Modul 3

Enkapsulasi

(Pembungkusan)

DAFTAR ISI

A. PENDAHULUAN

Kompetensi Dasar:
Mengaplikasikan enkapsulasi pada Java

1. **Identitas Modul**

   Mata Pelajaran : Pemrograman Berorientasi Objek
   Kelas : XI
   Judul Modul : Enkapsulasi/Pembungkusan (*Encapsulation*)

2. **Tujuan Pembelajaran**

   Setelah mempelajari bab ini diharapkan mahasiswa akan mampu :

   a. Menerapkan konsep *encapsulation*.
   b Menyembunyikan data.
   c Mengakses data yang disembunyikan

3. **Deskripsi Materi**

   Modul 3 yang berjudul *Encapsulation* ini memaparkan tentang bagaimana cara suatu data dapat disembunyikan dan bagaimana data yang disembunyikan itu dapat diakses. Saat suatu aplikasi didistribusikan kepada pengguna, maka *programmer* tidak perlu menunjukkan detail kode programnya, tetapi menjelaskan bagaimana cara menggunakan aplikasi tersebut. Seperti itulah ilustrasi penggunaan konsep *encapsulation*.

   Untuk menerapkan konsep *encapsulation* dalam membangun suatu program, maka Anda harus mendeklarasikan atribut dengan modifier private sehingga atribut tersebut tidak dapat diakses sembarangan oleh class lain dan membuat method tertentu untuk mengakses atribut tersebut bila diperlukan.

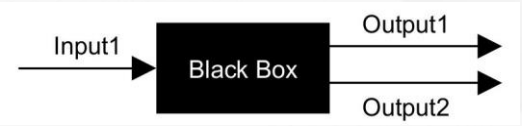
B. DASAR TEORI

1. **Konsep Enkapsulasi**

Enkapsulasi merupakan suatu cara pembungkusan data dan method yang menyusun suatu kelas sehingga kelas dapat dipandang sebagai suatu modul dan cara bagaimana menyembunyikan informasi detail dari suatu class (*information hiding*). Dalam OOP, enkapsulasi sangat penting untuk keamanan serta menghindari kesalahan permrograman, enkapsulasi dimaksudkan untuk menjaga suatu proses program agar tidak dapat diakses secara sembarangan atau di intervensi oleh program lain.

Dalam suatu obyek seringkali terdapat data-data privat yang hanya digunakan untuk operasi internal di dalam obyek tersebut. Data-data privat tersebut sebaiknya dilindungi untuk menghindari modifikasi dari obyek lain yang berpotensi mengganggu operasi internal dalam obyek. Perlindungan data-data privat juga dapat mengurangi kompleksitas kode karena pemrogram tidak perlu dipusingkan dengan data dan proses mikro. Sebaliknya, pemrogram hanya perlu fokus pada fungsi-fungsi publik yang memang disediakan sebagai sarana interaksi antar obyek.

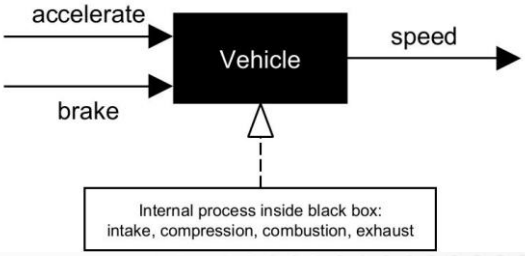
Enkapsulasi dapat merubah cara pandang kita terhadap obyek dimana suatu obyek dapat diibaratkan sebagai *black box*. Dengan paradigma obyek sebagai *black box*, pemrogram tidak perlu mengetahui proses apa saja yang terjadi di dalam obyek (*black box*). Pemrogram hanya perlu memberikan input terhadap *black box* dan mengambil output dari proses yang dilakukan dalam *black box*.



Ilustrasi black box pada enkapsulasi

Sebagai contoh pada kelas Vehicle (kendaraan) adalah sbb:

- Input dapat berupa fungsi-fungsi accelerate (akselerasi), brake (rem), dsb.
- Output dapat berupa getSpeed (mendapatkan kecepatan), dsb.
- Operasi internal kendaraan yang berada di dalam black box dapat berupa segala mekanisme teknis yang terjadi di dalam mesin seperti intake (pengambilan bahan bakar), compression (kompresi bahan bakar), combustion (pembakaran), exhaust (pembuangan), dsb.



Contoh enkapsulasi (black box) pada kelas Vehicle

Untuk mengatur hak akses dari masing-masing method dan atribut digunakan modifier akses. Modifier akses yang paling umum adalah public yang artinya dapat diakses dari obyek manapun, dan private yang artinya hanya bisa diakses dalam obyek yang bersangkutan.

Konsep enkapsulasi sangat penting dilakukan untuk menjaga kebutuhan program agar dapat diakses sewaktu-waktu, sekaligus menjaga program tersebut.

2. **Penyembunyian Data dan Pengaksesan Data**

Dua hal yang mendasar dalam enkapsulasi adalah adanya data yang disembunyikan dan bagaimana cara untuk mengakses data tersebut. *Information hiding* adalah proses menyembunyikan informasi dari suatu *class* sehingga informasi tersebut tidak dapat diakses dari luar dengan cara memberikan *modifier* "*private*" ketika mendeklarasikan *attribute* atau *method*. *Interface to access data* adalah cara melakukan perubahan terhadap *attribute* yang telah disembunyikan. Caranya yaitu dengan membuat *interface* berupa *method* untuk menginisialisasi atau merubah nilai dari *attribute* tersebut

- *Information Hiding*

Sebelumnya untuk pengaksesan atribut atau method menggunakan objek secara langsung. Hal ini karena akses kontrol yang diberikan pada atribut dan method di dalam kelas tersebut adalah *public*. Untuk menyembunyikan informasi dari suatu kelas sehingga anggota kelas tersebut tidak dapat diakses kelas lain yaitu dengan memberi hak akses *private* pada atributnya. Proses ini disebut dengan *information hiding*.

- *Interface to Access Data*

Interface to access data ini merupakan cara melakukan perubahan terhadap atribut yang disembunyikan, caranya adalah dengan membuat suatu *interface* berupa *method* untuk menginisialisasi atau merubah nilai dari suatu atribut tersebut. Manfaat utama teknik *encapsulation* adalah kita mampu memodifikasi kode tanpa merusak kode yang telah digunakan pada *class* lain.

3. **Jenis Enkapsulasi**

Ada 3 jenis dalam oop untuk mengatur hak akses dari suatu property dan method, yaitu :

a. Public. Ketika sebuah property atau method dinyatakan sebagai public, maka seluruh kode program di luar class bisa mengaksesnya, termasuk class turunan.

b. Protected, jika sebuah property atau method dinyatakan sebagai protected, berarti property atau method tersebut tidak bisa diakses dari luar class, namun bisa diakses oleh class itu sendiri atau turunan class tersebut. Apabila kita mencoba mengakses protected property atau protected method dari luar class, akan menghasilkan error

b Private, Jika sebuah property atau method di-set sebagai private, maka satu-satunya yang bisa mengakses adalah class itu sendiri. Class lain tidak bisa mengaksesnya, termasuk class turunan.

4. **Manfaat Enkapsulasi**

Enkapsulasi memiliki manfaat sebagi berikut:

- Bersifat independen

  Suatu modul yang terencapsulasi dengan baik akan bersifat independen, sehingga tidak akan terikat pada bagian tertentu dari program.

- Bersifat transparan

  Bila melakukan modifikasi pada suatu modul, maka perubahan tersebut akan dirasakan juga oleh bagian program yang menggunakan modul tersebut.

- Menghindari efek diluar perencanaan

  Modul yang terencapsulasi dengan baik hanya akan berinteraksi dengan bagian program lainnya melalui variable-variabel input/output yang telah didefinisikan sebelunnya.

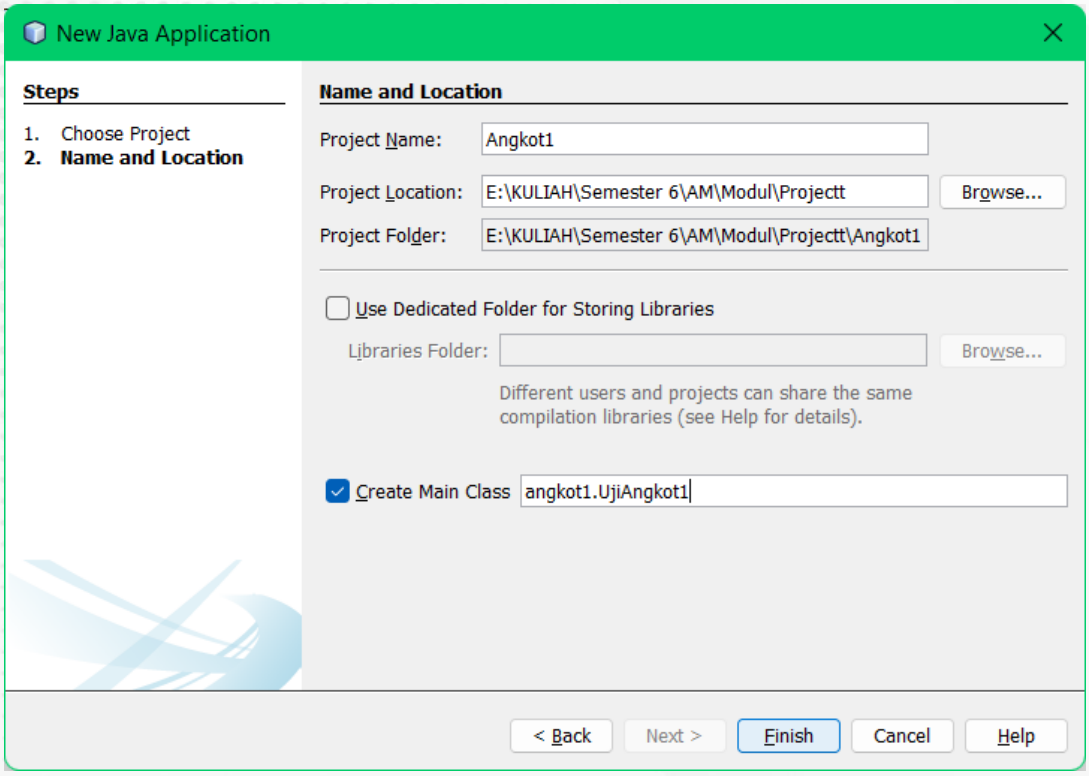
- Melindungi listing program

  Saat program didistribusikan pada khalayak, untuk melindungi listing program Anda dapat menerapkan prinsip enkapsulasi. Di sini pengguna hanya dapat menggunakan program melalui variable input atau output yang didefinisikan tanpa disertai bagimana proses yang terjadi di dalam modul tersebut

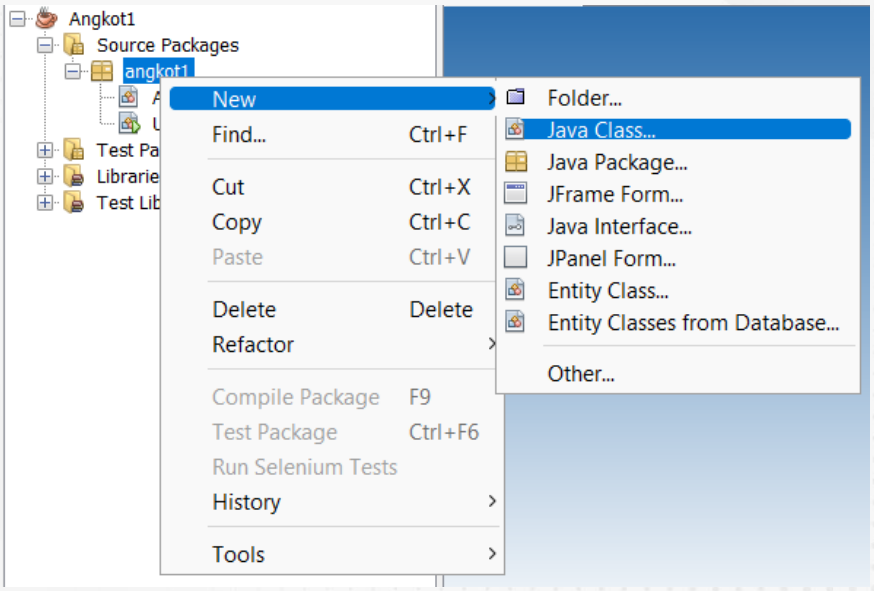
C. TUGAS PRAKTIKUM

1. **Program Tanpa Penerapan *Encapsulation***

   Buatlah Project Baru di Netbeans dengan nama `Angkot1`, kemudian untuk Main Class berinama `angkot1.UjiAngkot1`



Kemudian buatlah sebuah class baru dengan nama `Angkot1`



Tulislah dan simpan kode program di bawah ini:

Kode program *class* `Angkot1.java`

```java
package angkot1;

public class Angkot1 {
    public int penumpang=2;
    public int maxPenumpang=10;
    public void cetak(){
        System.out.println("Penumpang bus mini sekarang adalah : "+penumpang);
        System.out.println("Penumpang maksimal seharusnya adalah : "+maxPenumpang);
    }
}
```

Kode program *class* `UjiAngkot1.java`

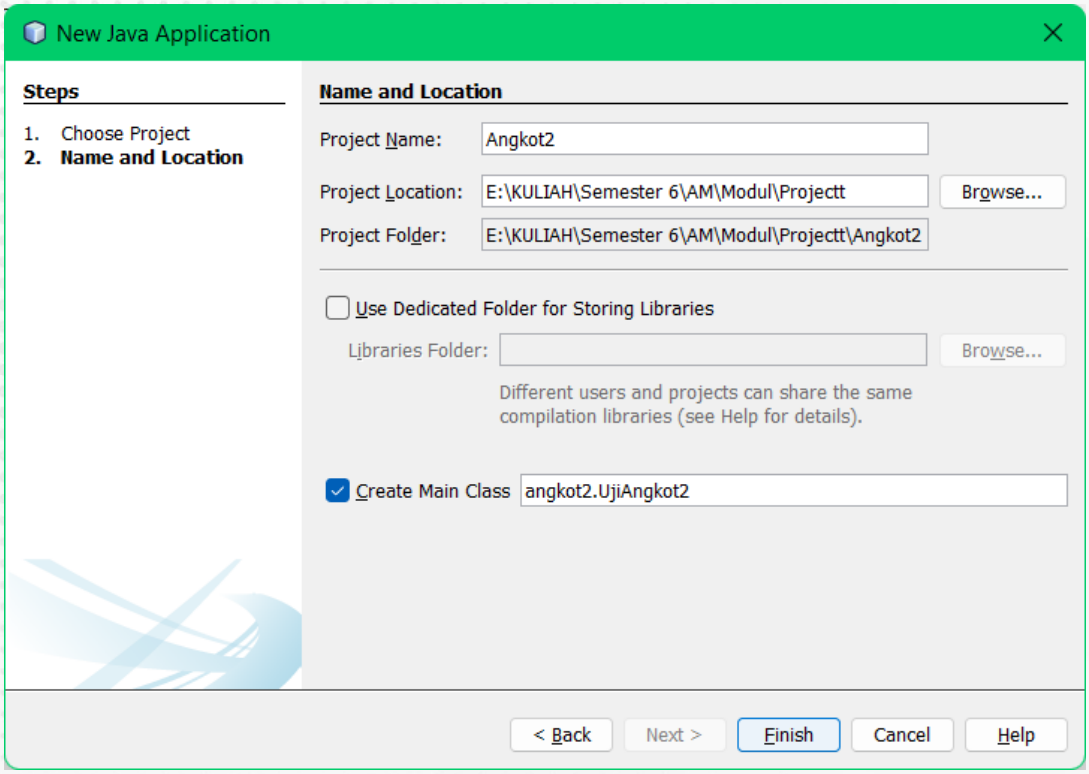
```java
package angkot1;

public class UjiAngkot1 {
    public static void main(String[] args) {
        System.out.println("\n**** PENUMPANG ANGKOT MINI LANDUNGSARI **** ");
        Angkot1 angkotMini = new Angkot1();
        System.out.println("\n\tJumlah Penumpang Awal");
        angkotMini.cetak();
        angkotMini.penumpang = angkotMini.penumpang + 7;
        System.out.println("\n\tJumlah Penumpang Setelah Ditambah 7");
        angkotMini.cetak();
        angkotMini.penumpang = angkotMini.penumpang - 4;
        System.out.println("\n\tJumlah Penumpang Setelah Dikurang 3");
        angkotMini.cetak();
        angkotMini.penumpang = angkotMini.penumpang + 7;
        System.out.println("\n\tJumlah Penumpang Setelah Ditambah 7");
        angkotMini.cetak();
    }
}
```

a. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!

b. Berikan penjelasan terkait jalannya program ini!

2. **Program dengan Penerapan *Encapsulation***

Buatlah Project Baru di Netbeans dengan nama `Angkot2`, kemudian untuk Main Class berinama `angkot2.UjiAngkot2`



Kemudian buatlah sebuah class baru dengan nama `Angkot2`

Tulislah dan simpan kode program di bawah ini:

Kode program *class* `Angkot2.java`

Angkot2.java × UjiAngkot2.java ×

```
1   package angkot2;
2
3   public class Angkot2 {
4       private int penumpang=1;
5       private int maxPenumpang=15;
6       public void addPenumpang(int penumpang) {
7           int add;
8           add = this.penumpang+penumpang;
9           if(add>maxPenumpang){
10              System.out.println("Penumpang Angkot Besar Sekarang : "+add);
11              System.out.println("Penumpang Maksimal Seharusnya   : "+maxPenumpang);
12              System.out.println("Jumlah Penumpang Melebihi Batas");
13          }else{
14              this.penumpang = add;
15          }
16      }
17      public void cetak() {
18          System.out.println("Penumpang Angkot Besar Sekarang : "+penumpang);
19          System.out.println("Penumpang Maksimal Seharusnya   : "+maxPenumpang);
20      }
21  }
```

Kode program *class* `UjiAngkot2.java`

Angkot2.java | UjiAngkot2.java

```java
package angkot2;

public class UjiAngkot2 {
    public static void main(String[] args) {
        System.out.println("\n**** PENUMPANG ANGKOT BESAR LANDUNGSARI **** ");
        Angkot2 angkotBesar = new Angkot2();
        System.out.println("\n\tJumlah Penumpang Awal");
        angkotBesar.cetak();
        System.out.println("\n\tJumlah Penumpang Setelah Ditambah 10");
        angkotBesar.addPenumpang(10);
        angkotBesar.cetak();
        System.out.println("\n\tJumlah Penumpang Setelah Dikurang 3");
        angkotBesar.addPenumpang(-3);
        angkotBesar.cetak();
        System.out.println("\n\tJumlah Penumpang Setelah Ditambah 9");
        angkotBesar.addPenumpang(9);
        angkotBesar.cetak();
    }
}
```

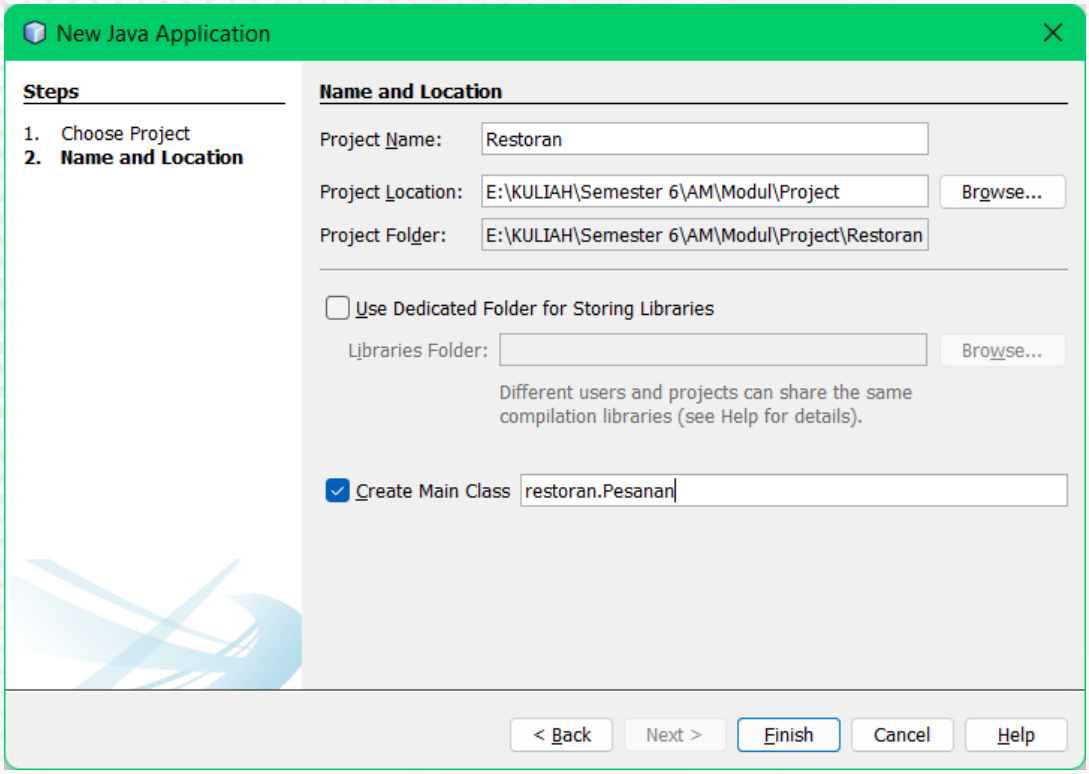
a. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
b. Berikan penjelasan terkait jalannya program ini!

Dari kedua program yang menerapkan dan tidak menerapkan prinsip enkapsulasi memiliki perbedaan. Dimana pada program yang tidak menerapkan prinsip enkapsulasi penambahan penumpang tetap terjadi meskipun sudah melebihi kapasitas penumpang.

Sedangkan pada program yang menerapkan prinsip enkapsulasi, program menjaga suatu proses program agar tidak dapat diakses secara sembarangan atau di intervensi oleh program lain. Sehingga ketika dalam contoh hal ini jumlah angkot memenuhi kapasitas, maka penumpang tidak bisa masuk dan tidak terjadi penambahan penumpang.

3. **Program Pemesanan di Restoran**

Buatlah Project Baru di Netbeans dengan nama `Restoran`, kemudian untuk Main Class berinama `restoran.Pesanan`



Kemudian buatlah sebuah class baru dengan nama `Restoran`

Tulislah dan simpan kode program di bawah ini:

Kode program *class* `Restoran.java`

Restoran.java | Pesanan.java

```java
package restoran;
public class Restoran {
    //Variable (Private)
    private String makanan;
    private String minuman;
    private int hargaMakanan;
    private int hargaMinuman;
    private int totalHarga;

    //Method Setter (Public) dengan Parameter
    public void setMakanan(String makanan){
        this.makanan = makanan;
    }
    public void setMinuman(String minuman){
        this.minuman = minuman;
    }
    public void setHargaMakanan(int hargaMakanan){
        this.hargaMakanan = hargaMakanan;
    }
    public void setHargaMinuman(int hargaMinuman){
        this.hargaMinuman = hargaMinuman;
    }
    public void setHargaTotal(int totalHarga){
        this.totalHarga = totalHarga;
    }

    //Method Getter (Public)
    public String getMakanan(){
        return makanan;
    }
    public String getMinuman(){
        return minuman;
    }
    public int getHargaMakanan(){
        return hargaMakanan;
    }
    public int getHargaMinuman(){
        return hargaMinuman;
    }
    public int getTotalHarga(){
        totalHarga = hargaMakanan+hargaMinuman;
        return totalHarga;
    }
}
```

Kode program *class* `Pesanan.java`

Restoran.java | Pesanan.java

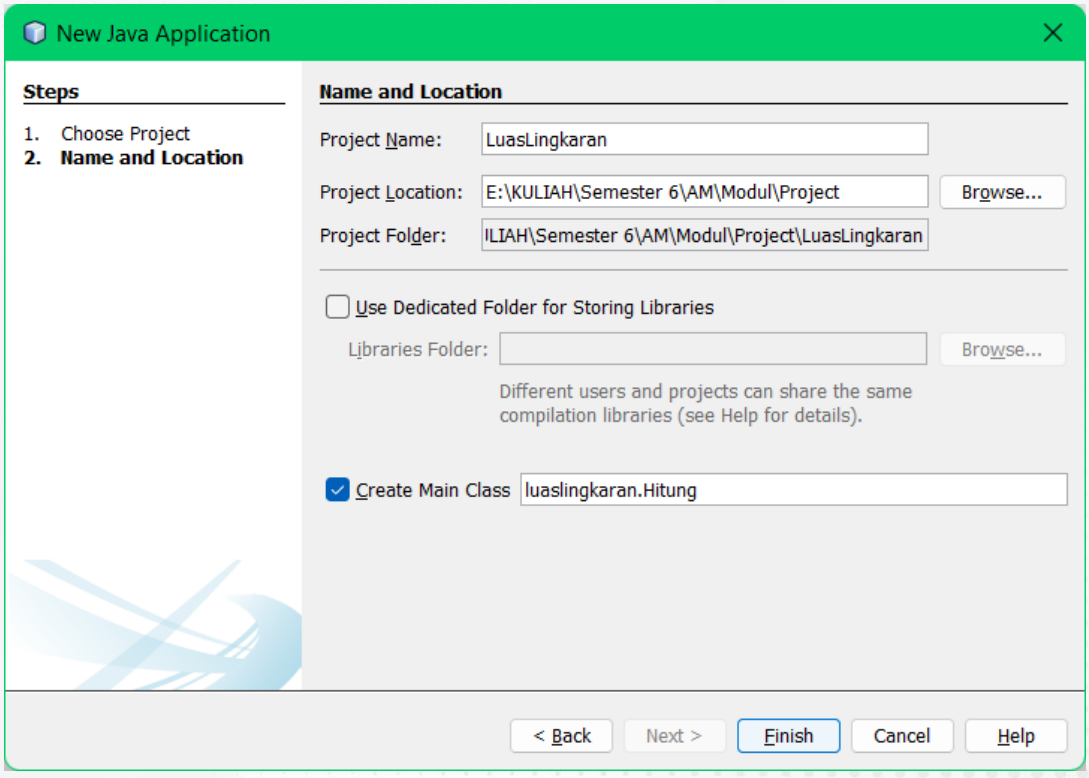
```
1  package restoran;
2  public class Pesanan {
3      public static void main(String[] args) {
4          //Membuat Instence/Objek dari Class restoran
5          Restoran data = new Restoran();
6
7          //Memasukan Data pada Variable
8          data.setMakanan("Coto Makassar");
9          data.setHargaMakanan(25000);
10         data.setMinuman("Greentea Latte");
11         data.setHargaMinuman(15000);
12
13         //Memanggil Method Get dari Class Restoran dan Menampilkannya
14         System.out.println("Menu Makanan  : "+data.getMakanan());
15         System.out.println("Harga Makanan : Rp."+data.getHargaMakanan());
16         System.out.println("Menu Minuman  : "+data.getMinuman());
17         System.out.println("Harga Minuman : Rp."+data.getHargaMinuman());
18         System.out.println("Total Harga   : Rp."+data.getTotalHarga());
19     }
20 }
```

a. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!

b. Berikan penjelasan terkait jalannya program ini!

4. **Program Luas Lingkaran**

Buatlah Project Baru di Netbeans dengan nama `LuasLingkaran`, kemudian untuk Main Class berinama `luaslingkaran.Hitung`



Kemudian buatlah sebuah class baru dengan nama `Lingkaran`

Tulislah dan simpan kode program di bawah ini:

Kode program *class* `Lingkaran.java`

Hitung.java | Lingkaran.java

```java
package luaslingkaran;
public class Lingkaran {
    // atribut
    private double jarijari;
    private double luas;

    // setter method untuk jejari
    public void setJariJari(int r){
        if (r > 0){
            this.jarijari = r;
        } else {
            this.jarijari = 0;
        }
    }

    // getter method untuk luas
    public double getLuas(){
        // hitung luasnya
        this.luas = Math.PI * Math.pow(this.jarijari, 2);
        return this.luas;
    }
}
```

Kode program *class* `Hitung.java`

Hitung.java | Lingkaran.java

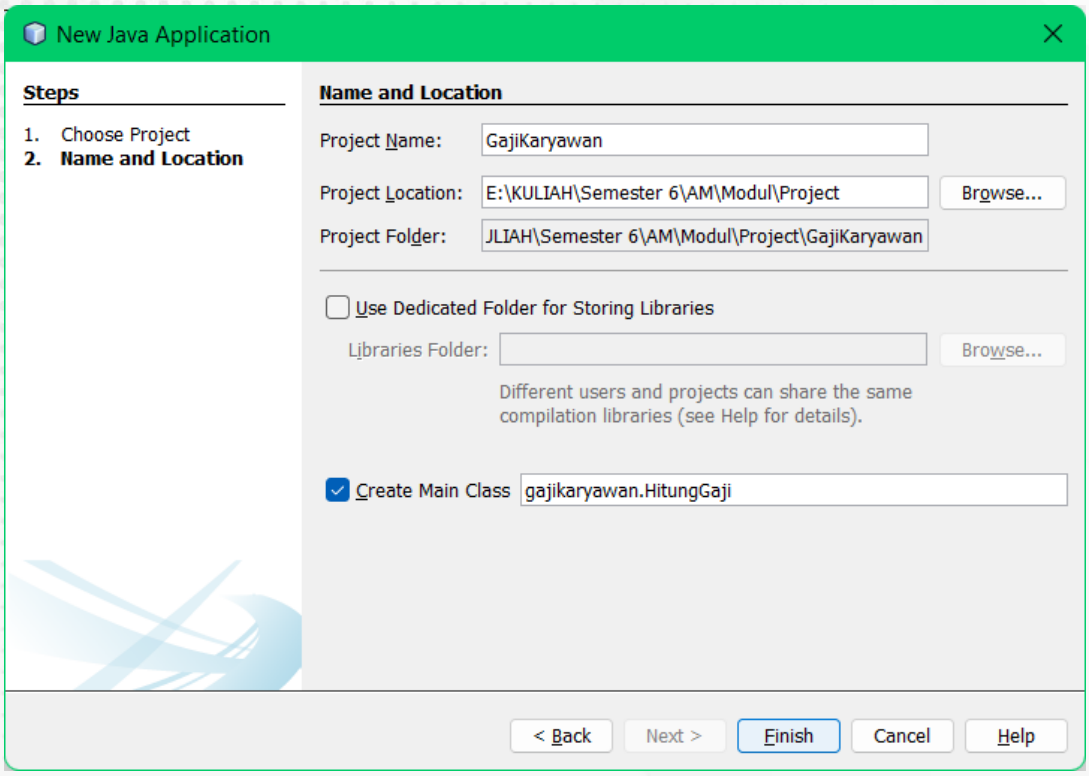
```java
package luaslingkaran;
public class Hitung {
    public static void main(String[] args) {
        // hitung luas lingkaran
        Lingkaran bentuk = new Lingkaran();
        bentuk.setJariJari(7);
        double luas = bentuk.getLuas();

        // Tampilkan
        System.out.println("PROGRAM HITUNG LUAS LINGKARAN");
        System.out.println("----------------------------");
        System.out.println("Luas Lingkaran : " + luas + " cm2");
    }
}
```

a. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!

b. Berikan penjelasan terkait jalannya program ini!

5. **Program Gaji Karyawan**

Buatlah Project Baru di Netbeans dengan nama `GajiKaryawan`, kemudian untuk Main Class berinama `gajikaryawan.HitungGaji`



Kemudian buatlah sebuah class baru dengan nama `GajiKaryawan`

Tulislah dan simpan kode program di bawah ini:

Kode program *class* `GajiKaryawan.java`

GajiKaryawan.java | HitungGaji.java

```java
package gajikaryawan;
public class GajiKaryawan {
    private double gajiKotor, gajiBersih, pajak;
    private double potongan = 50000;
    private String nama = "Hanif";
    public void setGaji(double gaji) {
        gajiKotor=gaji;
    }
    public void hitungPajak() {
        pajak = 0.02*gajiKotor;
    }
    public void hitungGaji() {
        gajiBersih = gajiKotor-pajak-potongan;
    }
    public String namaPegawai(){
        return nama;
    }
    public double getGajiKotor(){
        return gajiKotor;
    }
    public double getPajak(){
        return pajak;
    }
    public double getPotongan(){
        return potongan;
    }
    public double getGajiBersih(){
        return gajiBersih;
    }
}
```

Kode program *class* `HitungGaji.java`

GajiKaryawan.java | HitungGaji.java

```java
package gajikaryawan;
import java.util.Scanner;

public class HitungGaji {
    public static void main(String[] args) {
        double gaji;
        GajiKaryawan hasilgaji = new GajiKaryawan();
        Scanner berimasukan = new Scanner(System.in);
        System.out.println("\n\t=== PROGRAM HITUNG GAJI KARYAWAN ===");
        System.out.println("\nNama Pegawai        : "+hasilgaji.namaPegawai());
        System.out.print("Gaji Pegawai        : Rp. ");
        gaji = berimasukan.nextDouble();
        hasilgaji.setGaji(gaji);
        hasilgaji.hitungPajak();
        hasilgaji.hitungGaji();
        System.out.println("--------------------");
        System.out.println("Total Gaji Kotor    : Rp. "+gaji);
        System.out.println("Pajak Yang Diterima : Rp. "+hasilgaji.getPajak());
        System.out.println("Pajak Potongan Gaji : Rp. "+hasilgaji.getPotongan());
        System.out.println("                      _____________ -");
        System.out.println("Besar Gaji Bersih   : Rp. "+hasilgaji.getGajiBersih());
    }
}
```

a. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
b. Lakukan kompilasi dan eksekusi program terhadap *class* TestSiswa kemudian tunjukkan hasilnya!
c. Tentukan *method* mana saja yang termasuk *mutator method* dan *method* mana saja yang termasuk *accessor method*!

D. EVALUASI

1. **Program Toko Buku**

   Tulislah dan simpan kode program di bawah ini:

   Kode program *class* `Buku.java`

Buku.java | BeliBuku.java

```
1   package buku;
2   public class Buku {
3       private String judulBuku="Pemrogaman Java";
4       private String pengarangBuku="Abdul";
5       private int stokBuku=37;
6       private int hargaBuku=75000;
7
8       public void setPembeli(String nama) {
9           namaPembeli = nama;
10      }
11      public void setAlamat(String alamat) {
12          alamatPembeli = alamat;
13      }
14      public void setPembelian(int pembelian) {
15          banyakPembelian = pembelian;
16      }
17      public void hitungBayar(){
18          bayarBuku= hargaBuku*banyakPembelian;
19      }
20      public void hitungSisa() {
21          sisaBuku = stokBuku - banyakPembelian;
22      }
23      public String getnamaPembeli(){
24          return namaPembeli;
25      }
26      public int getbayarBuku(){
27          bayarBuku = hargaBuku*banyakPembelian;
28          return bayarBuku;
29      }
30      public int getsisaBuku(){
31          sisaBuku = stokBuku-banyakPembelian;
32          return sisaBuku;
33      }
34  }
```

   a. Lengkapi kode program di atas untuk membuat sebuah program berbasis console di bidang pertokoan! Terapkan penggunaan konsep encapsulation! Buatlah sebuah class lagi yang berisi method main untuk menjalankan program tersebut! Buatlah agar program dapat menerima input dari user! Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!

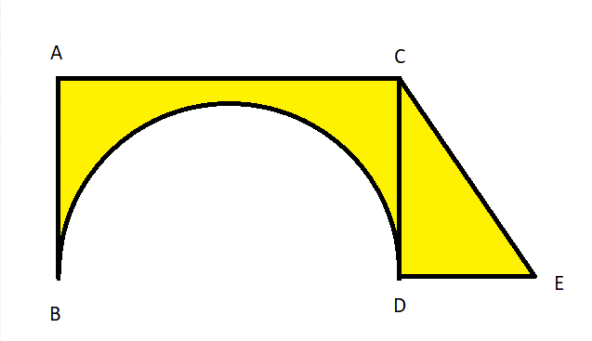
   b. Berikan penjelasan terkait jalannya program yang Anda buat!

      Contoh hasil eksekusi program:

```
Output - Buku (run)
run:

        DATA BUKU YANG AKAN DIBELI
Judul Buku        : Buku Pertama Belajar Pemrogaman Java
Nama Pengarang    : Abdul Kadir
Stok Awal Buku    : 37
Harga Buku        : Rp. 75000

        DATA PEMBELI
Nama Pembeli      : Hanif Sjahbandi
Alamat Pembeli    : Gading Kasri
Banyak Pembelian : 5

        LAPORAN PEMBELIAN
Nama Pembeli      : Hanif Sjahbandi
Alamat Pembeli    : Gading Kasri
Banyak Pembelian : 5
Total Pembelian   : Rp. 375000

        LAPORAN STOK AKHIR
Stock Akhir Buku : 32 buah
BUILD SUCCESSFUL (total time: 36 seconds)
```

2. **Program Hitung Luas Bidang**

Diketahui sebuah obyek bangun datar sebagaimana tampak pada gambar di bawah ini.



Jika diketahui panjang dan lebar persegi panjang ABCD adalah 15 cm x 8 cm, tinggi dan alas segitiga CDE adalah 8 cm x 5 cm, dan jari-jari lingkaran adalah 7 cm. Buatlah program Java untuk menghitung luas bangun daerah yang berwarna kuning.

Pada studi kasus di atas, kita dapat identifikasi bahwa terdapat beberapa gabungan bentuk bangun datar, kita akan mendapatkan luas daerah yang berwarna kuning dengan menjumlahkan luas dari segitiga CDE dengan luas persegi panjang ABCD dikurangi luas setengah lingkaran.

Lengkapi kode program di bawah ini:

Kode program *class* `Segitiga.java`

LuasBidang.java | Segitiga.java | Persegi.java | SetLingkaran.java

```java
package luasbidang;
public class Segitiga {
    // atribut

    private int tinggi;


    // setter method untuk alas
    public void setAlas(int a){
        if (a > 0){
            this.alas = a;
        } else {
            this.alas = 0;
        }
    }

    // getter method untuk luas
    public double getLuas(){
        // hitung luasnya
        this.luas = this.alas * this.tinggi * 0.5;
        return this.luas;
    }
}
```

Kode program *class* `Persegi.java`

LuasBidang.java | Segitiga.java | Persegi.java | SetLingkaran.java

```java
package luasbidang;
public class Persegi {
    // atribut
    private int panjang;
    private int lebar;


    // getter method untuk luas
    public double getLuas(){
        // hitung luasnya
        this.luas = this.panjang * this.lebar;
        return this.luas;
    }
}
```

Kode program *class* `SetLingkaran.java`

LuasBidang.java | Segitiga.java | Persegi.java | **SetLingkaran.java**

```
1   package luasbidang;
2   public class SetLingkaran {
3       // atribut
4
5
6       // getter method untuk luas
7       public double getLuas(){
8           // hitung luasnya
9           this.luas = Math.PI * Math.pow(this.jejari, 2);
10          return this.luas;
11      }
12  }
```

Kode program *class* `LuasBidang.java`

LuasBidang.java | Segitiga.java | Persegi.java | SetLingkaran.java

```
1   package luasbidang;
2   public class LuasBidang {
3       public static void main(String[] args) {
4           // hitung luas segitiga
5           Segitiga st = new Segitiga();
6           st.setAlas(5);
7
8
9           // hitung luas persegipanjang
10          pp.setLebar(8);
11          double luasPersegi = pp.getLuas();
12
13          // hitung luas setengah lingkaran
14          double luasSetLing = 0.5 * setLing.getLuas();
15
16          // hitung luas daerah warna kuning
17          double luasKuning = luasSegitiga + luasPersegi - luasSetLing;
18      }
19  }
```

a. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!

b. Berikan penjelasan terkait jalannya program yang Anda buat!