

Nama: Rara Deninda Hurianto

NIM: 2041720110

JOBSHEET 4

PERCOBAAN 1

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:

Setter berguna untuk mengeset isi method pada class Processor dan Laptop. Sedangkan Getter untuk menampilkan isi method yang sudah di set pada class Processor dan class Laptop

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 digunakan apabila user tidak ingin langsung mengisi nilai atribut pada saat instansiasi objek atau menggunakan method setter untuk mengisi nilai atribut, sedangkan konstruktor berparameter digunakan pada saat user ingin mengisi nilai saat instansiasi objek.

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

Jawab:

Atribut *proc*

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

Jawab:

`private Processor proc;`

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

Jawab:

Untuk memanggil method `info` yang terdapat pada class Processor

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:

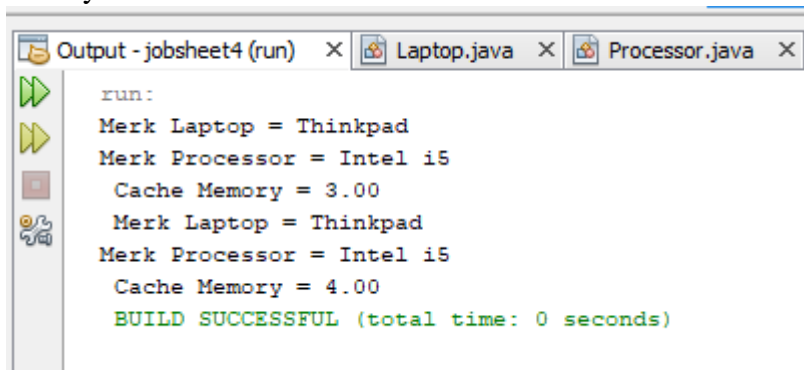
```
Laptop l = new Laptop("Thinkpad", new Processor("Intel i5", 3));
```

Bagaimanakah hasil program saat dijalankan, apakah ada perubahan ?

Jawab:

p pada syntax baris tersebut adalah argument bertipe data objek Processor, yang memuat data yang telah dimasukkan kedalam class Processor, dan diberi nama p

hasilnya:



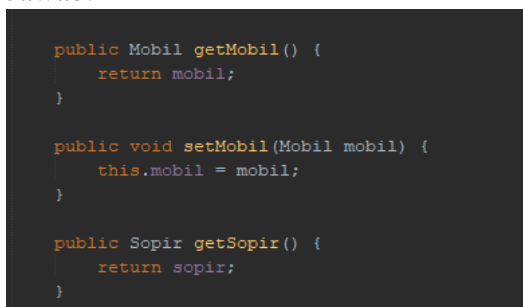
```
run:
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 3.00
Merk Laptop = Thinkpad
Merk Processor = Intel i5
Cache Memory = 4.00
BUILD SUCCESSFUL (total time: 0 seconds)
```

Hasilnya sama, karena instansiasinya berisi argument yang sama

PERCOBAAN 2

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

Jawab:



```
public Mobil getMobil() {
    return mobil;
}

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

public Sopir getSopir() {
    return sopir;
}
```

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

Jawab:

Karena biaya menyewa mobil dan sopir dihitung dari jumlah hari kita menyewa, sehingga membutuhkan parameter untuk memasukkan jumlah hari penyewaan

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

Jawab:

Untuk menghitung total biaya penyewaan pada pelanggan yang diambil dari *method* hitungBiayaSopir pada *class* sopir dan *method* hitungBiayaSewaMobil pada *class* Mobil yang sebelumnya telah diset.

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

Jawab:

Untuk mengeset nilai dengan argument objek yg nilainya biaya dan nama

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

Jawab:

Untuk menghitung biaya total yang terdapat pada class `Pelanggan` dan yang telah diset melalui class `Mobil` pada method `hitungBiayaSewaMobil` serta pada class `Sopir` pada method `hitungBiayaSopir`

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:

Untuk menampilkan nama merk dari mobilnya.

PERCOBAAN 3

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

Jawab:

Baris tersebut digunakan untuk memanggil method `info()` yang terdapat pada class `masinis` dan `asisten`

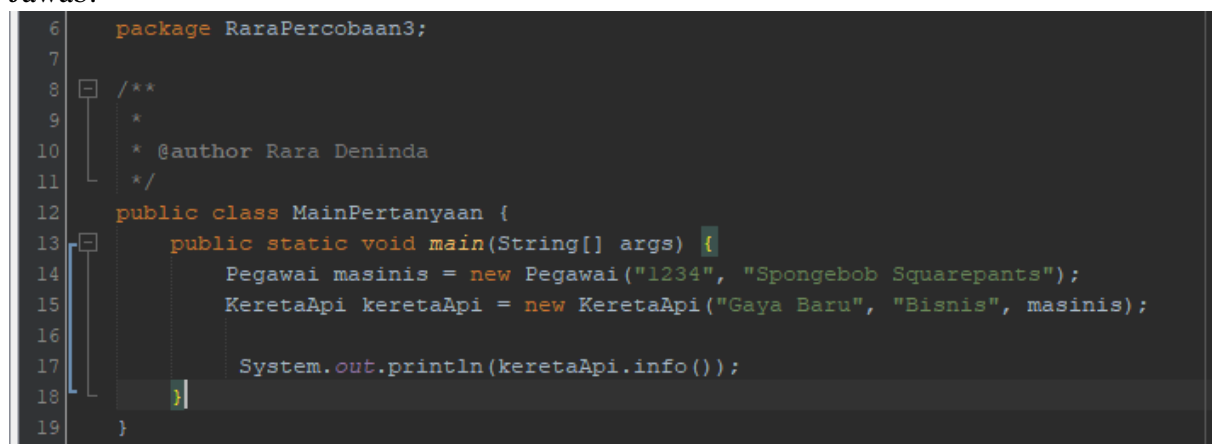
2. Buatlah *main* program baru dengan nama class `MainPertanyaan` pada *package* yang sama. Tambahkan kode berikut pada *method* `main()` !

```
Pegawai masinis = new Pegawai("1234", "Spongebob Squarepants");
```

```
KeretaApi keretaApi = new KeretaApi("Gaya Baru", "Bisnis", masinis);
```

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

Jawab:



```
6 package RaraPercobaan3;
7
8 /**
9  *
10  * @author Rara Deninda
11  */
12 public class MainPertanyaan {
13     public static void main(String[] args) {
14         Pegawai masinis = new Pegawai("1234", "Spongebob Squarepants");
15         KeretaApi keretaApi = new KeretaApi("Gaya Baru", "Bisnis", masinis);
16
17         System.out.println(keretaApi.info());
18     }
19 }
```

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

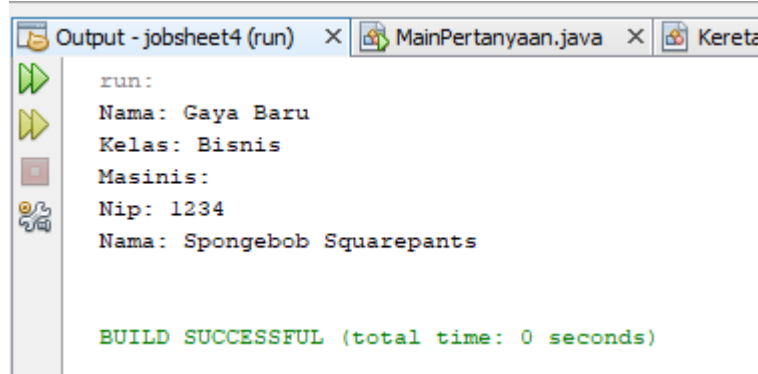
Jawab:

Hasilnya terjadi error. Karena pada method info() pada class keretaApi mengharuskan outputnya menampilkan data asisten juga. Sedangkan data asisten belum diinputkan pada class KeretaApi, sehingga program memunculkan bahwa ada atribut yang kurang dan tidak boleh bernilai null atau kosong.

4. Perbaiki class KeretaApi sehingga program dapat berjalan !

Jawab:

```
63 public String info() {
64     String info = "";
65     info += "Nama: " + this.nama + "\n";
66     info += "Kelas: " + this.kelas + "\n";
67     info += "Masinis: \n" + this.masinis.info() + "\n";
68     //info += "Asisten: \n" + this.asisten.info() + "\n";
69     if(asisten != null){
70         info += "Asisten: \n" + this.asisten.info() + "\n";
71     }
72     return info;
73 }
74 }
```



PERCOBAAN 4

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

Jawab:

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:

Baris kode tersebut maksudnya adalah memeriksa nilai didalam atribut penumpang yang ada pada class Kursi, jika atribut penumpang tidak kosong maka tambahan info tersebut dengan info yang didapat dari pemanggilan info dalam class Penumpang

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

Jawab:

Agar array menerima index yang benar karena array dimulai dari index 0, bukannya 1. Jadi Ketika kita ingin melakukan inputan pada index yang benar namun inputannya kita gunakan range mulai dari 1, maka nilai harus dikurangi 1

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

Jawab:

```
12 public class MainPercobaan4 {
13     public static void main(String[] args) {
14         Penumpang p = new Penumpang("12345", "Mr. Krab");
15         Gerbong gerbong = new Gerbong("A", 10);
16         gerbong.setPenumpang(p, 1);
17         System.out.println(gerbong.info());
18
19         Penumpang Budi = new Penumpang("1941720147", "Budi");
20         gerbong.setPenumpang(Budi, 1);
21         System.out.println(gerbong.info());
22     }
23 }
```

Output - jobsheet4 (run) X MainPertanyaan.java X Keri

```
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

Kode: A
Nomor : 1
Penumpang : Ktp: 1941720147
Nama: Budi

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

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

yang terjadi adalah data yang terdapat pada kursi nomor 1 tertimpa oleh data budi.

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

Jawab:

```

37 public void setPenumpang(Penumpang penumpang, int nomor){
38     int index = nomor = 1;
39     if(arrayKursi[index].getPenumpang() != null){
40         System.out.println("Kursi telah terisi, silahkan pilih kursi lain!");
41     } else {
42         this.arrayKursi[index].setPenumpang(penumpang);
43     }
44 }
45 }
46

```

Output - jobsheet4 (run) x MainPertanyaan.java x Kereta

```

run:
Kode: A
Nomor : 1
Nomor : 2
Penumpang : Ktp: 12345
Nama: Mr. Krab

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

Kursi telah terisi, silahkan pilih kursi lain!
Kode: A
Nomor : 1
Nomor : 2
Penumpang : Ktp: 12345
Nama: Mr. Krab

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

BUILD SUCCESSFUL (total time: 0 seconds)

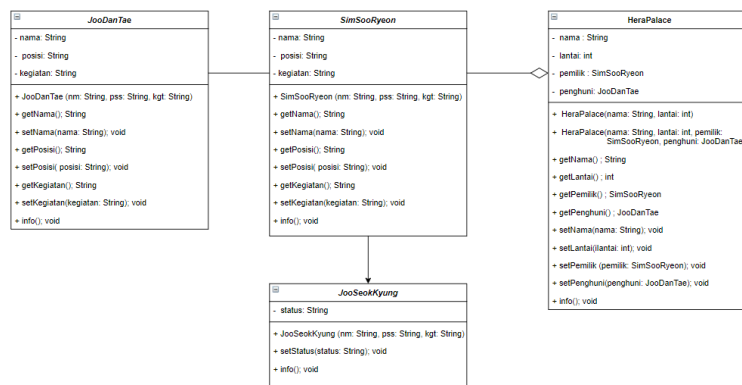
```

TUGAS

1. 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:

- **Class diagram**



- Class HeraPalace



```
10  /**
11  */
12  public class HeraPalace {
13      private String nama;
14      private int lantai;
15      private SimSooRyeon pemilik;
16      private JooDanTae penghuni;
17
18      HeraPalace(String nama, int lantai) {
19          this.nama = nama;
20          this.lantai = lantai;
21      }
22
23      public HeraPalace(String nama, int lantai, SimSooRyeon pemilik, JooDanTae penghuni) {
24          this.nama = nama;
25          this.lantai = lantai;
26          this.pemilik = pemilik;
27          this.penghuni = penghuni;
28      }
29
30      public String getNama() {
31          return nama;
32      }
33
34      public void setNama(String nama) {
35          this.nama = nama;
36      }
37
38      public int getLantai() {
39          return lantai;
40      }
41
42      public void setLantai(int lantai) {
43          this.lantai = lantai;
44      }
45
46      public SimSooRyeon getPemilik() {
47          return pemilik;
48      }
49
50      public void setPemilik(SimSooRyeon pemilik) {
51          this.pemilik = pemilik;
52      }
53
54      public JooDanTae getPenghuni() {
55          return penghuni;
56      }
57
58      public void setPenghuni(JooDanTae penghuni) {
59          this.penghuni = penghuni;
60      }
61
62      public void info() {
63          System.out.println("Lokasi : "+this.nama);
64          System.out.println("Lantai : "+ this.lantai);
65          System.out.println("\nPemilik : ");
66          this.pemilik.info();
67          System.out.println("\nPenghuni : ");
68          this.penghuni.info();
69          //return info;
70      }
71  }
```

- Class sim soo ryeon

```
Output - jobsheet4 (run) x HeraPalace.java x SimSooRyeon.java x JooDanTae.java x JooSeokKy...
Source History
11  */
12  public class SimSooRyeon {
13      private String nama;
14      private String posisi;
15      private String kegiatan;
16
17      public SimSooRyeon(String nama, String posisi, String kegiatan) {
18          this.nama = nama;
19          this.posisi = posisi;
20          this.kegiatan = kegiatan;
21      }
22
23      public String getName() {
24          return nama;
25      }
26
27      public void setName(String nama) {
28          this.nama = nama;
29      }
30
31      public String getPosisi() {
32          return posisi;
33      }
34
35      public void setPosisi(String posisi) {
36          this.posisi = posisi;
37      }
38
39      public String getKegiatan() {
40          return kegiatan;
41      }
42
43      public void setKegiatan(String kegiatan) {
44          this.kegiatan = kegiatan;
45      }
46
47      public void info() {
48          System.out.println("Nama      : "+this.nama);
49          System.out.println("Pekerjaan : "+ this.posisi);
50          System.out.println("Kegiatan  : "+ this.kegiatan);
51          // return info;
52      }
53
54  }
```

- Class Joo Dan Tae

```
Output - jobsheet4 (run) x HeraPalace.java x SimSooRyeon.java x JooDanTae.java x JooSeokKy...
Source History
10  * @author Rara Deninda
11  */
12  public class JooDanTae {
13      private String nama;
14      private String posisi;
15      private String kegiatan;
16
17      public JooDanTae(String nama, String posisi, String kegiatan) {
18          this.nama = nama;
19          this.posisi = posisi;
20          this.kegiatan = kegiatan;
21      }
22
23      public String getName() {
24          return nama;
25      }
26
27      public void setName(String nama) {
28          this.nama = nama;
29      }
30
31      public String getPosisi() {
32          return posisi;
33      }
34
35      public void setPosisi(String posisi) {
36          this.posisi = posisi;
37      }
38  }
```



```

38
39     public String getKegiatan() {
40         return kegiatan;
41     }
42
43     public void setKegiatan(String kegiatan) {
44         this.kegiatan = kegiatan;
45     }
46
47     public void info() {
48         System.out.println("Nama      : "+this.nama);
49         System.out.println("Pekerjaan : "+ this.posisi);
50         System.out.println("Kegiatan  : "+ this.kegiatan);
51         System.out.println("\n");
52     }
53 }

```

- Class Joo Seok Kyung

```

10     * @author Rara Deninda
11     */
12     public class JooSeokKyung extends SimSooRyeon {
13         private String status;
14
15         public JooSeokKyung(String nama, String posisi, String kegiatan) {
16             super(nama, posisi, kegiatan);
17         }
18
19         public void setStatus(String status) {
20             this.status = status;
21         }
22
23         public void info() {
24             System.out.println("Inheritance");
25             super.info();
26             System.out.println("Status      : "+ status);
27         }
28     }
29 }

```

- Class main

```

10     * @author Rara Deninda
11     */
12     public class Main {
13         public static void main(String[] args){
14
15             SimSooRyeon pemilik = new SimSooRyeon("Sim Soo Ryeon", "Pemilik Hera Palace dan seorang ibu", "Mengurus pekerjaan rumah");
16             JooDanTae penghuni = new JooDanTae ("Joo Dan Tae", "Pembisnis Real Estate", "Mengurus bisnis");
17             JooSeokKyung anak = new JooSeokKyung("Joo Seok Kyung", "Anak Sim Soo Ryeon & Joo Dan Tae", "Sekolah");
18             HeraPalace hera = new HeraPalace("Hera Palace", 100, pemilik, penghuni);
19
20             anak.setNama("Joo Seok Kyung");
21             anak.setPosisi("Anak Sim Soo Ryeon & Joo Dan Tae");
22             anak.setKegiatan("Sekolah, belajar, bermain");
23             anak.setStatus("Pelajar");
24
25             hera.info();
26             anak.info();
27         }
28     }

```

Output

```
Output - jobsheet4 (run) x HeraPalace.java x SimSooRyeon.java x JooDanTae.java x JooSeokKyung.java x
run:
Lokasi : Hera Palace
Lantai : 100

Pemilik
Nama      : Sim Soo Ryeon
Pekerjaan : Pemilik Hera Palace dan seorang ibu
Kegiatan  : Mengurus pekerjaan rumah

Penghuni
Nama      : Joo Dan Tae
Pekerjaan : Pembisnis Real Estate
Kegiatan  : Mengurus bisnis

Inheritance
Nama      : Joo Seok Kyung
Pekerjaan : Anak Sim Soo Ryeon & Joo Dan Tae
Kegiatan  : Sekolah, belajar, bermain
Status    : Pelajar
BUILD SUCCESSFUL (total time: 0 seconds)
```