1. **DEFINISI MASALAH**

Soal 1

Tugas Anda adalah membuat sebuah class yang memuat data-data pada buku alamat.

Tabel berikut mendefinisikan informasi yang dimiliki oleh buku alamat.

|  |  |
| --- | --- |
| Attribut | Deskripsi |
| Nama | Nama Lengkap perseorangan |
| Alamat | Alamat Lengkap |
| Nomor Telepon | Nomor telepon personal |
| Alamat E-Mail | Alamat E-Mail personal |

Tabel 1: Atribut dan Deskripsi Atribut

Buat implementasi dari method sebagai berikut :

1. Menyediakan accessor dan mutator method terhadap seluruh atribut

2. Constructor

1. **SOURCE CODE**

Masalah 1

|  |  |
| --- | --- |
| MainMahasiswa.java | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62 | import java.util.Scanner;  public class MainMahasiswa {  static Scanner sc = new Scanner(System.in);  static Scanner sc2 = new Scanner(System.in);  static int pilih;  static InputDataMahasiswa input = new InputDataMahasiswa();  public static void main(String[] args) {  menuutama();  }  static void menuutama() {  System.out.println("1 .Menambah Data");  System.out.println("2. Menghapus Data");  System.out.println("3. Melihat Data");  System.out.println("4. keluar");  System.out.print("pilih : ");  pilih = sc.nextInt();  switch (pilih) {  case 1:  System.out.print("Masukan banyaknya mahasiswa : ");  int masuk = sc.nextInt();  for (int i = 0; i < masuk; i++) {  System.out.print("Nama " + (i + 1) + " : ");  String namay = sc2.nextLine();  System.out.print("Alamat " + (i + 1) + " : ");  String alamay = sc2.nextLine();  System.out.print("Telepon " + (i + 1) + " : ");  String telepoy = sc2.nextLine();  System.out.print("Email " + (i + 1) + " : ");  String emaiy = sc2.nextLine();  input.insertData(namay, alamay, telepoy, emaiy);  System.out.println("-----------------------");  }  menuutama();  break;  case 2:  System.out.print("mahasiswa keberapa yang ingin dihapus ?");  pilih = sc.nextInt();  input.deleteData(pilih-1);  menuutama();  break;  case 3:  System.out.println("Data buku alamat mahasiswa :");  System.out.println("-----------------------------");  for (mhs mhsw : input.getALL()) {  System.out.println("Nama : " + mhsw.getNama());  System.out.println("Alamat : " + mhsw.getAlamat());  System.out.println("Telepon : " + mhsw.getTelepon());  System.out.println("Email : " + mhsw.getEmail());  System.out.println("-----------------------------");  }  menuutama();  break;  case 4:  System.out.println("program terhenti");  break;  default:  System.out.println("program tidak ditemukan");  menuutama();  break;  }  }  } |

|  |  |
| --- | --- |
| InputDataMahasiswa.java | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | import java.util.ArrayList;  public class InputDataMahasiswa {  ArrayList<mhs> listmahasiswa;  public InputDataMahasiswa() {  listmahasiswa = new ArrayList();  }  public void insertData(String nama, String alamat, String telepon, String email) {  mhs mhsw = new mhs(nama, alamat, telepon, email);  listmahasiswa.add(mhsw);  }  public ArrayList<mhs> getALL() {  return listmahasiswa;  }  public void deleteData(int index) {  listmahasiswa.remove(index);  }  } |

|  |  |
| --- | --- |
| mhs.java | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | public class mhs {  private String nama, alamat, telepon, email;  public mhs(String nama, String alamat, String telepon, String email) {  this.nama = nama;  this.alamat = alamat;  this.telepon = telepon;  this.email = email;  }  public String getNama() {  return nama;  }  public String getAlamat() {  return alamat;  }  public String getTelepon() {  return telepon;  }  public String getEmail() {  return email;  }  } |

1. **PEMBAHASAN**

Masalah 1

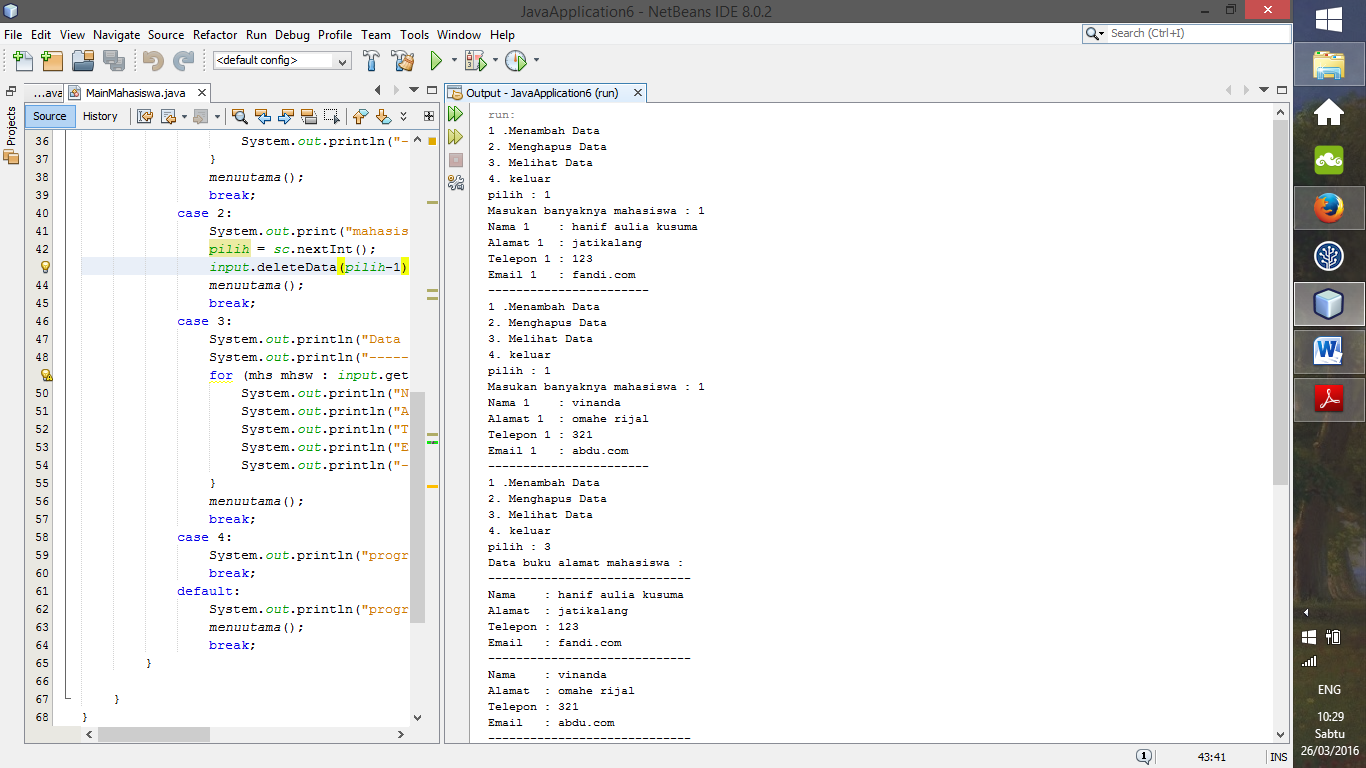
|  |  |
| --- | --- |
| MainMahasiswa.java | |
| 1  2  3  4  5  6  7  8  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  51  52  53  54  55  56  57  58  59 | Deklarasi fungsi scanner dari library java  Nama class MainMahasiswa  Memanggil fungsi Scanner bervariabel sc bertipe static  Memanggil fungsi Scanner bervariabel sc2 bertipe static  Varabel pilih dengan tipe data integer dan bertipe static  Memanggil class InputDataMahasiswa dengan variable input dan bertipe static  Deklarasi main class  Memanggil method menuutama  Deklarasi method menuutama dengan tipe static  Mencetak 1. Menambah data  Mencetak 2. Menghapus Data  Mencetak 3. Melihat Data  Mencetak 4. Keluar  Mencetak pilih :  Menginisialisasi variable pilih dengan inputan  Syntax switch dengan parameter pilih  Pilihan 1  Mencetak Masukan banyaknya mahasiswa  Menginisialisasi variable masuk dengan inutan  Perulangan for menambah nilai variable I dari 0 hingga masuk  Mencetak nama +urutan nama+:  Penginisialisasisan variable namay dengan inputan  Mencetak alamat +urutan alamat+:  Penginisialisasisan variable alamay dengan inputan  Mencetak telepon+urutan telepon+:  Penginisialisasisan variable telepoy dengan inputan  Mencetak email+urutan email+:  Penginisialisasisan variable emaiy dengan inputan  Memanggil method insertData dengan parameter namay,alamay,telepoy,emaiy menggunakan variable input  Menctetak garis pembatas  Memanggil method menuutama  Syntax penghenti  Pilihan 2  Mencetak Mahasiswa keberapa yang ingin dihapus?  Penginisialisasian variable pilih dengan inputan  Memanggil method deleteData berparameter pilih-1 dengan menggunakan variable input  Memanggil method menuutama  Syntax pengheti  Pilihan 3  Menncetak Data buku alamat mahasiswa  Mencetak garis pembatas  Syntax for untuk kelas mhs bervariabel mhsw didalam method getALL yang menggunakan variable input  mencetak Nama : memanggil method getNama menggunakan variable mhsw  mencetak Alamat : memanggil method getAlamat menggunakan variable mhsw  mencetak Telepon : memanggil method getTelepon menggunakan variable mhsw  mencetak Email : memanggil method getEmail menggunakan variable mhsw  mencetak garis pembatas  memanggil method menuutama  syntax penghenti  pilihan 4  mencetak program terhenti  syntax penghenti  syntax jika yang dipilih selain 1,2,3,4  maka akan mencetak Program tidak ditemukan  memanggil method menuutama  syntax penghenti |

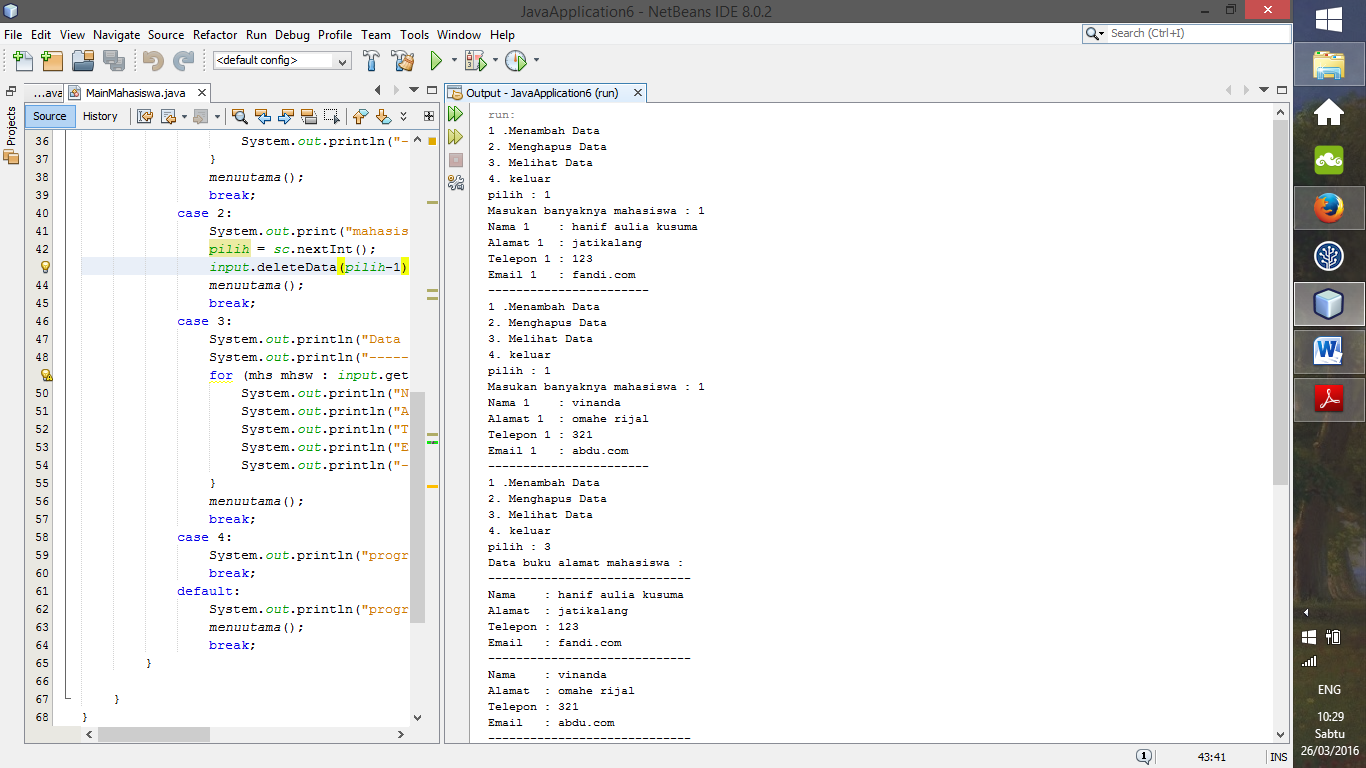
|  |  |
| --- | --- |
| InputDataMahasiswa.java | |
| 1  2  3  4  5  7  8  9  11  12  14  15 | Deklarasi fungsi ArrayList dari library java  Nama class InputDataMahasiswa  Menggunakan syntax ArrayList dengan tipe data class mhs dan variable listmahasiswa  Menggunakan constructor yang tidak berparameter  Instansiasi dari variable listmahasiswa  Method void insertData berparameter String nama,String alamat,String telepon, String email  Menginstansiasi class mhs dengan variable mhsw  Variable listmahasiswa menampung variable mhsw  Method dengan tipe Arraylist yang menggunakan kelas mhs sebagai tipe data dan variable getALL  Method ini memiliki nilai kembalian variabel listmahasiswa  Method void deleteData dengan parameter int index  Fungsi menghapus pada index tertentu pada fungsi ArrayList yang menggunakan variable listmahasiswa |

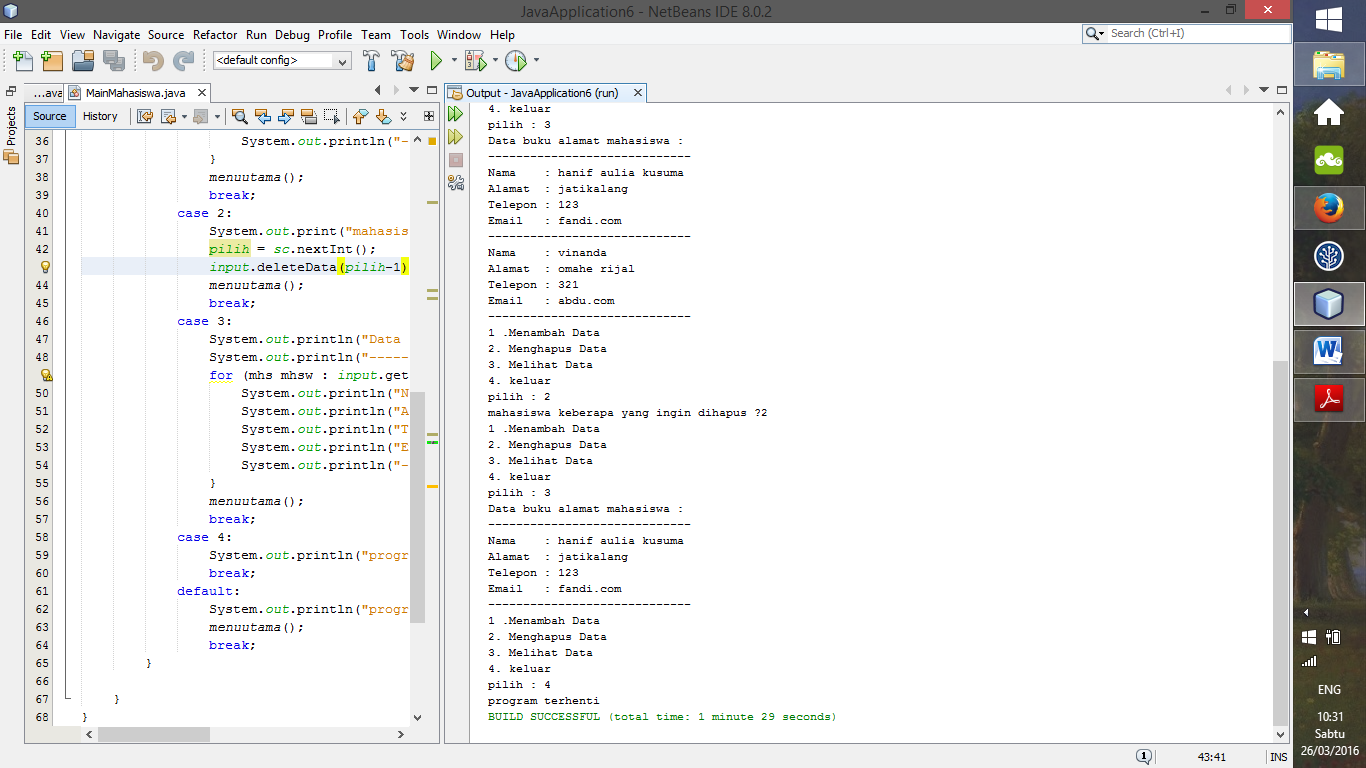
|  |  |
| --- | --- |
| mhs.java | |
| 1  2  3-4  5  6  7  8  10  11  13  14  16  17  19  20 | Nama class mhs  Tipe data String dengan nama variable nama,alamat,telepon,email  Constructor dengan parameter String nama,String alamat,String telepon,String email  Mendefinisikan variable nama = nema sebagai objek  Mendefinisikan variable alamat = alamat sebagai objek  Mendefinisikan variable telepon= telepon sebagai objek  Mendefinisikan variable email= email sebagai objek  Method String dengan variable getNama  Yang memilki nilai kembalian nama  Method String dengan variable getAlamat  Yang memilki nilai kembalian alamat  Method String dengan variable getTelepon  Yang memilki nilai kembalian telepon  Method String dengan variable getEmail  Yang memilki nilai kembalian email  //sengaja tanpa setter dikarenakan sudah ada constructor sebagai setternya |

1. **SCREENSHOT PROGRAM**

Masalah 1







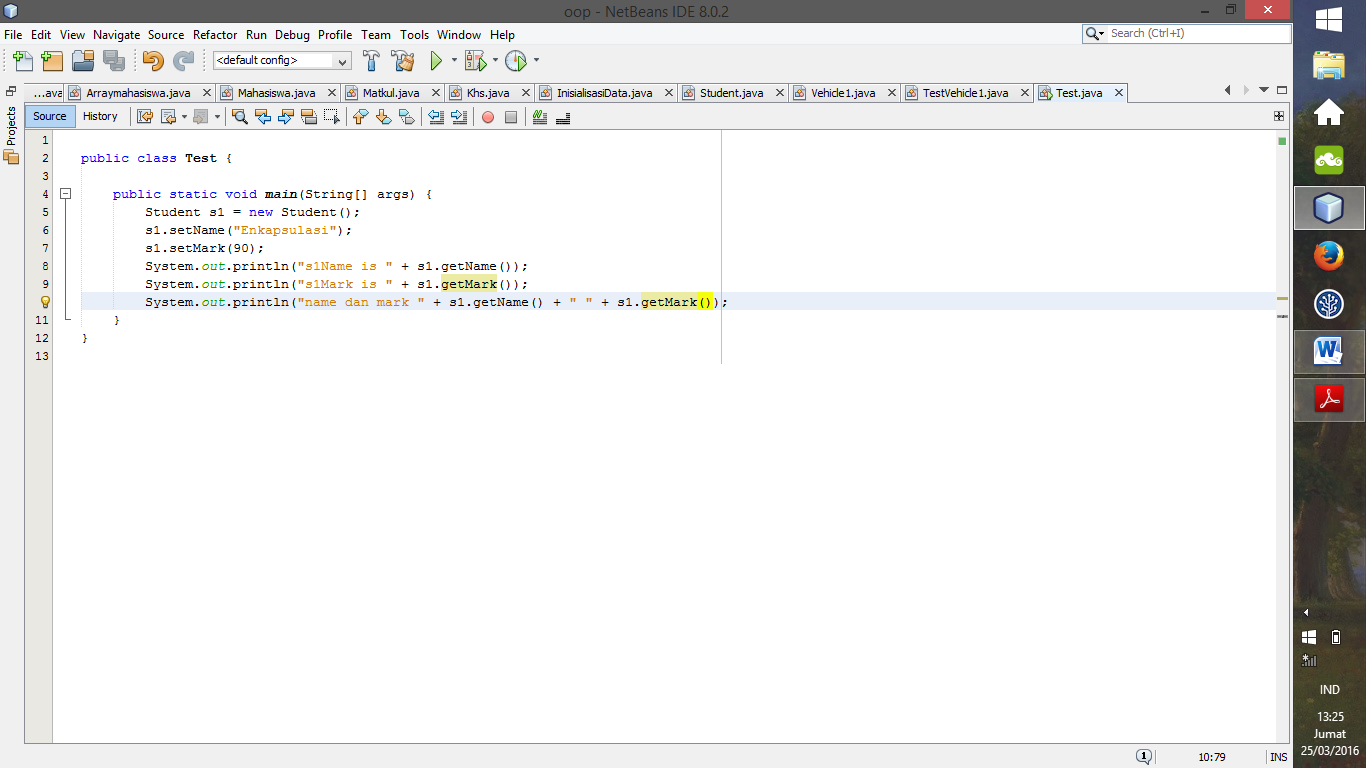
1. **PRAKTIKUM**
2. **Encapsulation 1**

Pertanyaan

1. Lakukan percobaan diatas dan benahi jika menemukan kesalahan!

**Jawab :** Penginisialisasian method yang dipanggil di class Test harus sesuai dengan tipe data parameter pada class Student

Pada class Student ubah syntaxnya menjadi seperti ini :

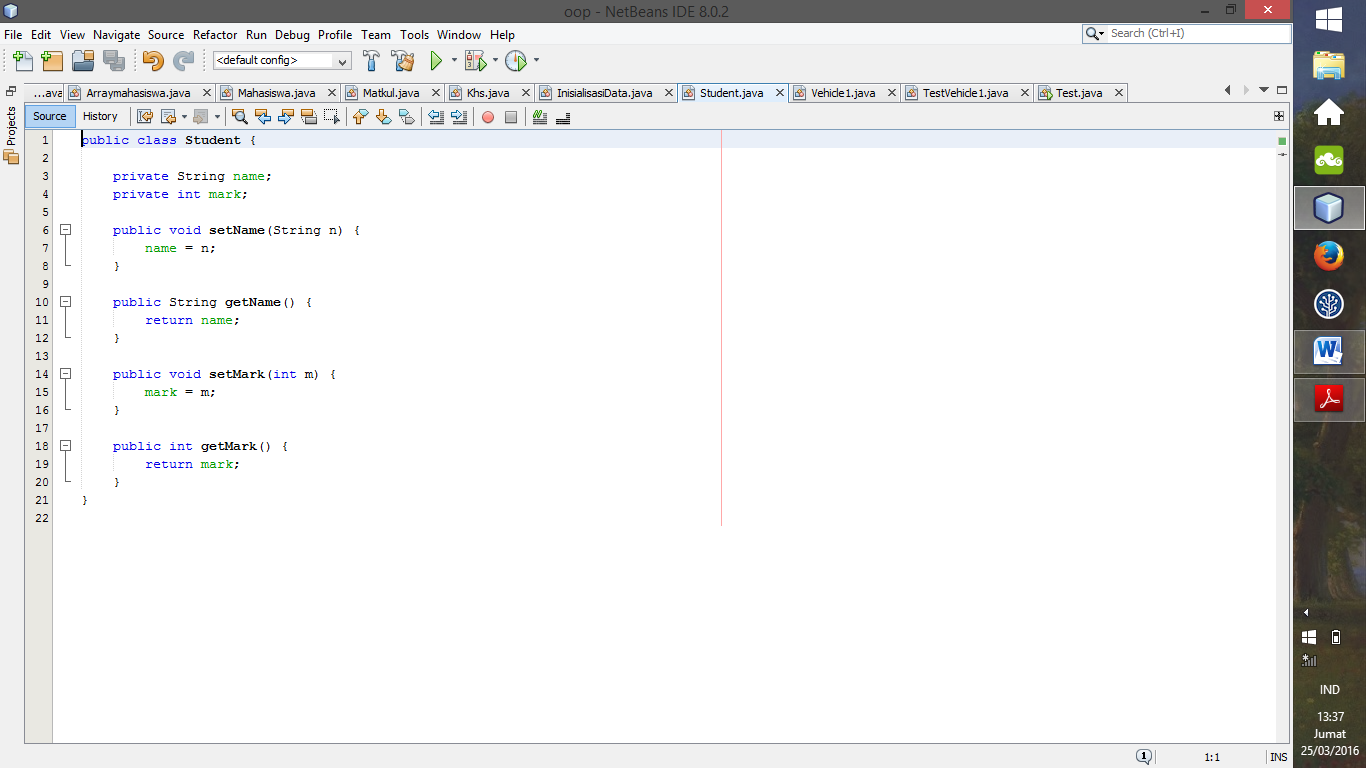


2. Jika pada baris 6 *s1.setName* diubah menjadi *s1.getName* apa yang terjadi? jelaskan!

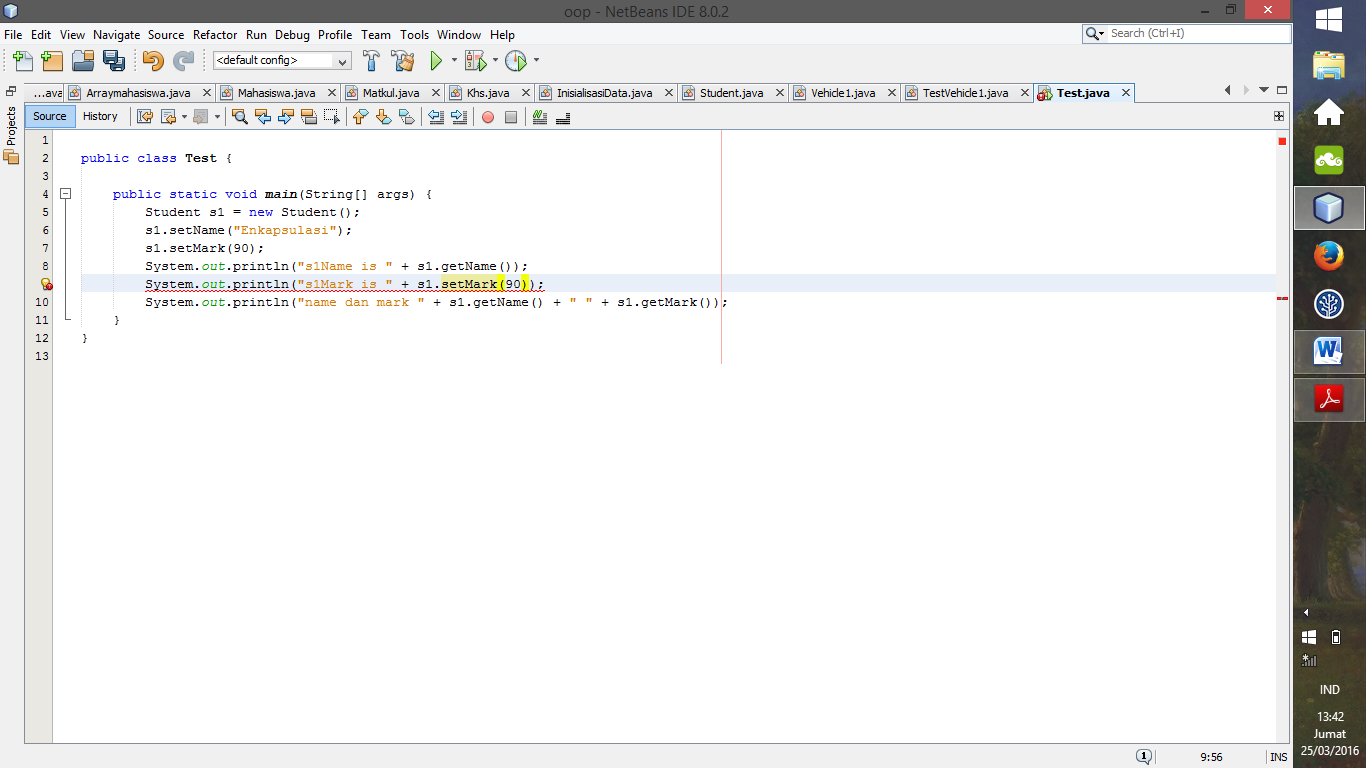
**Jawab :** Tidak error ,dikarenakan method get name pada class Student tidak memiliki parameter dan pemanggilan method tersebut pada class Test juga harus tidak berparameter.

3. Lakukan perubahan pada baris 5 dengan menghilangkan String di tanda dalam kurung lalu ubah baris 7 seperti no. 3 apa yang terjadi? jelaskan!

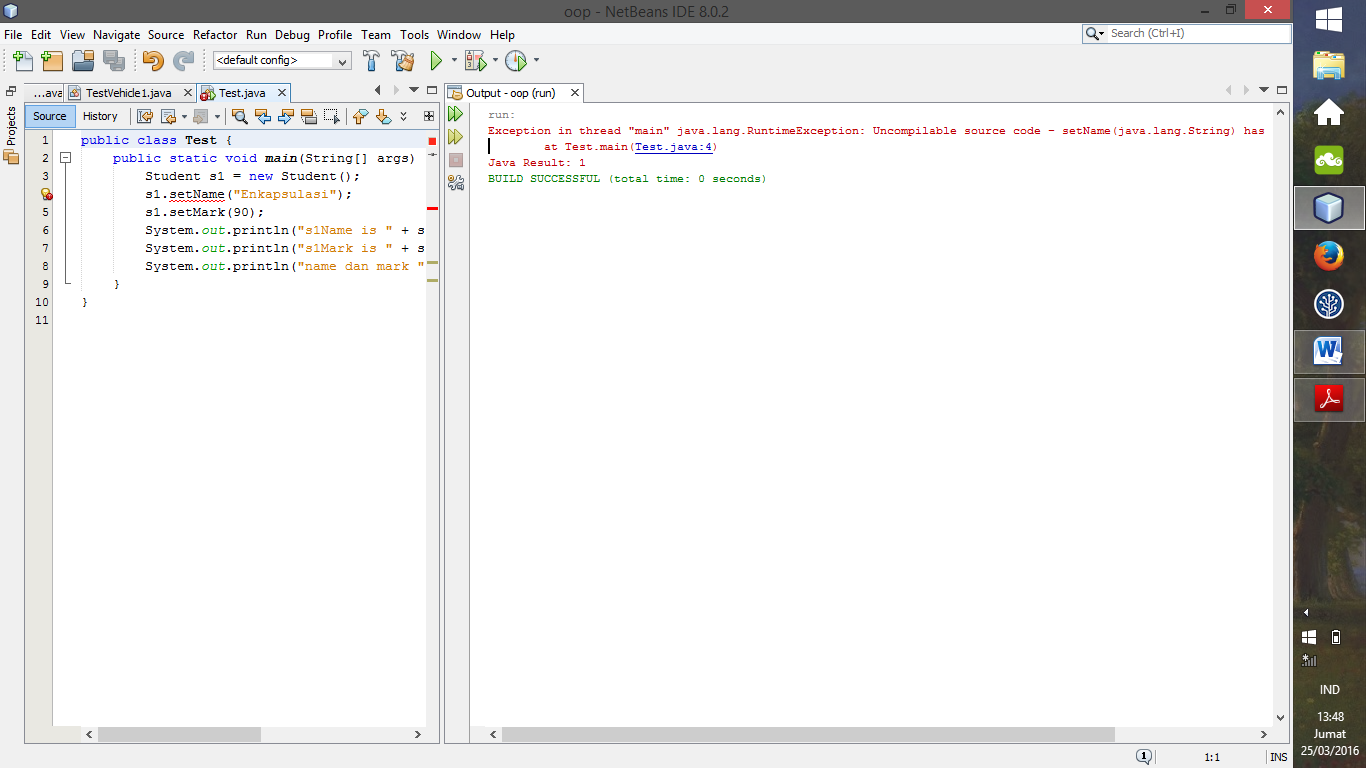
**Jawab :** -Tidak error, dikarenakan pada class Student, method setMark berparameter integer , jadi jika method itu dipanggil di class Test dengan parameter 90(integer) tidak menyebabkan error



-jika baris 7 diubah menjadi seperti no.3 akan terjadi error dikarenakan setMark adalah method void, dan method void tidak bisa dicetak.



4. Setelah diperbaiki, ubahlah hak akses pada baris 4 (pada class Student) menjadi *private* apa yang terjadi jika class Test dijalankan? Jelaskan!

**Jawab :** Terjadi error dikarenakan visibilitas private hanya bisa diakses oleh class itu sendiri, jadi class Test tidak bisa mengakses method setName

5. Jika kedua kelas diatas terdapat dalam package yang sama apakah konsep enkapsulasi tetap berfungsi? jelaskan!

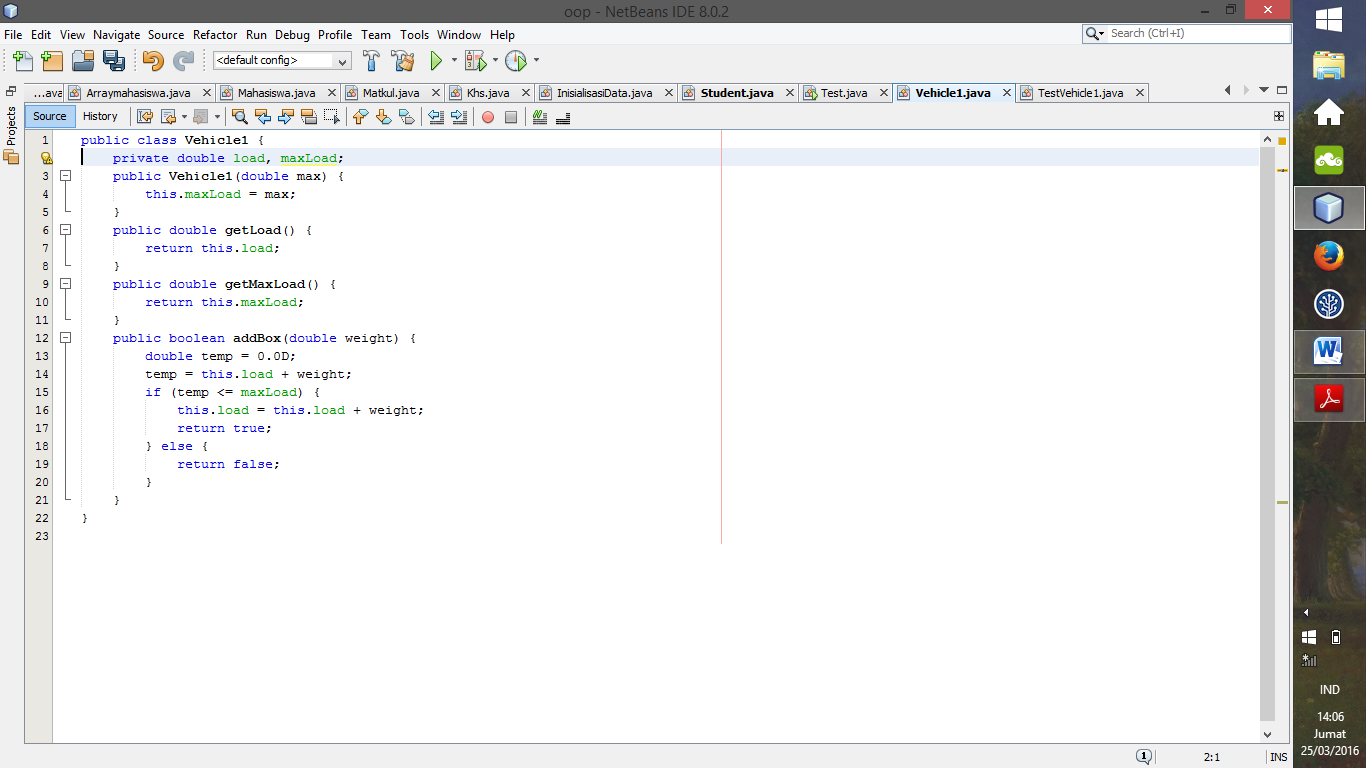
**Jawab :** ya, tetapi jika semua method pada class Student visibilitasnya diganti dengan private maka method tersebut tidak bisa diakses di class Test, dikarenakan hak akses private hanya bisa diakses di class itu sendiri(dalam hal ini konsep enkapsulasi berlaku tetapi tidak bisa diakses oleh class lain)

**B. Encapsulation 2**

Pertanyaan

1. Method apakah yang menjadi accessor (getter) ?

**Jawab :** Method yang memiliki return value(pada class Vehicle1)

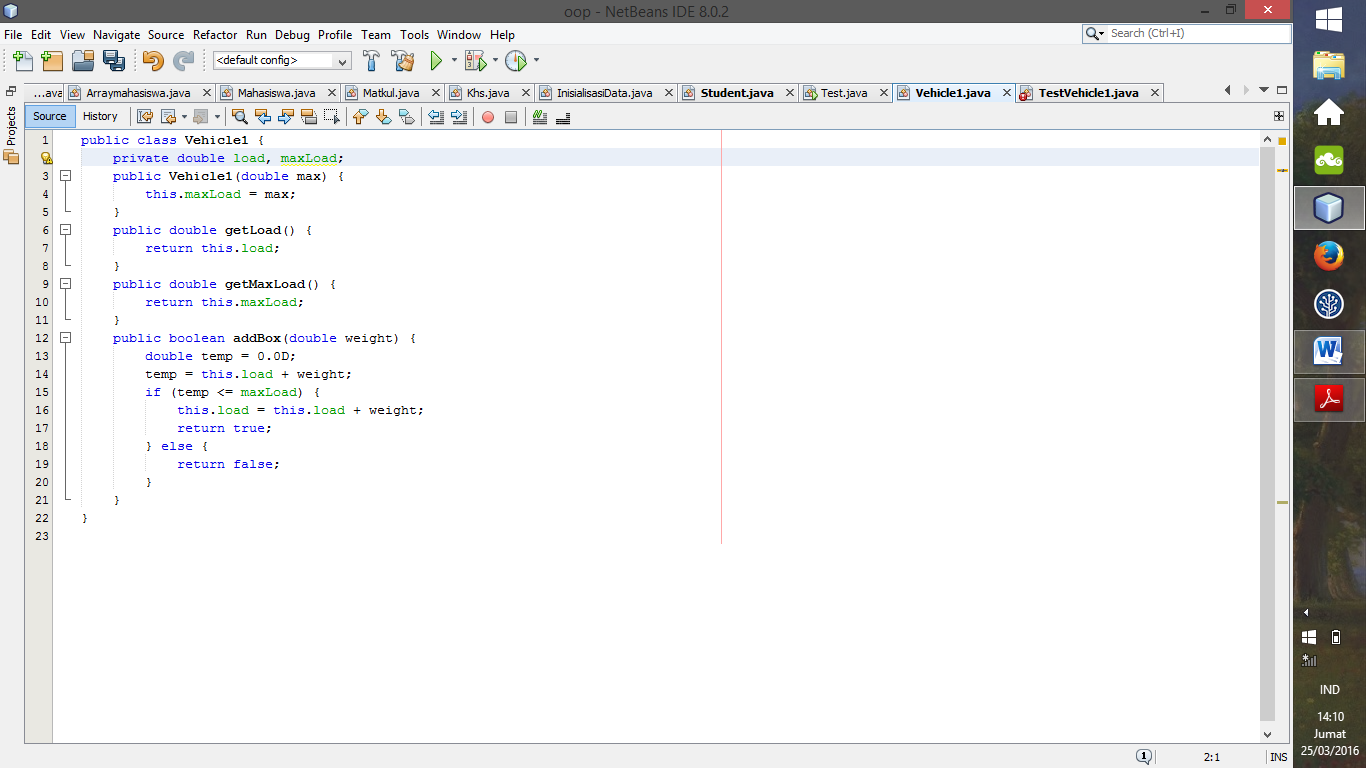


2. Tambahkan source code berikut dibawah baris ke 6 pada class TestVehicle1.

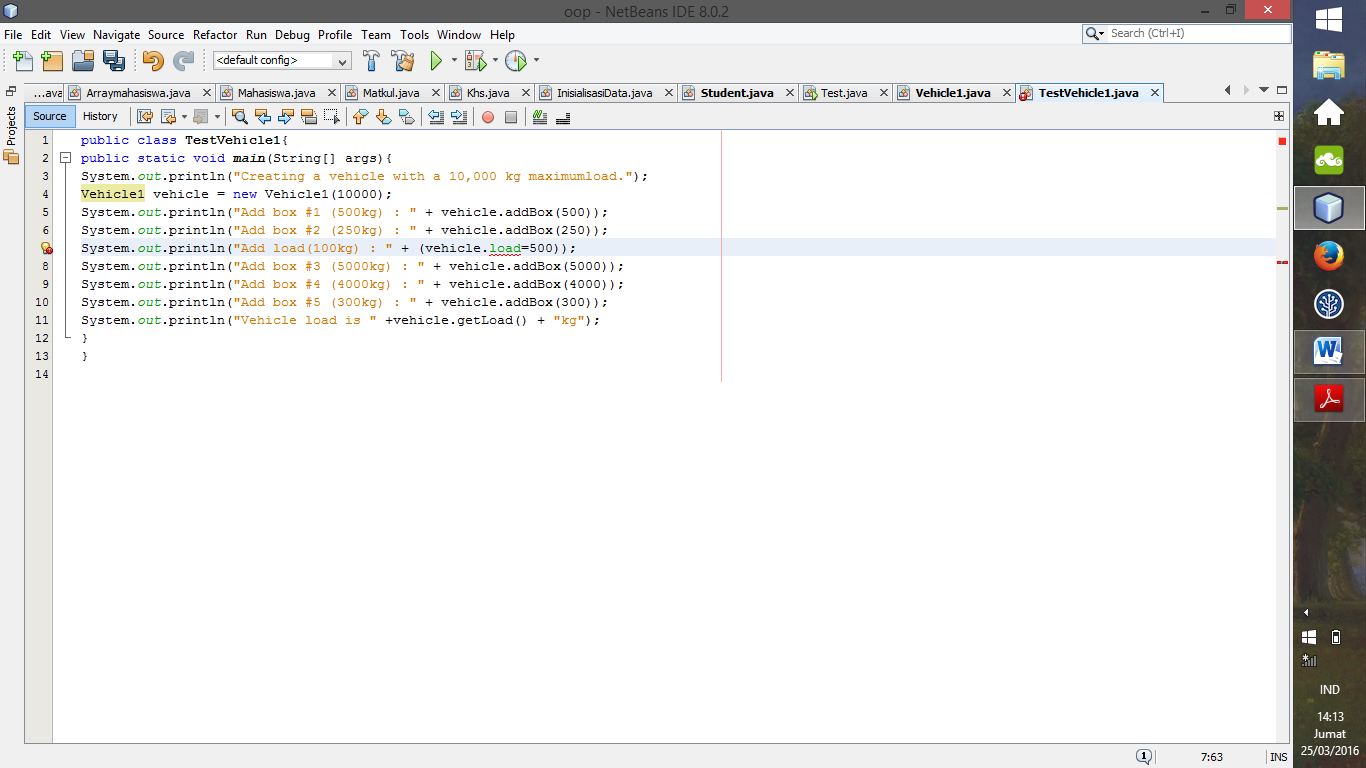
System.out.println("Add load(100kg) : " + (vehicle.load=500));

Jalankan program, apakah output dari program tersebut?

**Jawab :**clsass TestVehicle1 tidak bisa menginisialisasi variabel load dikarenakan pada class Vehicle1 variabel load bervisibilitas private



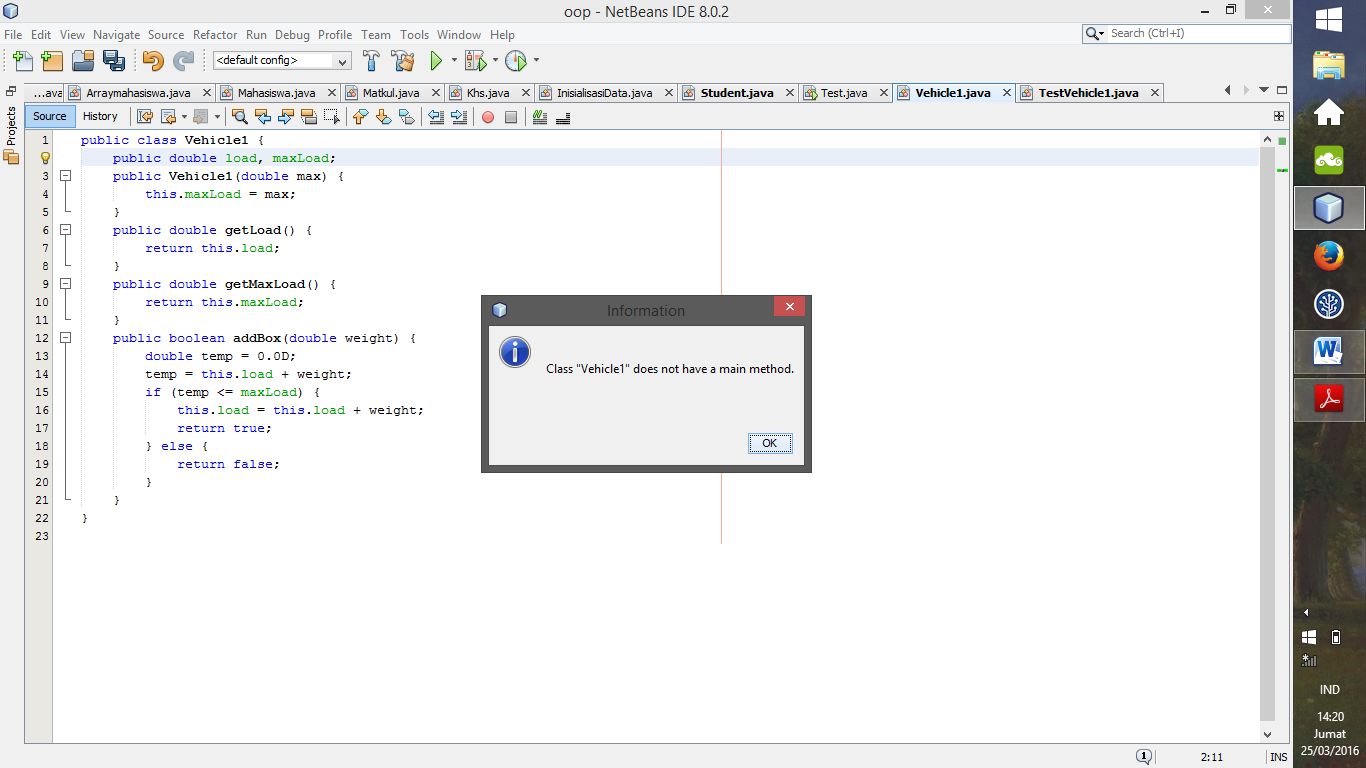
Screenshoot pada class Testvehicle1 :



Kembalikan program seperti semula.

3. Ubahlah tipe data pada atribut load dan maxload pada class Vehicle1 menjadi **public**.

Jalankan program, apakah output dari program tersebut?

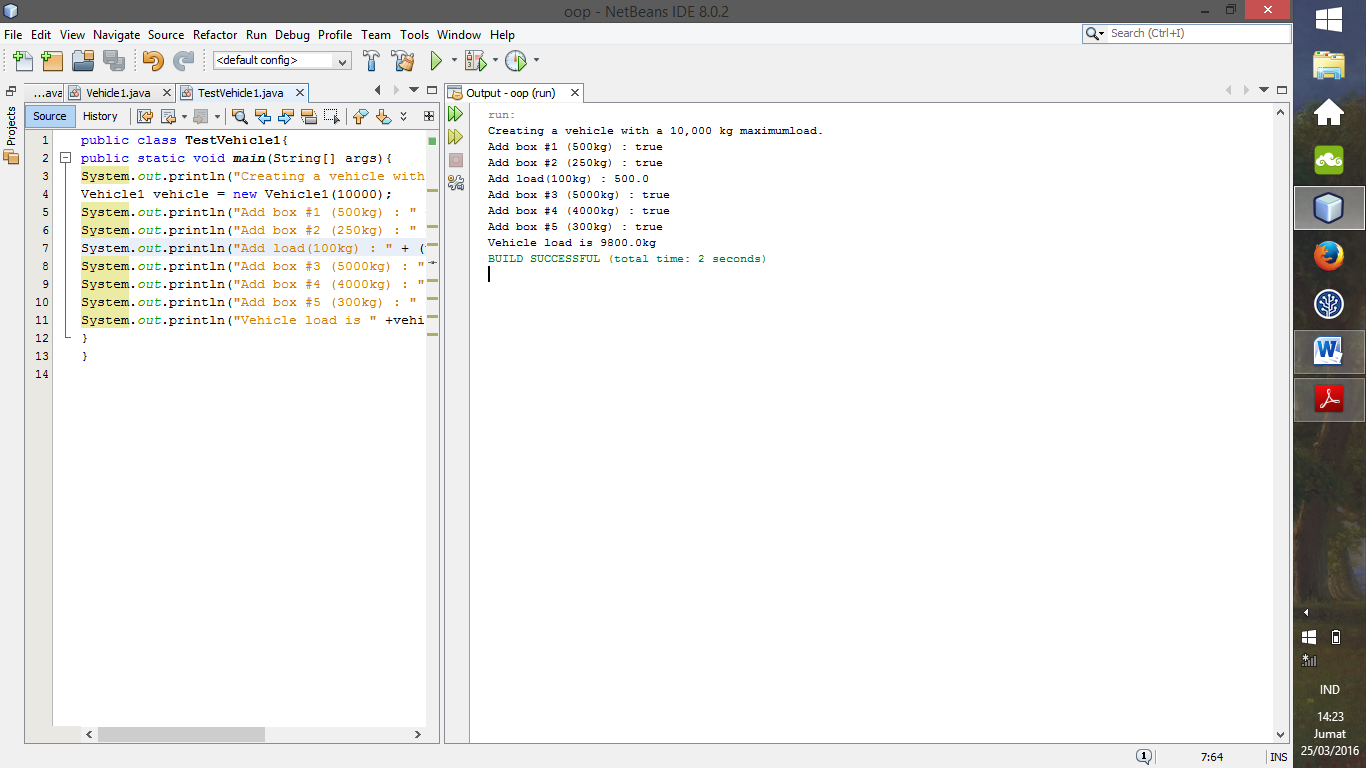
**Jawab :** Tidak bisa dijalankan karena class ini tidak memiliki main method

a. Tambahkan source kode berikut dibawah baris ke 6 pada class TestVehicle1.

System.out.println("Add load(100kg) : " + (vehicle.load=500));

Jalankan program, apakah output dari program tersebut?

**Jawab :**



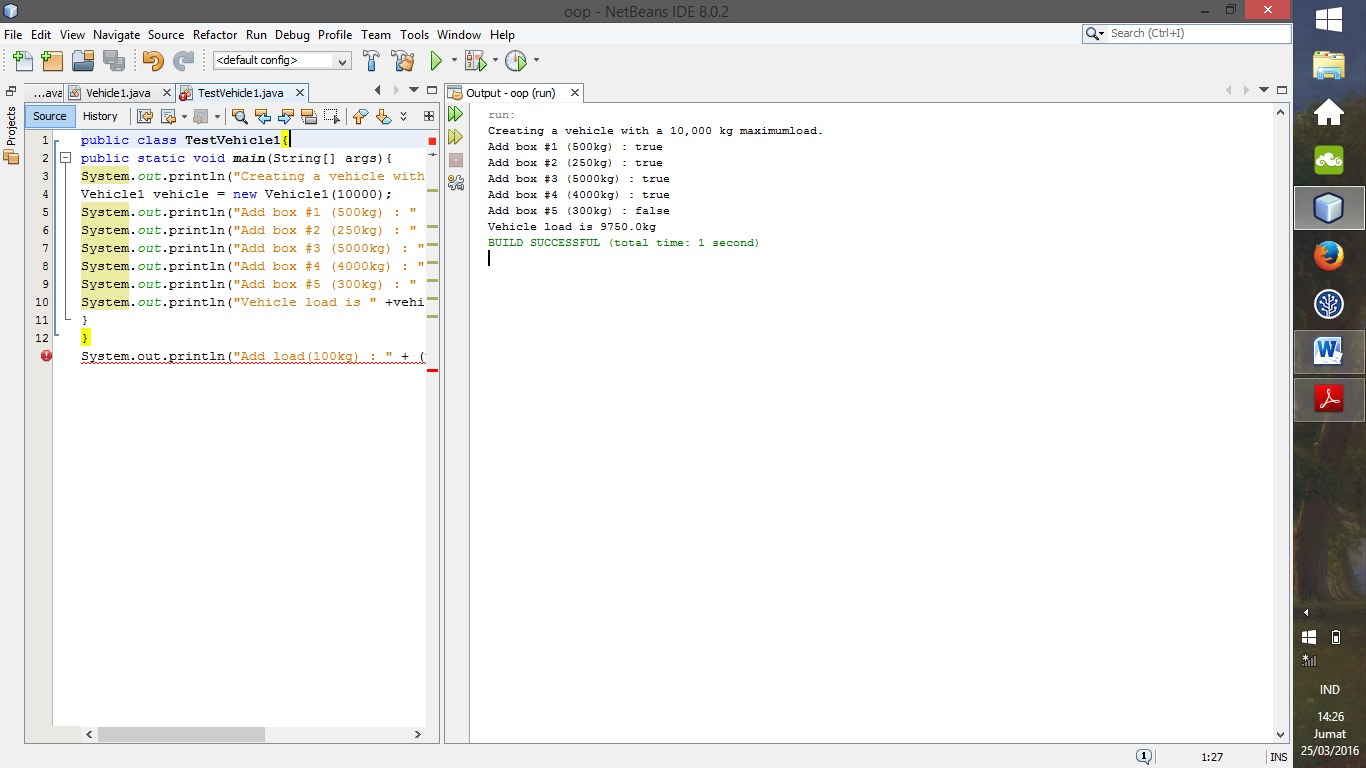
Kembalikan program seperti semula.

b. Tambahkan source kode berikut dibawah baris ke 12 pada class TestVehicle1.

System.out.println("Add load(100kg) : " + (vehicle.load=500));

Jalankan program, apakah output dari program tersebut?

**Jawab :**



Kembalikan program seperti semula.

4. Ulangi instruksi pada nomer 4 dengan mengubah tipe data pada atribut load dan maxload pada class Vehicle1 menjadi **protected.**

**Jawab :** Tetap bisa diakses dikarenakan protected hanya tidak bisa diakses oleh root/network selain itu bisa.

5. Ulangi instruksi pada nomer 4 dengan mengubah tipe data pada atribut load dan maxload pada class Vehicle1 menjadi **default.**

**Jawab :** Tetap bisa diakses dikarenakan default hanya tidak bisa diakses oleh sub class dan atau root/network, sedangkan class Testvehicle1 bukanlah turunan dari class Vehicle1 ,tetapi class Testvehicle1 telah menginstansiasi class Vehicle1, jadi tetap bisa diakses.

1. **KESIMPULAN**

Enkapsulasi adalah pembungkus, pembungkus disini dimaksudkan untuk menjaga suatu proses program agar tidak dapat diakses secara sembarangan atau di intervensi oleh program lain. Konsep enkapsulasi sangat penting dilakukan untuk menjaga kebutuhan program agar dapat diakses sewaktu-waktu, sekaligus menjaga program tersebut.

Dua hal yang mendasar dari enkapsulasi yakni :

1. Information Hiding

Sebelumnya, kita dapat mengakses anggota class baik berupa atribut maupun method secara langsung dengan menggunakan objek yang telah kita buat. Hal ini dikarenakan akses kontrol yang diberikan kepada atribut maupun method yang ada di dalam class tersebut adalah 'public'. Kita dapat menyembunyikan informasi dari suatu class sehingga anggota class tersebut tidak dapat diakses dari luar, caranya adalah hanya dengan memberikan akses kontrol 'private' ketika mendeklarasikan atribut atau method. Proses ini disebut dengan information hiding.

2. Interface to Access Data

Jika kita telah melakukan information hiding terhadap suatu atribut pada suatu class, lalu bagaimana melakukan perubahan terhadap atribut yang kita sembunyikan tersebut. 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. Enkapsulasi memiliki manfaat sebagai berikut:

- Modularitas ModulPraktikum – PemrogramanLanjut

Source code dari sebuah class dapat dikelola secara independen dari source code class yang lain. Perubahan internal pada sebuah class tidak akan berpengaruh bagi class yang menggunakannya.

- Information Hiding

Penyembunyian informasi yang tidak perlu diketahui objek lain.

Access modifier merupakan keyword yang digunakan untuk menentukan spesifikasi tingkat akses suatu vaiable atau method (anggota kelas). Acces modifier didalam java diantara lain yakni :

1. Default : tidak ada keyword untuk mendeklarasikan acces modifier ini, access modifier ini memiliki hak akses untuk class itu sendiri dan class dengan 1 package
2. Private : dengan menggunaka keyword private pada syntax tertentu maka syntax tersebut hanya bisa diakses oleh class itu sendiri
3. Protected : dengan menggunakan keyword protected pada syntax tertentu maka syntax tersebut hanya bisa diakses oleh class itu sendiri atau class dengan 1 package atau class turunanya
4. Public : dengan menggunkan keyword public pada syntax tertentu maka syntax tersebut bias diakses bebas