

Object Oriented Programming Relation

**Name**

Virza Aulia Rachman

NIM

2241720078

Class

1i

Department

Information Technology

Study Program

D4 Informatics Engineering

Practicum 1

```
package OOP.relasiclass.percobaan1;

7 usages
public class Processor {
    3 usages
    private String merk;
    3 usages
    private double cache;
    1 usage
    Processor();{};
    1 usage
    Processor(String merk,double cache){
        this.cache=cache;
        this.merk=merk;
    }
    1 usage
    void setMerk(String merk){
        this.merk = merk;
    }
    1 usage
    void setCache(double cache){
        this.cache = cache;
    }

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

```
package OOP.relasiclass.percobaan1;

4 usages
public class Laptop {
    3 usages
    private String merk;
    3 usages
    private Processor proc;

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

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

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

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

```
package OOP.relasiclass.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",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();
    }
}
```

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

Question

1. Di dalam `class Processor` dan `class Laptop`, terdapat method *setter* dan *getter* untuk masing-masing atributnya. Apakah gunanya *method setter* dan *getter* tersebut ?
2. Di dalam `class Processor` dan `class Laptop`, masing-masing terdapat konstruktor default dan konstruktor berparameter. Bagaimanakah beda penggunaan dari kedua jenis konstruktor tersebut ?
3. Perhatikan `class Laptop`, di antara 2 atribut yang dimiliki (*merk* dan *proc*), atribut manakah yang bertipe *object* ?
4. Perhatikan `class Laptop`, pada baris manakah yang menunjukkan bahwa `class Laptop` memiliki relasi dengan `class Processor` ?
5. Perhatikan pada `class Laptop`, Apakah guna dari sintaks `proc.info()` ?
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 ?

Answer

1. Setter digunakan untuk memasukkan nilai suatu atribut sedangkan *getter* untuk mengambil nilai atribut
2. Konstruktor default dapat menginstasiasikan namun objek harus menggunakan setter untuk mendapatkan nilai atribut sedangkan konstruktor berparameter bisa langsung memberikan nilai pada atribut objek
3. `Proc`
4. Baris ke 5
5. Untuk mengeluarkan informasi dari `processor`.
6. `p` adalah nama dari objek `processor` yang digunakan sebagai referensi atribut dari `laptop l`, Program tetap bisa dijalankan.

Practicum

```
package OOP.relasticlass.percobaan2;

4 usages
public class Mobil {
    1 usage
    private String merk;
    2 usages
    private int biaya;

    1 usage
    Mobil(){};
    1 usage
    public void setMerk(String merk) {
        this.merk = merk;
    }
    1 usage
    public void setBiaya(int biaya) {
        this.biaya = biaya;
    }
    1 usage
    public int hitungBiayaMobil(int hari){
        return biaya * hari;
    }
}
```

```
package OOP.relasticlass.percobaan2;

4 usages
public class Sopir {
    1 usage
    private String nama;
    2 usages
    private int biaya;

    1 usage
    public void setNama(String nama) {
        this.nama = nama;
    }
    1 usage
    public void setBiaya(int biaya) {
        this.biaya = biaya;
    }
    1 usage
    public int hitungBiayaSopir(int hari){
        return biaya * hari;
    }
}
```

```
package OOP.relasticlass.percobaan2;

2 usages
public class pelanggan {
    1 usage
    String nama;
    2 usages
    Mobil mobil;
    2 usages
    Sopir sopir;
    3 usages
    int hari;
    1 usage
    public void setHari(int hari) {
        this.hari = hari;
    }
    1 usage
    public void setMobil(Mobil mobil) {
        this.mobil = mobil;
    }
    1 usage
    public void setNama(String nama) {
        this.nama = nama;
    }
    1 usage
    public void setSopir(Sopir sopir) {
        this.sopir = sopir;
    }
    1 usage
    public int hitungBiayaTotal(){
        return mobil.hitungBiayaMobil(hari) + sopir.hitungBiayaSopir(hari);
    }
}
```

```
package OOP.relasticlass.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("teo aa");
        s.setBiaya(200000);
        pelanggan p = new pelanggan();
        p.setNama("Ananda");
        p.setMobil(m);
        p.setSopir(s);
        p.setHari(2);
        System.out.println("Biaya Total = " + p.hitungBiayaTotal());
    }
}
```

Biaya Total = 1100000

Process finished with exit code 0

Question :

1. Perhatikan *class* Pelanggan. Pada baris program manakah yang menunjukkan bahwa *class* Pelanggan memiliki relasi dengan *class* Mobil dan *class* Sopir ?
2. Perhatikan *method* *hitungBiayaSopir* pada *class* Sopir, serta *method* *hitungBiayaMobil* pada *class* Mobil. Mengapa menurut Anda *method* tersebut harus memiliki argument hari ?
3. Perhatikan kode dari *class* Pelanggan. Untuk apakah perintah *mobil.hitungBiayaMobil(hari)* dan *sopir.hitungBiayaSopir(hari)* ?
4. Perhatikan *class* MainPercobaan2. Untuk apakah sintaks *p.setMobil(m)* dan *p.setSopir(s)* ?
5. Perhatikan *class* MainPercobaan2. Untuk apakah proses *p.hitungBiayaTotal()* tersebut ?
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?

Answer :

1. Baris ke 5 dan 6
2. Karena *method* tersebut memerlukan parameter tersebut untuk melakukan perhitungan.
3. Untuk menghitung biaya total pelanggan.
4. Untuk memberikan value ke atribut pelanggan p.
5. Untuk menghitung biaya total objek p.
6. Untuk mendapatkan value dari objek p, sesuai dengan atribut yang dipilih.

Practicum 3

```
package OOP.relasticlass.percobaan3;

11 usages
public class Pegawai {
    3 usages
    private String nip;
    3 usages
    private String nama;
    2 usages
    Pegawai(String nip,String nama){
        this.nip = nip;
        this.nama = nama;
    }

    no usages
    public void setName(String nama) {
        this.nama = nama;
    }

    no usages
    public void setNip(String nip) {
        this.nip = nip;
    }

    no usages
    public String info(){
        String info = ""; info += "Nip: " +this.nip+ "\n";
        info += "Nama: " + this.nama + "\n";
        return info;
    }
}
```

```
Nama: Gaya baru
Kelas: Bisnis
Masinis: OOP.relasticlass.percobaan3.Pegawai@312b1dae
asisten: OOP.relasticlass.percobaan3.Pegawai@7530d0a
```

```
package OOP.relasticlass.percobaan3;

2 usages
public class KeretaApi {
    4 usages
    private String nama;
    4 usages
    private String kelas;
    4 usages
    private Pegawai masinis;
    3 usages
    private Pegawai asisten;

    no usages
    KeretaApi(String nama,String kelas,Pegawai masinis){
        this.nama = nama;
        this.kelas = kelas;
        this.masinis = masinis;
    }

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

    no usages
    public void setName(String nama) {
        this.nama = nama;
    }

    no usages
    public void setAsisten(Pegawai asisten) {
        this.asisten = asisten;
    }

    no usages
    public void setKelas(String kelas) {
        this.kelas = kelas;
    }

    no usages
    public void setMasinis(Pegawai masinis) {
        this.masinis = masinis;
    }

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

```
package OOP.relasticlass.percobaan3;

public class MainPercobaan3 {
    public static void main(String[]largs){
        Pegawai masinis = new Pegawai( nip: "1234", nama: "Teo ananda");
        Pegawai asisten = new Pegawai( nip: "4567", nama: "Benjamin au");
        KeretaApi keretaApi = new KeretaApi( nama: "Gaya baru", kelas: "Bisnis",masinis,asisten);

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

Question:

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

3. Apa hasil output dari *main* program tersebut ? Mengapa hal tersebut dapat terjadi ?
4. Perbaiki *class* `KeretaApi` sehingga program dapat berjalan !

Answer :

1. Untuk mengambil informasi dari atribut asisten dan masinis yang dimana kedaunya termasuk objek
2. This is the code:

```
package OOP.relasiclass.percobaan3;  
  
public class MainPertanyaan {  
    public static void main (String[] args){  
        Pegawai masinis = new Pegawai( nip: "1234", nama: "Teo ananda");  
        KeretaApi keretaApi = new KeretaApi( nama: "Gaya Baru", kelas: "Bisnis", masinis);  
        System.out.println(keretaApi.info());  
    }  
}
```

3. Karena atribut asisten tidak dimasukkan dan *method* `info` pada program pegawai kurang tepat.

```
Nama: Gaya Baru  
Kelas: Bisnis  
Masinis: OOP.relasiclass.percobaan3.Pegawai@312b1dae  
asisten: null
```

4. This is the code :

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

Practicum 4

```
package OOP.relasticlass.percobaan4;

6 usages
public class Penumpang {
    4 usages
    private String ktp;
    4 usages
    private String nama;
    1 usage
    Penumpang(String ktp,String nama){
        this.ktp = ktp;
        this.nama= nama;
    }
    no usages
    public String getNama() {
        return nama;
    }
    no usages
    public String getKtp() {
        return ktp;
    }
    no usages
    public void setNama(String nama) {
        this.nama = nama;
    }
    no usages
    public void setKtp(String ktp) {
        this.ktp = ktp;
    }
    public String info(){
        String info = "";
        info += "Ktp: " + ktp + "\n";
        info += "Nama: " + nama + "\n";
        return info;
    }
}
```

```
package OOP.relasticlass.percobaan4;

5 usages
public class Kursi {
    4 usages
    String nomor;
    4 usages
    Penumpang penumpang;

    1 usage
    Kursi(String nomor){
        this.nomor = nomor;
    }

    no usages
    public void setNomor(String nomor) {
        this.nomor = nomor;
    }

    no usages
    public String getNomor() {
        return nomor;
    }

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

    no usages
    public Penumpang getPenumpang() {
        return penumpang;
    }
    public String info(){
        String info = "";
        info += "Nomor = " + nomor + "\n";
        if(this.penumpang != null){
            info += "penumpang: " + penumpang.info()+"\n";
        }
        return info;
    }
}
```

```
package OOP.relasticlass.percobaan4;

2 usages
public class Gerbong {
    4 usages
    String kode;
    6 usages
    Kursi[] arrayKursi;

    1 usage
    Gerbong(String kode,int jumlah){
        this.kode = kode;
        this.arrayKursi = new Kursi[jumlah];
        this.initkursi();
    }

    no usages
    public void setKode(String kode) {
        this.kode = kode;
    }

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

    no usages
    public String getKode() {
        return kode;
    }

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

    no usages
    public Kursi[] getArrayKursi() {
        return arrayKursi;
    }

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

```
package OOP.relasticlass.percobaan4;

public class mainPercobaan4 {
    public static void main(String[]args){
        Penumpang p = new Penumpang( ktp: "12345", nama: "MR. Krab");
        Gerbong gerbong = new Gerbong ( kode: "A", jumlah: 10);
        gerbong.setPenumpang(p, nomor: 1);
        System.out.println(gerbong.info());
    }
}
```

```
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
```


Question :

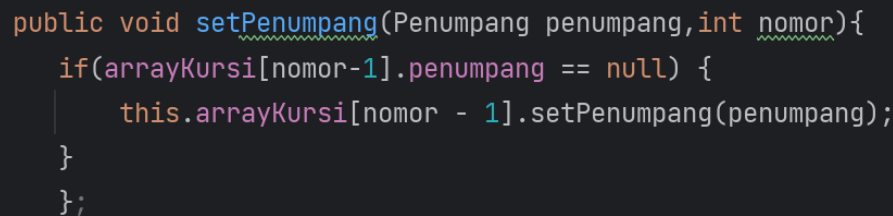
1. Pada *main* program dalam *class* MainPercobaan4, berapakah jumlah kursi dalam Gerbong A ?
2. Perhatikan potongan kode pada *method* *info()* dalam *class* Kursi. Apa maksud kode tersebut ?

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

3. Mengapa pada *method* *setPenumpang()* dalam *class* Gerbong, nilai nomor dikurangi dengan angka 1 ?
4. Instansiasi objek baru budi dengan tipe Penumpang, kemudian masukkan objek baru tersebut pada gerbong dengan *gerbong.setPenumpang(budi, 1)*. Apakah yang terjadi ?
5. Modifikasi program sehingga tidak diperkenankan untuk menduduki kursi yang sudah ada penumpang lain !

Answer:

1. 10
2. Method tersebut hanya akan mengeksekusi bagian penumpang hanya jika value nya tidak null
3. Karena dalam array indeks pertama terltetak diangka 0 bukan 1.
4. Penumpang di kursi 1 menjadi budi
5. This is the code:



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

Assignment

```
package Assignment;

2 usages
public class Person {
    2 usages
    private String name;

    1 usage
    public Person(String name) {
        this.name = name;
    }

    1 usage
    public String getName() {
        return name;
    }
}
```

```
package Assignment;

2 usages
public class Address {
    2 usages
    private String street;

    1 usage
    public Address(String street) {
        this.street = street;
    }

    1 usage
    public String getStreet() {
        return street;
    }
}
```

```
package Assignment;

2 usages
public class BankAccount {
    2 usages
    private String accountNumber;

    1 usage
    public BankAccount(String accountNumber) {
        this.accountNumber = accountNumber;
    }

    1 usage
    public String getAccountNumber() {
        return accountNumber;
    }
}
```

```
package Assignment;

2 usages
public class Transaction {
    2 usages
    private double amount;

    1 usage
    public Transaction(double amount) {
        this.amount = amount;
    }

    1 usage
    public double getAmount() {
        return amount;
    }
}
```

```
package Assignment;

public class Main {
    public static void main(String[] args) {
        // Contoh penggunaan objek dalam program
        Person person = new Person( name: "John Doe");
        Address address = new Address( street: "123 Main St");
        BankAccount account = new BankAccount( accountNumber: "123456789");
        Transaction transaction = new Transaction( amount: 100.0);

        // Mengakses atribut unik objek
        System.out.println("Name: " + person.getName());
        System.out.println("Street: " + address.getStreet());
        System.out.println("Account Number: " + account.getAccountNumber());
        System.out.println("Amount: " + transaction.getAmount());
    }
}
```