2. 1 Percobaan 1

Pertanyaan :

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 : pada method set digunakan untuk mengisi data di atribut dan untuk method get digunakan untuk memanggil data atribut yang sudah diset
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 : untuk konstruktor default adalah jenis konstruktor yang disebut secara otomatis ketika pemrogram belum mendefinisikan konstruktor dalam program. Sedangkan konstruktor berparameter adalah jenis konstruktor yang ditentukan oleh pemrogram dengan melewatkan parameter untuk memberikan nilai awal pada variabel instan pada class. Dan saat instansiasi konstruktor default memanggil satu persatu sedangkan kontruktor berparameter langsung menuliskan parameternya. Contoh: Laptop l = new Laptop("Thinkpad", p);
3. Perhatikan *class* Laptop, di antara 2 atribut yang dimiliki (*merk* dan *proc*), atribut manakah yang bertipe *object* ?

   Jawab : yang bertipe objek adalah proc, karena proc menggunakan tipe data Procesor(nama dari class)
4. Perhatikan *class* Laptop, pada baris manakah yang menunjukan 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 menampilkan/memanggil method info 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 adalah nama objek/instansiasi dari class processor. Saat program dirubah tidak ada perubahan, dikarenakan p juga berisi (“Intel i5”,3) hanya saja supaya lebih ringkas maka hanya ditulis p

2.2 Percobaan 2

Pertanyaan :

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

   Jawab :
   ```
   private Mobil mobil;
   private Sopir 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 pada saat kita menyewa sopir dan mobil kita menghitung dari berapa hari kita menyewa lalu dikalikan dengan biaya(tarif) dari masing-masing

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

   Jawab : untuk menjumlahkan biaya dari mobil dan sopir, dengan memanggil method yang ada pada class mobil dan sopir

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

   Jawab : untuk memanggil method pada class pelanggan yang berelasi dengan class mobil dan sopir

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

   Jawab : untuk memanggil method hitungBiayaTotal pada class pelanggan dan method tersebut digunakan untuk menjumlahkan hitungBiayaMobil dan 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 memanggil merek mobil tersebut yaitu Avanza

2.2 Percobaan 3

Pertanyaan :

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

   Jawab : untuk memanggil info 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());
```

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

```
run:
Exception in thread "main" java.lang.NullPointerException
        at ac.id.polinema.percobaan3.KeretaApi.info(KeretaApi.java
        at ac.id.polinema.percobaan3.mainPercobaan3.main(mainPerco
Java Result: 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

Jawab :

Hasilnya akan error dikarenakan di class kereta api pada method info terdapat info asisten dan ketika dijalankan maka akan error karena pada method info tidak terpenuhi

4. Perbaiki *class* KeretaApi sehingga program dapat berjalan !

Jawab : kita dapat menghilangkan info asisten pada class kereta api

```
public 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;
}
```

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

atau bisa juga dengan menginputkan parameter asisten kembali

```
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());
```

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

Asisten: Nip: 4567
Nama: Patrick Star


BUILD SUCCESSFUL (total time: 1 second)
```

2.2 Percobaan 4

Pertanyaan :

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

   Jawab : terdapat 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 : ketika penumpang tidak kosong/telah menempati kursi maka akan ditampilkan info dari penumpang tersebut

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

   Jawab : karena ketika kita ngeset penumpang maka 1 nomer akan hilang karena sudah terpakai maka dari itu nilai nomer 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 : terdapat 2 output, yang pertama ketika kita menambahkan program diatas dengan satu sop info maka hasilnya kursi satu akan ditempati budi:

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

    Penumpang p1 = new Penumpang("1456","Budi");
    gerbong.setPenumpang(p1, 1);
    System.out.println(gerbong.info());
}
```

```
run:
Kode: A
Nomor: 1
Penumpang: Ktp: 1456
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)
```

Tetapi ketika kita menuliskan dua sop info maka kursi no 1 akan ditempati oleh Mr.Krab dan Budi

```
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());
        Penumpang budi = new Penumpang("1456","Budi");
        gerbong.setPenumpang(budi, 1);
        System.out.println(gerbong.info());
    }
}
```

```
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: 1456
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)
```

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

```
public void setPenumpang(Penumpang penumpang, int nomor) {
    if(penumpang != null){
        System.out.println("Sudah Terisi");
    }else{
    this.arrayKursi[nomor - 1].setPenumpang(penumpang);
    }
}
```

```
Sudah Terisi
Kode: A
Nomor: 1
Nomor: 2
Nomor: 3
Nomor: 4
Nomor: 5
Nomor: 6
Nomor: 7
Nomor: 8
Nomor: 9
Nomor: 10

BUILD SUCCESSFUL (total time: 2 seconds)
```

NAMA :
NIM :
KELAS :
MATERI :