LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
PERTEMUAN 5

2022

Praktikan

2141762056

Margaretha Violina Putri Purnomo

SIB 2F /11

DAFTAR ISI

1. Kompetensi

Setelah menempuh pokok bahasan ini, mahasiswa mampu:

1. Memahami konsep relasi kelas;
2. Mengimplementasikan relasi *has-a* dalam program.

Link Github

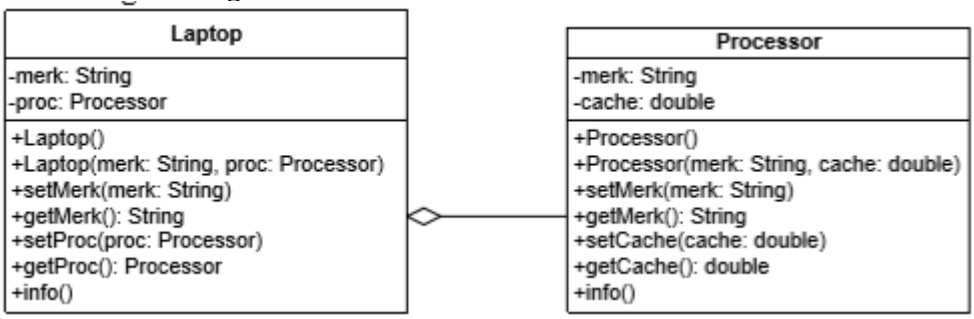
https://github.com/MargarethaViolinaPutri/PBO-

2. Praktikum

2.1 Percobaan 1

2.1.1 Langkah-Langkah Percobaan

a. Perhatikan diagram class berikut:



b. Buka *project* baru di *Netbeans* dan buat *package* dengan format berikut: `<identifier>.relasiclass.percobaan1` (ganti `<identifier>` dengan identitas anda atau nama domain), Contoh: `ac.id.polinema`, `jti.polinema`, dan sebagainya).

Catatan: Penamaan *package* dengan tambahan identifier untuk menghindari adanya kemungkinan penamaan *class* yang bentrok.

c. Buatlah class `Processor` dalam *package* tersebut.

```
11  public class Processor {
12
13  }
```

d. Tambahkan atribut merk dan *cache* pada `class Processor` dengan akses modifier

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

e. Buatlah *constructor default* untuk `class Processor`

```
Processor(){

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

f. Buatlah *constructor* untuk `class Processor` dengan parameter merk dan *cache*

g. Implementasikan setter dan getter untuk `class Processor`.

```
public void setMerk(String merk){
    this.merk = merk;
}
public void setCache(double cache){
    this.cache = cache;
}
public String getMerk(){
    return merk;
}
public double getCache(){
    return cache;
}
```

h. Implementasikan *method* `info()` seperti berikut:

```
public void info(){
    System.out.printf("Merk Processor = %s\n", merk);
    System.out.printf("Cache Memory = %.2f\n", cache);
}
```

i. Kemudian buatlah class Laptop di dalam package yang telah anda buat.

j. Tambahkan atribut merk dengan tipe String dan proc dengan tipe Object Processor

```
public class Laptop {
    private String merk;
    private Processor proc;
```

k. Buatlah *constructor* default untuk *class* Laptop

```
Laptop(){

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

l. Buatlah *constructor* untuk *class* Laptop dengan parameter `merk` dan `proc`

```
public void setMerk(String merk){
    this.merk = merk;
}
public void setProc(Processor proc){
    this.proc = proc;
}
public String getMerk(){
    return merk;
}
public Processor getProc(){
    return proc;
}
```

m. Selanjutnya implementasikan method `info()` pada *class* Laptop sebagai berikut

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

n. Pada *package* yang sama, buatlah `class MainPercobaan1` yang berisi method `main()`.

o. Deklarasikan Object `Processor` dengan nama `p` kemudian instansiasi dengan informasiatribut Intel i5 untuk nilai merk serta 3 untuk nilai *cache* .

```
Processor p = new Processor("Intel i5", 3);
```

p. Kemudian deklarasikan serta instansiasi Objek Laptop dengan nama L dengan informasiatribut `Thinkpad` dan Objek `Processor` yang telah dibuat.

```
Laptop L = new Laptop("Thinkpad", p);
```

q. Panggil method `info()` dari

objek L

```
L.info();
```

r. Tambahkan baris kode berikut

```
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();
```

s. *Compile* kemudian *run class* `MainPercobaan1`, akan didapatkan hasil seperti berikut:

```
--- exec-maven-plugin:3.0.0:exec (defau
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 3.00
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 4.00
------------------------------------------
BUILD SUCCESS
------------------------------------------
Total time:  3.318 s
Finished at: 2022-10-13T20:21:41+07:00
------------------------------------------
```

2.1.2 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 ?

Jawab : Method setter untuk memberikan nilai pada atribut yang bersifat private sehingga jika ingin mengisi nilai atau merubah nilai dapat menggunakan method setter. Method getter digunakan untuk menampilkan atau memanggil isi atribut yang bersifat private.

2. Di dalam *class Processor* dan *class Laptop*, masing-masing terdapat konstruktor default dan konstruktor berparameter. Bagaimanakah beda penggunaan dari kedua jenis konstruktor tersebut ?

Jawab : konstruktor default, instansiasinya tanpa emmberikan nilai, jika ingin memberi nilai pada objek panggil method setter dan jika menampilkannya menggunakan method getter.

Untuk konstruktor berparameter instansianya menggunakan nilai sesuai dengan tipedata pada parameter tersebut. Pada konstruktor ini bisa tidak menggunakan method setter getter untuk mengisi dan menampilkan nilai

3. Perhatikan *class Laptop*, di antara 2 atribut yang dimiliki (merk dan proc), atribut manakah yang bertipe object ?

Jawab : Atribut proc kaarena bertipe Processor

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

Jawab : Pada baris ke `13` `private Processor proc;` saat deklarasi atribut proc bertipe data Processor

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

Jawab : Untuk memanggil method info pada objek Processor dan menampilkan nilai pada objek processor di class laptop.

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

```
Laptop l = new Laptop("Thinkpad", p);
```

Apakah p tersebut ?

Jawab : p adalah instansiasi dari class Processor dan menjadi objek di class Main

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 ?

Jawab :

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

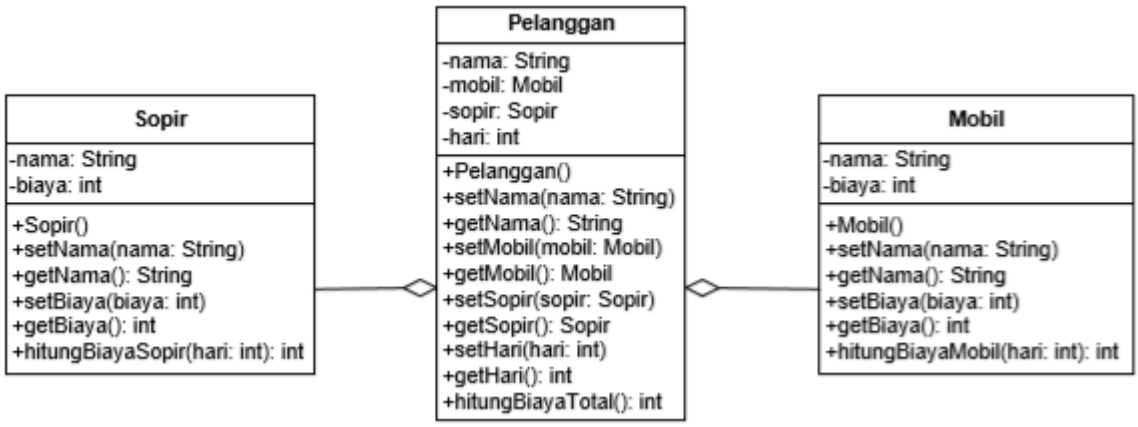
```
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 3.00
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 4.00
--------------------------------
BUILD SUCCESS
--------------------------------
```

Tidak ada perubahan

2.2 Percobaan 2

2.2.1 Langkah-Langkah Percobaan

Perhatikan diagram class berikut yang menggambarkan sistem rental mobil. Pelanggan bisa menyewa mobil sekaligus sopir. Biaya sopir dan biaya sewa mobil dihitung per hari.



a. Tambahkan package <identifier>.relasiclass.percobaan2.

b. Buatlah class Mobil di dalam package tersebut.

c. Tambahkan atribut merk tipe String dan biaya tipe int dengan akses modifier private.

```
public class Mobil {
    private String nama;
    private int biaya;
```

d. Tambahkan constructor default serta setter dan getter.

```
    Mobil(){
    }
    void setNama(String nama){
        this.nama=nama;
    }
    String getNama(){
        return nama;
    }
    void setBiaya(int biaya){
        this.biaya = biaya;
    }
    int getBiaya(){
        return biaya;
    }
```

e. Implementasikan method hitungBiayaMobil

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

f. Tambahkan class Sopir dengan atribut nama tipe String dan biaya tipe int dengan akses modifier private berikut dengan constructor default.

```
public class Sopir {
    private String nama;
    private int biaya;

    Sopir(){
    }
    void setNama(String nama){
        this.nama = nama;
    }
    String getNama(){
        return nama;
    }
    void setBiaya(int biaya){
        this.biaya = biaya;
    }
    int getBiaya(){
        return biaya;
    }
```

g. Implementasikan method hitungBiayaSopir

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

h. Tambahkan class Pelanggan dengan constructor default.

i. Tambahkan atribut-atribut dengan akses modifier private berikut:

Atribut	Tipe
nama	String
mobil	Mobil
sopir	Sopir
hari	int

```
*/
public class Pelanggan {
    private String nama;
    private Mobil mobil;
    private Sopir sopir;
    private int hari;

    Pelanggan(){

    }
```

j. Implementasikan setter dan getter.

```
void setNama(String nama){
    this.nama=nama;
}
String getNama(){
    return nama;
}
void setMobil(Mobil mobil){
    this.mobil=mobil;
}
Mobil getMobil(){
    return mobil;
}
void setSopir(Sopir sopir){
    this.sopir=sopir;
}
Sopir getSopir(){
    return sopir;
}
void setHari(int hari){
    this.hari = hari;
}
```

k. Tambahkan method hitungBiayaTotal

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

l. Buatlah class MainPercobaan2 yang berisi method main(). Tambahkan baris kode berikut:

```
public class MainPercobaan2 {
    public static void main(String[] args){
    Mobil m = new Mobil();
    m.setNama("Avanza");
    m.setBiaya(350000);
    Sopir s = new Sopir();
    s.setNama("John Doe");
    s.setBiaya(200000);
    Pelanggan p = new Pelanggan();
    p.setNama("Jane Doe");
    p.setMobil(m);
    p.setSopir(s);
    p.setHari(2);
    System.out.println("Biaya Total = "
    +  p.hitungBiayaTotal());
}
}
```

m. Compile dan jalankan class MainPercobaan2, dan perhatikan hasilnya!

```
Biaya Total = 1100000
------------------------------
BUILD SUCCESS
------------------------------
```

2.2.2 Pertanyaan

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

Jawab : Pada baris ke 13-14 ketika deklarasi atribut sopir dan mobil

```
11  public class Pelanggan {
12      private String nama;
13      private Mobil mobil;
14      private Sopir sopir;
15      private int hari;
```

2. Perhatikan method `hitungBiayaSopir` pada class `Sopir`, serta method `hitungBiayaMobil` pada class Mobil. Mengapa menurut Anda method tersebut harus memiliki argument hari ?

Jawab : karena method tersebut memiliki perhitungan antara biaya dengan hari yang nantinya harus diinputkan melalui parameter method.

3. Perhatikan kode dari class `Pelanggan`. Untuk apakah perintah `mobil.hitungBiayaMobil(hari)` dan `sopir.hitungBiayaSopir(hari)` ?

Jawab : untung memanggil method hitungBiayaMobil pada class mobil dan method hitungBiayaSopir pada class sopir dengan masing-masing argument hari.

4. Perhatikan class MainPercobaan2. Untuk apakah sintaks `p.setMobil(m)` dan `p.setSopir(s)` ?

Jawab : untuk memberikan nilai mobildan nilai sopir pada object p

5. Perhatikan class `MainPercobaan2`. Untuk apakah proses `p.hitungBiayaTotal()` tersebut ?

Jawab : untuk menghitung antara biaya mobil dan biaya sopir serta hari

6. 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?

Jawab :

```
24          System.out.println("Biaya Total = " +  p.hitungBiayaTotal());
            System.out.println(p.getMobil().getNama());
26      }
27      }
```

```
--- exec-maven-plugin:3.0.0:exec (defa
Biaya Total = 1100000
Avanza
------------------------------------------
BUILD SUCCESS
------------------------------------------
Total time:  2.376 s
Finished at: 2022-10-16T17:20:35+07:00
------------------------------------------
```

Untuk menampilkan nilai atribut nama pada objek mobil

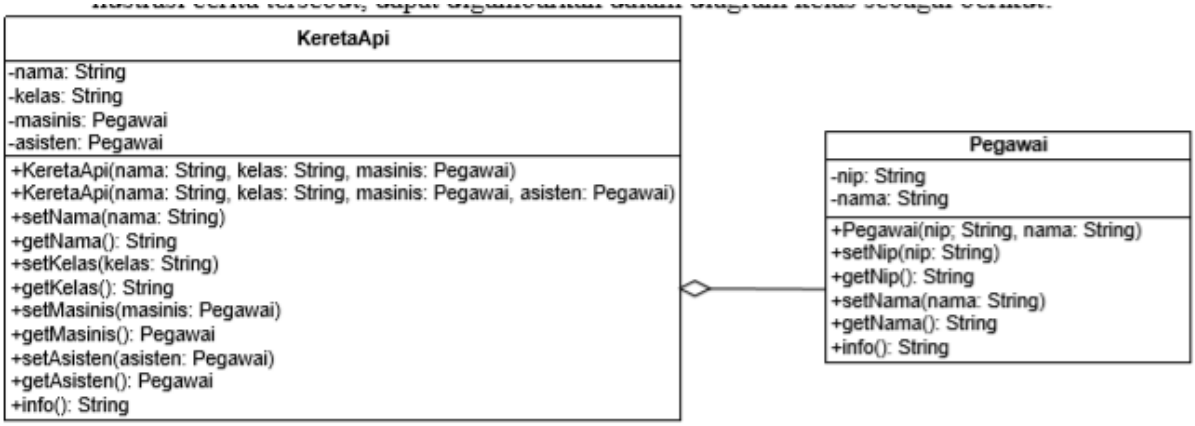
2.3 Percobaan 3

Pada percobaan-percobaan sebelumnya, relasi dalam class dinyatakan dalam one-to-one. Tetapi ada kalanya relasi class melibatkan lebih dari satu. Hal ini disebut dengan multiplicity. Untuk relasi lebih rinci mengenai multiplicity, dapat dilihat pada tabel berikut.

Multiplicity	Keterangan
0..1	0 atau 1 instance
1	Tepat 1 instance
0..*	0 atau lebih instance
1..*	setidaknya 1 instance
n	Tepat n instance (n diganti dengan sebuah angka)
m..n	Setidaknya m instance, tetapi tidak lebih dari n

2.3.1 Langkah-Langkah Percobaan

a. Sebuah Kereta Api dioperasikan oleh Masinis serta seorang Asisten Masinis. Baik Masinis maupun Asisten Masinis keduanya merupakan Pegawai PT. Kereta Api Indonesia. Dari ilustrasi cerita tersebut, dapat digambarkan dalam diagram kelas sebagai berikut:



b. Perhatikan dan pahami diagram kelas tersebut, kemudian bukalah IDE anda!

c. Buatlah package <identifier>.relasiclass.percobaan3, kemudian tambahkan class Pegawai.

d. Tambahkan atribut-atribut ke dalam class Pegawai

```
public class Pegawai {
    private String nip;
    private String nama;
```

e. Buatlah constructor untuk class Pegawai dengan parameter nip dan nama.

```
Pegawai(String nip, String nama){
    this.nip=nip;
    this.nama=nama;
}
```

f. Tambahkan setter dan getter untuk masing-masing atribut.

```
void setNip(String nip){
    this.nip=nip;
}
String getNip(){
    return nip;
}
void setNama(String nama){
    this.nama=nama;
}
String getNama(){
    return nama;
}
```

g. Implementasikan method info() dengan menuliskan baris kode berikut:

```
    String info(){
        String info = " ";
        info += "Nip : " + this.nip + "\n";
        info += "Nama : " + this.nama + "\n";
        return info;
    }
}
```

h. Buatlah class KeretaApi berdasarkan diagram class.

i. Tambahkan atribut-atribut pada class KeretaApi berupa nama, kelas, masinis, dan asisten.

```
public class KeretaApi {
    private String nama;
    private String kelas;
    private Pegawai masinis;
    private Pegawai asisten;
```

j. Tambahkan constructor 3 parameter (nama, kelas, masinis) serta 4 parameter (nama, kelas, masinis, asisten).

```
KeretaApi(String nama, String kelas, Pegawai masinis){
    this.nama = nama;
    this.kelas = kelas;
    this.masinis = masinis;
}
KeretaApi(String nama, String kelas, Pegawai masinis, Pegawai asisten){
    this.nama = nama;
    this.kelas = kelas;
    this.masinis = masinis;
    this.asisten = asisten;
}
```

k. Tambahkan setter dan getter untuk atribut-atribut yang ada pada class KeretaApi .

```
void setNama(String nama){
    this.nama = nama;
}
String getNama(){
    return nama;
}
void setKelas(String kelas){
    this.kelas = kelas;
}
String getKelas(){
    return kelas;
}
void setMasinis(Pegawai masinis){
    this.masinis = masinis;
}
Pegawai getMasinis(){
    return masinis;
}
void setAsisten(Pegawai asisten){
    this.asisten = asisten;
}
Pegawai getAsisten(){
    return asisten;
}
```

l. Kemudian implementasikan method info()

```
    String info(){
        String info = "";
        info += "Nama: " + this.nama + "\n";
        info += "Kelas: " + this.kelas + "\n";
        info += "Masinis: " + this.masinis.info() + "\n";
        info += "Asisten: " + this.asisten.info() + "\n";
        return info;
    }
}
```

m. Buatlah sebuah class MainPercobaan3 dalam package yang sama.

n. Tambahkan method main() kemudian tuliskan baris kode berikut.

```
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());
    }
}
```

o. Complie

```
Nama: Gaya Baru
Kelas: Bisnis
Masinis:  Nip : 1234
Nama : Spongebob Squarepants

Asisten:  Nip : 4567
Nama : Patrick Star


--------------------------------
BUILD SUCCESS
--------------------------------
```

2.3.2 Pertanyaan

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

Jawab : `this.masinis.info()` untuk menampilkan nilai pada objek masinis dan `this.asisten.info()` untuk menampilkan nilai pada objek asisten

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());
```

Jawab :

```
11      public class MainPertanyaan {
12          Pegawai masinis = new Pegawai("1234", "Spongebob Squarepants");
13          KeretaApi keretaApi = new KeretaApi("Gaya Baru", "Bisnis", masinis);
            System.out.println(keretaApi.info());
15      }
16
```

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

Jawab :

```
------------------< com.mycompany.prak_pbo:Prak_PBO >-------------------
Building Prak_PBO 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ Prak_PBO ---
Error: Could not find or load main class com.mycompany.prak_pbo.Prak_PBO
Caused by: java.lang.ClassNotFoundException: com.mycompany.prak_pbo.Prak_PBO
Command execution failed.
org.apache.commons.exec.ExecuteException: Process exited with an error: 1 (Exit value: 1)
    at org.apache.commons.exec.DefaultExecutor.executeInternal (DefaultExecutor.java:404)
    at org.apache.commons.exec.DefaultExecutor.execute (DefaultExecutor.java:166)
    at org.codehaus.mojo.exec.ExecMojo.executeCommandLine (ExecMojo.java:982)
    at org.codehaus.mojo.exec.ExecMojo.executeCommandLine (ExecMojo.java:929)
```

Karena pada KeretaApi.info() berisi menampilkan nama, kelas, masinis(nip, nama) dan asisten (nip, nama). Sedangkan di class MainPertanyaan hanya menginstansiasikan pegawai masinis dan kereta saja sehingga pegawai asisten null

4. Perbaiki class KeretaApi sehingga program dapat berjalan !

Jawab :

```
String info2(){
    String info2= "";
    info2 += "Nama: " + this.nama + "\n";
    info2 += "Kelas: " + this.kelas + "\n";
    info2 += "Masinis: " + this.masinis.info() + "\n";
    return info2;
}
```

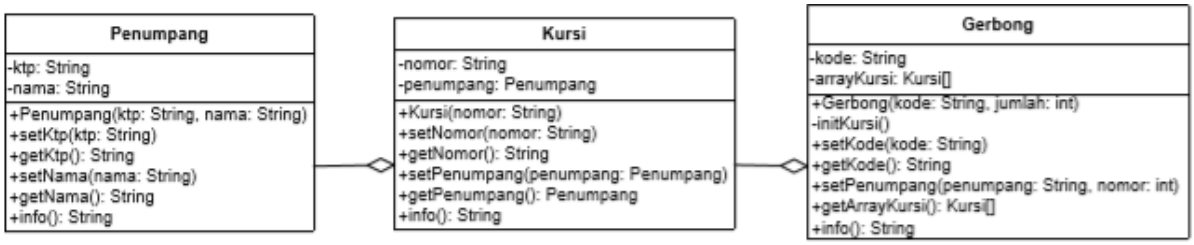
Tambahkan method infoBaru di class KeretaApi untuk menampilkan pegawai masinis dan kereta saja

```
--- exec-maven-plugin:3.0.0:exec (defau
Nama: Gaya Baru
Kelas: Bisnis
Masinis:  Nip : 1234
Nama : Spongebob Squarepants


------------------------------------------
BUILD SUCCESS
```

2.4 Percobaan 4

2.4.1 Langkah-Langkah Percobaan



a. Perhatikan dan pahami diagram class tersebut.

b. Buatlah masing-masing class Penumpang, Kursi dan Gerbong sesuai rancangan tersebut pada package <identifier>.relasiclass.percobaan4.

Class Penumpang

```
Penumpang(String ktp, String nama){
    this.ktp = ktp;
    this.nama = nama;
}
void setKtp(String ktp){
    this.ktp = ktp;
}
String getKtp(){
    return ktp;
}
void setNama(String nama){
    this.nama = nama;
}
String getNama(){
    return nama;
}
```

Class Kursi

```
public class Kursi {
    private String nomor;
    private Penumpang penumpang;

    Kursi(String nomor){
        this.nomor = nomor;
    }
    void setNomor(String nomor){
        this.nomor = nomor;
    }
    String getNomor(){
        return nomor;
    }
    void setPenumpang(Penumpang penumpang){
        this.penumpang = penumpang;
    }
    Penumpang getPenumpang(){
        return penumpang;
    }
```

Class Gerbong

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

    Gerbong(String kode, int jumlah){
        this.kode = kode;
        this.arrayKursi = new Kursi[jumlah];
        this.initKursi();
    }
    private void initKursi(){
        for(int i = 0; i < arrayKursi.length; i++){
        this.arrayKursi[i] = new Kursi(String.valueOf(i+1));
        }
    }
    void setKode(String kode){
        this.kode = kode;
    }
    String getKode(){
        return kode;
    }
    void setPenumpang(Penumpang penumpang, int nomor){
        this.arrayKursi[nomor-1].setPenumpang(penumpang);
    }
    Kursi[] getArrayKursi(){
        return arrayKursi;
    }
```

c. Tambahkan method info() pada class Penumpang

```
String info(){
    String info = "";
    info += "Ktp: " + ktp + "\n";
    info += "Nama: " + nama + "\n";
    return info;
}
```

d. Tambahkan method info() pada class Kursi

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

e. Pada class Gerbong buatlah method initKursi() dengan akses private.

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

f. Panggil method initKursi() dalam constructor Gerbong sehingga baris kode menjadi berikut:

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

g. Tambahkan method info() pada class Gerbong

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

h. Implementasikan method untuk memasukkan penumpang sesuai dengan nomor kursi.

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

i. Buatlah class MainPercobaan4 yang berisi method main(). Kemudian tambahkan baris berikut!

```
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());
    }
}
```

j. Compile

```
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 SUCCESS
------------------------
```

2.4.2 Pertanyaan

1. Pada main program dalam class `MainPercobaan4`, berapakah jumlah kursi dalam Gerbong A ?

Jawab : Jumlah kursi pada gerbong A ada 10 kursi

2. Perhatikan potongan kode pada method `info()` dalam class `Kursi`. Apa maksud kode tersebut?

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

Jawab : Untuk memastikan penumpang tidak sama dengan null/kosong. Dan jika ada penumpang maka akan ditampilkan info tentang penumpang itu.

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

Jawab : Untuk menyesuaikan array yang dibuat yaitu dimulai dari indeks ke 0

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

Jawab :

```
11  public class MainPercobaan4 {
12      public static void main(String[] args){
13          Penumpang p = new Penumpang("12345", "Mr. Krab");
14          Gerbong gerbong = new Gerbong("A", 10);
15          gerbong.setPenumpang(p, 1);
            Penumpang budi = new Penumpang("56789", "Budi");
17          gerbong.setPenumpang(budi, 1);
18          System.out.println(gerbong.info());
19      }
```

```
--- exec-maven-plugin:3.
Kode: A
Nomor: 1
Penumpang: Ktp: 56789
Nama: Budi

Nomor: 2
Nomor: 3
Nomor: 4
Nomor: 5
Nomor: 6
Nomor: 7
Nomor: 8
Nomor: 9
Nomor: 10

------------------------
BUILD SUCCESS
------------------------
```

Penumpang sebelumnya yaitu “Mr. Krab” tergantikan oleh penumpang baru yaitu “Budi”

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

Jawab :

Ubah method setPenumpang pada class gerbong

```
30      }
        void setPenumpang(Penumpang penumpang, int nomor){
32          if(this.arrayKursi[nomor-1].getPenumpang()!=null) {
33              System.out.println("Maaf, Kursi telah dipesan!");
34          } else{
35              this.arrayKursi[nomor-1].setPenumpang(penumpang);
36          }
37      }
```

```
--- exec-maven-plugin:3.0.0:e
Maaf, Kursi telah dipesan!
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 SUCCESS
------------------------------
```

Kursi nomo 1 tidak akan tergantikan oleh penumpang abru yaitu budi dan tetap menginfokan pesan bahwa kursi 1 telah dipesan beserta data penumpang di kursi 1.

```
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);
        Penumpang budi = new Penumpang("56789", "Budi");
        gerbong.setPenumpang(budi, 2);
        System.out.println(gerbong.info());
    }
}
```

```
--- exec-maven-plugin:
Kode: A
Nomor: 1
Penumpang: Ktp: 12345
Nama: Mr. Krab

Nomor: 2
Penumpang: Ktp: 56789
Nama: Budi

Nomor: 3
Nomor: 4
Nomor: 5
Nomor: 6
Nomor: 7
Nomor: 8
Nomor: 9
Nomor: 10

------------------------
BUILD SUCCESS
```

Kecuali kita ganti Budi duduk di kursi nomor 2 maka akan muncul data budi di kursi nomor 2 beserta data penumpang nomor 1.

3. 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).

Jawab :