class **Pegawai**, informasi tentang masinis dan asisten masinis dapat dicetak ke dalam string yang akan dikembalikan oleh method **info()** dari class **KeretaApi**.
2. 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());

```

11 public class MainPertanyaan {
12     public static void main(String[] args) {
13         Pegawai masinis = new Pegawai(nip:"1234", nama:"Spongebob Squarepants");
14         KeretaApi keretaApi = new KeretaApi(nama:"Gaya Baru", kelas:"Bisnis", masinis);
15         System.out.println(x: keretaApi.info());
16     }
17 }
18
19

```

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



```

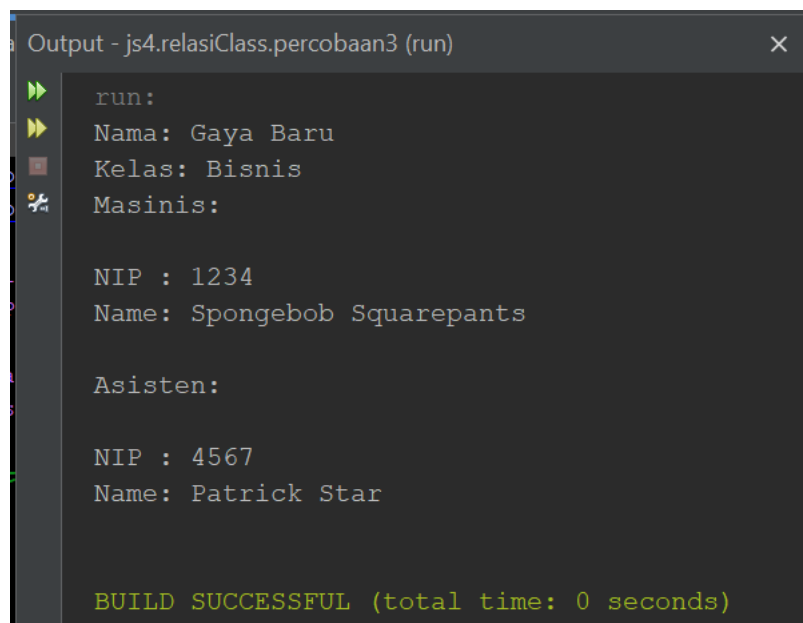
Output - js4.relasiClass.percobaan3 (run)
run:
Nama Kereta: Gaya Baru
Kelas      : Bisnis
Masinis
NIP : 1234
Name: Spongebob Squarepants

BUILD SUCCESSFUL (total time: 0 seconds)

```

Output ini terjadi karena asisten tidak diinisialisasi dengan nilai (null) saat membuat objek **KeretaApi**. Oleh karena itu, dalam outputnya, atribut **asisten** dicetak sebagai **null**.

4. Perbaiki class KeretaApi sehingga program dapat berjalan !



```

Output - js4.relasiClass.percobaan3 (run)
run:
Nama: Gaya Baru
Kelas: Bisnis
Masinis:

NIP : 1234
Name: Spongebob Squarepants

Asisten:

NIP : 4567
Name: Patrick Star

BUILD SUCCESSFUL (total time: 0 seconds)

```

4th Practicum

Penumpang Class

```
6 public class Penumpang {
7     private String ktp;
8     private String nama;
9
10    public Penumpang(String ktp, String nama) {
11        this.ktp = ktp;
12        this.nama = nama;
13    }
14
15    public void setKtp(String ktp) {
16        this.ktp = ktp;
17    }
18
19    public String getKtp() {
20        return ktp;
21    }
22
23    public void setName(String nama) {
24        this.nama = nama;
25    }
26
27    public String getName() {
28        return nama;
29    }
30
31    public String info() {
32        String info = "";
33        info += "Ktp: " + ktp + "\n";
34        info += "Nama: " + nama + "\n";
35        return info;
36    }
```

Kursi Class

```
6   public class Kursi {
7       String nomor;
8       Penumpang penumpang;
9
10      public Kursi(String nomor) {
11          this.nomor = nomor;
12      }
13      public void setNomor(String nomor) {
14          this.nomor = nomor;
15      }
16      public String getNomor() {
17          return nomor;
18      }
19      public void setPenumpang(Penumpang penumpang) {
20          this.penumpang = penumpang;
21      }
22      public Penumpang getPenumpang() {
23          return penumpang;
24      }
25      public String info() {
26          String info = "";
27          info += "Nomor: " + nomor + "\n";
28          if (this.penumpang != null) {
29              info += "Penumpang: " + penumpang.info() + "\n";
30          }
31          return info;
32      }
33  }
34
```

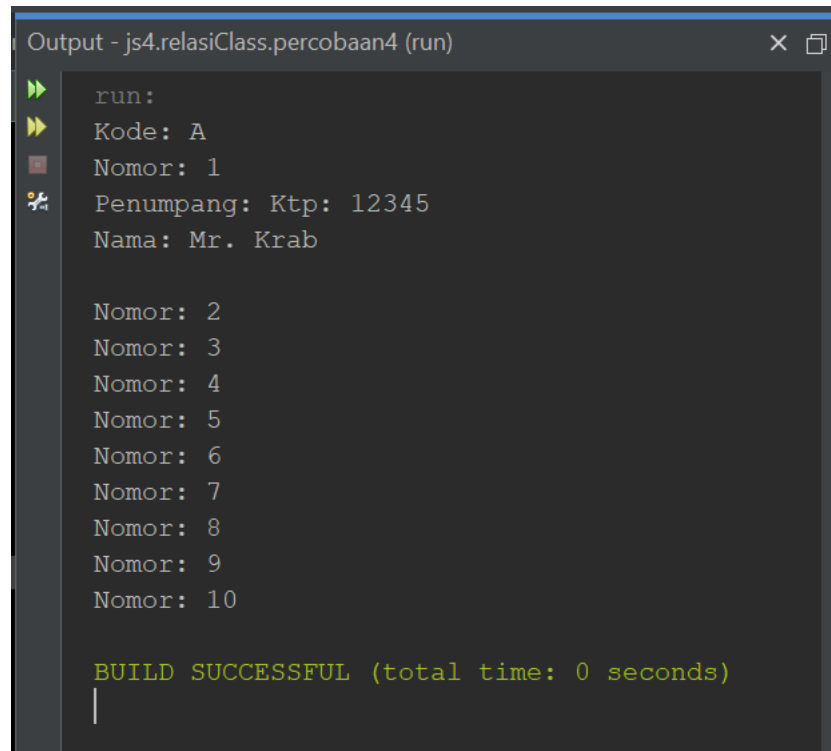
Gerbong Class

```
5  package js4.relasiclass.percobaan4;
6  public class Gerbong {
7      String kode;
8      Kursi[] arrayKursi;
9
10     public Gerbong(String kode, int jumlah) {
11         this.kode = kode;
12         this.arrayKursi = new Kursi[jumlah];
13         this.initKursi();
14     }
15
16     private void initKursi() {
17         for (int i = 0; i < arrayKursi.length; i++) {
18             this.arrayKursi[i] = new Kursi(nomor: String.valueOf(i + 1));
19         }
20     }
21
22     public void setPenumpang(Penumpang penumpang, int nomor) {
23         this.arrayKursi[nomor - 1].setPenumpang(penumpang);
24     }
25
26     public String info() {
27         String info = "Kode: " + kode + "\n";
28         for (Kursi kursi : arrayKursi) {
29             info += kursi.info();
30         }
31         return info;
32     }
33 }
34
```

Main Class

```
5  package js4.relasiclass.percobaan4;
6  public class MainPercobaan4 {
7      public static void main(String[] args) {
8          Penumpang p = new Penumpang(ktp:"12345", nama:"Mr. Krab");
9          Gerbong gerbong = new Gerbong(kode:"A", jumlah: 10);
10         gerbong.setPenumpang(penumpang: p, nomor: 1);
11         System.out.println(x: gerbong.info());
12     }
13 }
14
```

Result



```
Output - js4.relatiClass.percobaan4 (run)

run:
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

BUILD SUCCESSFUL (total time: 0 seconds)
```

Questions

1. Pada main program dalam class **MainPercobaan4**, berapakah jumlah kursi dalam **Gerbang A** ?

Dalam main program class **MainPercobaan4**, gerbang A memiliki 10 kursi, karena parameter jumlah yang diberikan saat membuat objek **Gerbang** adalah 10 (**Gerbang gerbang = new Gerbang("A", 10);**).

2. Perhatikan potongan kode pada method **info()** dalam class **Kursi**. Apa maksud kode tersebut ? ... **if (this.penumpang != null) { info += "Penumpang: " + penumpang.info() + "\n"; } ...**

Potongan kode pada method **info()** dalam class **Kursi** digunakan untuk memeriksa apakah kursi memiliki penumpang atau tidak. Jika kursi memiliki penumpang (nilai **penumpang** tidak null), maka informasi penumpang akan ditambahkan ke string info. Jika kursi belum ditempati oleh penumpang, potongan kode ini tidak akan menambahkan informasi penumpang ke string info.

3. Mengapa pada method **setPenumpang()** dalam class **Gerbang**, nilai nomor dikurangi dengan angka 1 ?

Pada method **setPenumpang()** dalam class **Gerbang**, nilai nomor dikurangi dengan angka 1 karena array dimulai dari indeks 0, bukan dari 1. Dalam representasi array, kursi pertama akan memiliki indeks 0, kursi kedua akan memiliki indeks 1, dan seterusnya. Jadi, ketika penumpang mengatakan nomor kursi 1, itu sebenarnya merujuk ke indeks 0 dalam array kursi.

4. Instansiasi objek baru **budi** dengan tipe **Penumpang**, kemudian masukkan objek baru tersebut pada gerbong dengan **gerbong.setPenumpang(budi, 1)**. Apakah yang Page 10 of 10 terjadi ?

Ketika Anda melakukan instansiasi objek baru **budi** dengan tipe **Penumpang** dan kemudian memasukkannya ke dalam gerbong dengan **gerbong.setPenumpang(budi, 1)**, objek penumpang **budi** akan ditempatkan di kursi nomor 1 dalam gerbong. Namun, jika kursi nomor 1 sudah ditempati oleh penumpang lain, objek penumpang yang sebelumnya ada di kursi nomor 1 akan digantikan oleh objek penumpang **budi**.

5. Modifikasi program sehingga tidak diperkenankan untuk menduduki kursi yang sudah ada penumpang lain !

```
22 public void setPenumpang(Penumpang penumpang, int nomor) {
23     if (this.arrayKursi[nomor - 1].getPenumpang() == null) {
24         this.arrayKursi[nomor - 1].setPenumpang(penumpang);
25         System.out.println("Penumpang berhasil ditempatkan pada kursi nomor " + nomor);
26     } else {
27         System.out.println("Maaf, kursi nomor " + nomor + " sudah ditempati.");
28     }
29 }
```

```
Output - js4.relatiClass.percobaan4 (run)
run:
Penumpang berhasil ditempatkan pada kursi nomor 1
Maaf, 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

BUILD SUCCESSFUL (total time: 0 seconds)
```

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).

```

5 package js.relaiclass.assignment;
6 public class Employee {
7     private String nip;
8     private String nama;
9
10    public void setNip(String nip) {
11        this.nip = nip;
12    }
13
14    public String getNip() {
15        return nip;
16    }
17
18    public void setNama(String nama) {
19        this.nama = nama;
20    }
21
22    public String getNama() {
23        return nama;
24    }
25 }
26

```

```

6 public class Proyek {
7     private String idProyek;
8     private String namaProyek;
9     private Employee[] pegawai;
10    private int jumlahPegawai;
11
12    public void setIDProyek(String idProyek) {
13        this.idProyek = idProyek;
14    }
15
16    public String getIDProyek() {
17        return idProyek;
18    }
19
20    public void setNamaProyek(String namaProyek) {
21        this.namaProyek = namaProyek;
22    }
23
24    public String getNamaProyek() {
25        return namaProyek;
26    }
27
28    public void addPegawai(Employee pegawai) {
29        // Logika untuk menambah pegawai ke dalam array pegawai
30    }
31
32    public int getJumlahPegawai() {
33        return jumlahPegawai;
34    }
35 }
36

```

```
5 package js.telasclass.assignment;
6 public class Tugas {
7     private String idTugas;
8     private String deskripsi;
9     private String deadline;
10
11     public void setIDTugas(String idTugas) {
12         this.idTugas = idTugas;
13     }
14
15     public String getIDTugas() {
16         return idTugas;
17     }
18
19     public void setDeskripsi(String deskripsi) {
20         this.deskripsi = deskripsi;
21     }
22
23     public String getDeskripsi() {
24         return deskripsi;
25     }
26
27     public void setDeadline(String deadline) {
28         this.deadline = deadline;
29     }
30
31     public String getDeadline() {
32         return deadline;
33     }
34 }
35
```

```

6 public class MainProgram {
7     public static void main(String[] args) {
8         // Contoh penggunaan class Pegawai, Proyek, dan Tugas
9         Employee pegawai1 = new Employee();
10        pegawai1.setNip(nip: "12345");
11        pegawai1.setNama(nama: "Sandy Squirrel");
12
13        Proyek proyek1 = new Proyek();
14        proyek1.setIDProyek(idProyek: "P001");
15        proyek1.setNamaProyek(namaProyek: "Proyek A");
16        proyek1.addPegawai(pegawai: pegawai1);
17
18        Tugas tugas1 = new Tugas();
19        tugas1.setIDTugas(idTugas: "T001");
20        tugas1.setDeskripsi(deskripsi: "Melakukan analisis kebutuhan proyek");
21        tugas1.setDeadline(deadline: "2023-12-31");
22
23        System.out.println(x: "Informasi Pegawai:");
24        System.out.println("NIP: " + pegawai1.getNip());
25        System.out.println("Nama: " + pegawai1.getNama());
26
27        System.out.println(x: "\nInformasi Proyek:");
28        System.out.println("ID Proyek: " + proyek1.getIDProyek());
29        System.out.println("Nama Proyek: " + proyek1.getNamaProyek());
30        System.out.println("Jumlah Pegawai: " + proyek1.getJumlahPegawai());
31
32        System.out.println(x: "\nInformasi Tugas:");
33        System.out.println("ID Tugas: " + tugas1.getIDTugas());
34        System.out.println("Deskripsi: " + tugas1.getDeskripsi());
35        System.out.println("Deadline: " + tugas1.getDeadline());
36    }

```

```

Output - js.RelasiClass.Assignment (run)
run:
Informasi Pegawai:
NIP: 12345
Nama: Sandy Squirrel

Informasi Proyek:
ID Proyek: P001
Nama Proyek: Proyek A
Jumlah Pegawai: 0

Informasi Tugas:
ID Tugas: T001
Deskripsi: Melakukan analisis kebutuhan proyek
Deadline: 2023-12-31
BUILD SUCCESSFUL (total time: 0 seconds)

```