1. **DEFINISI MASALAH**

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

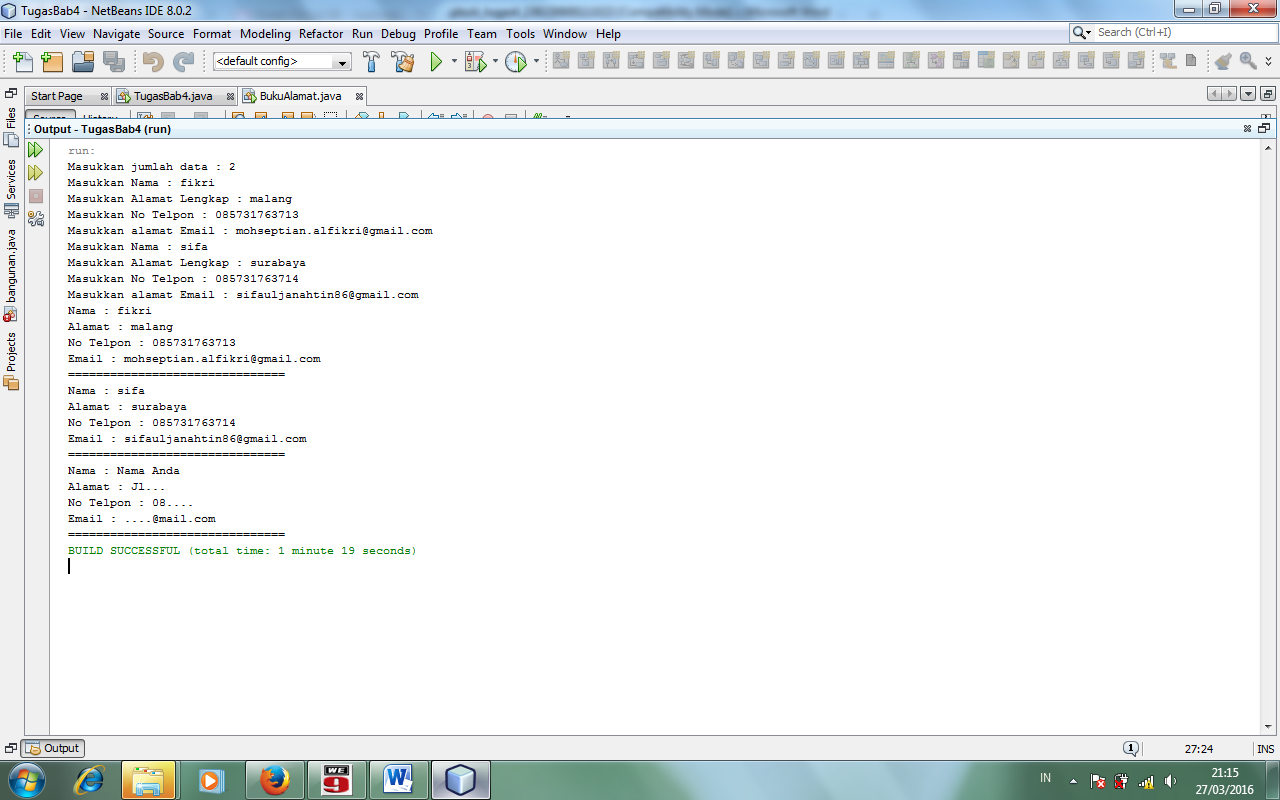
1. **SOURCE CODE**

|  |  |
| --- | --- |
|  | BukuAlamat.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 BukuAlamat {  private String Nama:  private String Alamat;  private String NomorTelp;  private String Email;  public BukuAlamat(){  this.Nama="Nama Anda";  this.Alamat="Jl...";  this.NomorTelp="08....";  this.Email="....@mail.com";}  public BukuAlamat(String nama, String alamat, String noTelp, String mail){  this.Nama=nama;  this.Alamat=alamat;  this.NomorTelp=noTelp;  this.Email=mail;}  public void setNama(String nama){  this.Nama=nama;}  public void setAlamat(String alamat){  this.Alamat=alamat;}  public void setNoTelp(String noTelp){  this.NomorTelp=noTelp;}  public void setMail(String mail){  this.Email=mail;}  public String getNama(){  return Nama;}  public String getAlamat(){  return alamat;}  public String getNoTelp(){  return NomorTelp; }  public String getMail(){  return Email;}  public void display(){  System.out.println("Nama : "+getNama());  System.out.println("Alamat : "+getAlamat());  System.out.println("No Telpon : "+getNoTelp());  System.out.println("Email : "+getMail());  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");}  public static void main(String[] args) {  Scanner in = new Scanner(System.in);  BukuAlamat[] ba1 = new BukuAlamat[100];  String[] nama = new String[100];  String[] alamat = new String[100];  String[] no = new String[100];  String[] mail = new String[100];  int ang=1;  System.out.print("Masukkan jumlah data : ");  int jum = in.nextInt();  do{  System.out.print("Masukkan Nama : ");  nama[ang] = in.next();  System.out.print("Masukkan Alamat Lengkap : ");  alamat[ang] = in.next();  System.out.print("Masukkan No Telpon : ");  no[ang] = in.next();  System.out.print("Masukkan alamat Email : ");  mail[ang] = in.next();  ba1[ang]=new BukuAlamat(nama[ang], alamat[ang], no[ang], mail[ang]);  ang++;  }while(ang<=jum);  BukuAlamat ba2 = new BukuAlamat();  for (int i = 1; i <=jum; i++) {  ba1[i].display(); }  ba2.display();}} |

1. **PEMBAHASAN**

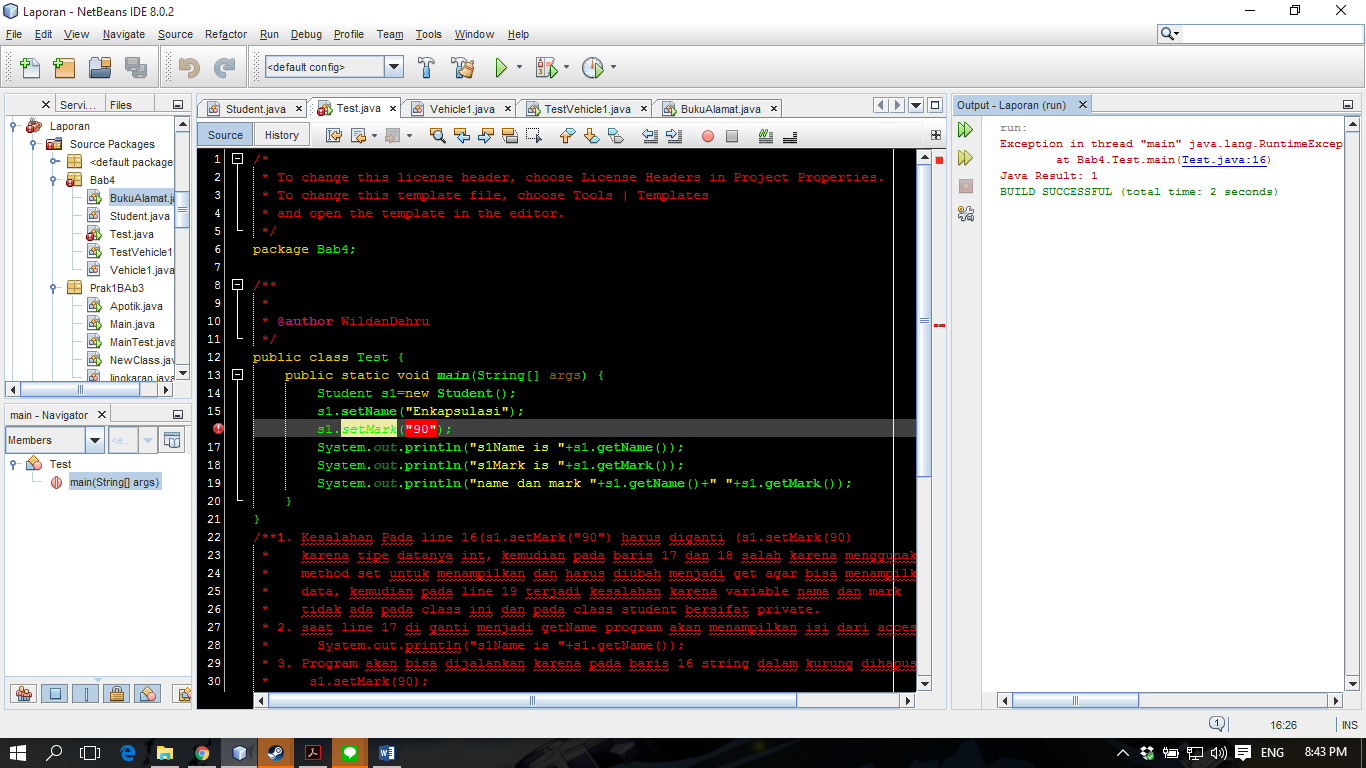
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62 | Mendeklarasikan import java.util.Scanner  Mendeklarasikan class dengan nama BukuAlamat  Mendeklarasikan instance variabel nama, alamat, noTelp, main dengan tipe data String bersifat private  Mendeklarasikan konstruktor default  Mendeklarasikan variabel instance nama =”Nama Anda”  Mendeklarasikan variabel instance alamat = “Jl….”  Mendeklarasikan variabel instance noTelp = “08….”  Mendeklarasikan variabel instance mail = “….@mail.com”  Mendeklarasikan konstruktor inisialisasi dengan parameter nama, alamat, noTelp, mail bertipe data String  Mendeklarasikan variabel instance nama = variabel local nama  Mendeklarasikan variabel instance alamat = variabel local alamat  Mendeklarasikan variabel instance noTelp = variabel local noTelp  variabel instance mail = variabel local mail  method setNama (mutator) dengan parameter nama bertipedata String  variabel instance nama=variabel local nama  method setAlamat (mutator) dengan parameter alamat bertipedata String  variabel instance alamat=variabel local alamat  method setNoTelp (mutator) dengan parameter noTelp bertipedata String Deklarasi variabel instance noTelp=variabel local noTelp  method setMail (mutator) dengan parameter mail bertipedata String  variabel instance nmail=variabel local mail  method getNama (accessor)  return value dari variabel nama  method getAlamat (accessor)  return value dari variabel alamat  method getNoTelp (accessor)  return value dari variabel noTelp  method getMail (accessor)  return value dari variabel mail  method void display  Mencetak kalimat “Nama : “ dengan nilai dari method getName()  Mencetak kalimat “Alamat : “ dengan nilai dari method getAlamat()  Mencetak kalimat “No Telpon : “ dengan nilai dari method getNoTelp()  Mencetak kalimat“Mail : “ dengan nilai dari method getMail()  Mencetak output “\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*”  method static void String  input dengan variabel in  objek array dari class BukuAlamat  array nama dengan tipedata String  array alamat dengan tipedata String  array no dengan tipedata String  array mail dengan tipedata String  variabel ang dengan tipedata int  Menampilkan output tulisan “Masukkan jumlah data : “  variabel jum dengan tipedata int sebagai input Scanner in  awal perulangan do while  Menampilkan output tulisan “Masukkan Nama : ”  variabel array nama sebagai input Scanner in  Menampilkan output tulisan “Masukkan Alamat Lengkap : ”  Deklarasi variabel array alamat sebagai input Scanner in  Menampilkan output tulisan “Masukkan No Telpon : ”  variabel array no sebagai input Scanner in  Menampilkan output tulisan “Masukkan alamat Email : ”  Deklarasi variabel mail nama sebagai input Scanner in  konstruktor dari objek ba1[ang] dengan parameter nama[ang], alamat[ang], no[ang],mail[ang]  increment ang++  penutup perulang do while dengan syarat ang<=jum  objek ba2 dengan konstruktor default  perulangan for dimulai dari i=1 sampai nilai jum  objek ba1[i] dengan method display()  objek ba2 dengan method display() |

1. **SCREENSHOT PROGRAM**



1. **PRAKTIKUM**
2. **Encapsulation 1**
   1. Lakukan percobaan diatas dan benahi jika menemukan kesalahan!

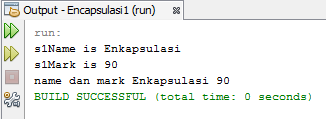
**Jawab :**



Terjadi kesalahan.

Yang benar adalah

|  |
| --- |
| S1.setName(“Enkapsulasi”);  S1.setMark(90); |



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

**Jawab :**

System.out.println(“s1Namw is “+s1.getName());

Terjadi pemanggilan method getName yang mengembalikan nilai name.

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

**Jawab :**

s1.setMark(90);

Program bisa dijalankan karena pada baris 7 sebelumnya terjadi error

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

**Jawab :**

|  |
| --- |
| Class |
| private void setName(String n){  name=n; |

|  |
| --- |
| MainClass |
| public statis void main (String[] args){  Student s1=new Student();  s1.setName(“Enkapsulasi”);  s1.setMark(90);  System.out.println(“s1Name is “+s1.getName());  System.out.println(“s1Mark is “+s1.getMark());  System.out.println(“name dan mark “+s1.getName()+” “+s1.getMark());  } |

Terjadi kesalahan karena hak akses setName adalah private sehingga saat main class (class Test) ingin mengakses ditolak.

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

**Jawab :**

Ya, konsep enkapsulasi tetap berfungsi. Hal ini disebabkan oleh hakakses yang ada pada tiap variabel atau method. Dan ketika menggunakan accesor serta mutator pada class tersebut

1. **Encapsulation 2**
2. Method apakah yang menjadi accessor (getter) ?

**Jawab :**

public double getLoad(){

return this.load;

}

public double getMaxLoad(){

return this.maxLoad;

}

1. 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?

Kembalikan program seperti semula.

**Jawab :**

**Main Class**

public class TestVehicle1 {

public static void main (String[] args){

System.out.println(“Creating a vehicle with a 10,000 kg maxium load.”);

Vehicle1 vehicle = new Vehicle1(10000);

System.out.println(“Add box #1 (500kg) : “ +vehicle.addBox(500));

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

System.out.println(“Add box #2 (250kg) : “ +vehicle.addBox(250));

System.out.println(“Add box #3 (5000kg) : “ +vehicle.addBox(5000));

System.out.println(“Add box #4 (4000kg) : “ +vehicle.addBox(4000));

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

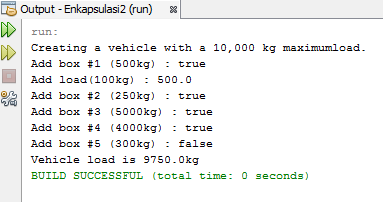
System.out.println(“Add box #5 (300kg) : “ + vehcle.addBox(300));

System.out.println”Vehicle load is “ +vehicle.getLoad() + “kg”);

}

}

**Output**



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

Jalankan program, apakah output dari program tersebut?

* 1. 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?

Kembalikan program seperti semula.

* 1. 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?

Kembalikan program seperti semula.

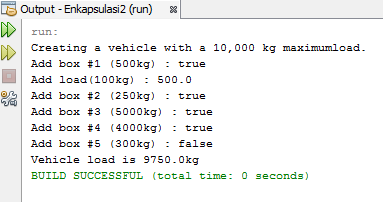
**Jawab :**

|  |
| --- |
| Class Vehicle1 |
| public double load, maxLoad;  public Vehicle1 (double max){  this.maxLoad = max; |

a.

|  |
| --- |
| MainClass |
| public class TestVehicle1 {  public static void main (String[] args){  System.out.println(“Creating a vehicle with a 10,000 kg maxium load.”);  Vehicle1 vehicle = new Vehicle1(10000);  System.out.println(“Add box #1 (500kg) : “ +vehicle.addBox(500));  System.out.println(“Add load (100kg) : “ +vehicle.load=500);  System.out.println(“Add box #2 (250kg) : “ +vehicle.addBox(250));  System.out.println(“Add box #3 (5000kg) : “ +vehicle.addBox(5000));  System.out.println(“Add box #4 (4000kg) : “ +vehicle.addBox(4000));  // System.out.println(“Add load(100kg) : “ + (vehicle.load=500));  System.out.println(“Add box #5 (300kg) : “ + vehcle.addBox(300));  System.out.println”Vehicle load is “ +vehicle.getLoad() + “kg”);  }  } |

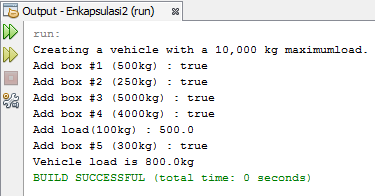
Output



b. Main Class

|  |
| --- |
| MainClass |
| public class TestVehicle1 {  public static void main (String[] args){  System.out.println(“Creating a vehicle with a 10,000 kg maxium load.”);  Vehicle1 vehicle = new Vehicle1(10000);  System.out.println(“Add box #1 (500kg) : “ +vehicle.addBox(500));  System.out.println(“Add load (100kg) : “ +vehicle.load=500);  System.out.println(“Add box #2 (250kg) : “ +vehicle.addBox(250));  System.out.println(“Add box #3 (5000kg) : “ +vehicle.addBox(5000));  System.out.println(“Add box #4 (4000kg) : “ +vehicle.addBox(4000));  // System.out.println(“Add load(100kg) : “ + (vehicle.load=500));  System.out.println(“Add box #5 (300kg) : “ + vehcle.addBox(300));  System.out.println”Vehicle load is “ +vehicle.getLoad() + “kg”);  }  } |

Output

` 

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

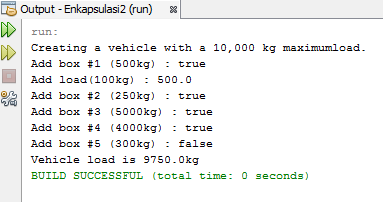
**Jawab :**

|  |
| --- |
| Class Vehicle1 |
| protected double load, maxLoad;  public Vehicle1 (double max)  { |

a.

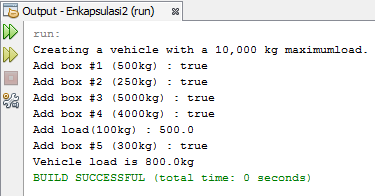
|  |
| --- |
| MainClass |
| public class TestVehicle1 {  public static void main (String[] args){  System.out.println(“Creating a vehicle with a 10,000 kg maxium load.”);  Vehicle1 vehicle = new Vehicle1(10000);  System.out.println(“Add box #1 (500kg) : “ +vehicle.addBox(500));  System.out.println(“Add load (100kg) : “ +vehicle.load=500);  System.out.println(“Add box #2 (250kg) : “ +vehicle.addBox(250));  System.out.println(“Add box #3 (5000kg) : “ +vehicle.addBox(5000));  System.out.println(“Add box #4 (4000kg) : “ +vehicle.addBox(4000));  // System.out.println(“Add load(100kg) : “ + (vehicle.load=500));  System.out.println(“Add box #5 (300kg) : “ + vehcle.addBox(300));  System.out.println”Vehicle load is “ +vehicle.getLoad() + “kg”);  }  } |

Output



|  |
| --- |
| MainClass |
| public class TestVehicle1 {  public static void main (String[] args){  System.out.println(“Creating a vehicle with a 10,000 kg maxium load.”);  Vehicle1 vehicle = new Vehicle1(10000);  System.out.println(“Add box #1 (500kg) : “ +vehicle.addBox(500));  System.out.println(“Add load (100kg) : “ +vehicle.load=500);  // System.out.println(“Add load(100kg) : “ + (vehicle.load=500));  System.out.println(“Add box #2 (250kg) : “ +vehicle.addBox(250));  System.out.println(“Add box #3 (5000kg) : “ +vehicle.addBox(5000));  System.out.println(“Add box #4 (4000kg) : “ +vehicle.addBox(4000));  System.out.println(“Add box #5 (300kg) : “ + vehcle.addBox(300));  System.out.println”Vehicle load is “ +vehicle.getLoad() + “kg”);  }  } |

Output



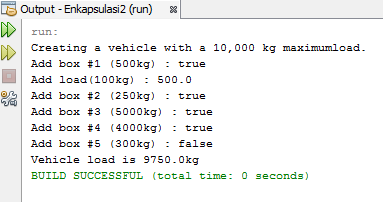
1. Ulangi instruksi pada nomer 4 dengan mengubah tipe data pada atribut load dan maxload pada class Vehicle1 menjadi **default.**

**Jawab :**

|  |
| --- |
| Class Vehicle1 |
| double load, maxLoad;  public Vehicle1 (double max)  { |

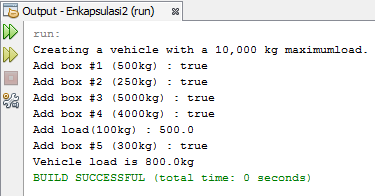
|  |
| --- |
| MainClass |
| public class TestVehicle1 {  public static void main (String[] args){  System.out.println(“Creating a vehicle with a 10,000 kg maxium load.”);  Vehicle1 vehicle = new Vehicle1(10000);  System.out.println(“Add box #1 (500kg) : “ +vehicle.addBox(500));  System.out.println(“Add load (100kg) : “ +vehicle.load=500);  System.out.println(“Add box #2 (250kg) : “ +vehicle.addBox(250));  System.out.println(“Add box #3 (5000kg) : “ +vehicle.addBox(5000));  System.out.println(“Add box #4 (4000kg) : “ +vehicle.addBox(4000));  // System.out.println(“Add load(100kg) : “ + (vehicle.load=500));  System.out.println(“Add box #5 (300kg) : “ + vehcle.addBox(300));  System.out.println”Vehicle load is “ +vehicle.getLoad() + “kg”);  }  } |

Output



|  |
| --- |
| MainClass |
| public class TestVehicle1 {  public static void main (String[] args){  System.out.println(“Creating a vehicle with a 10,000 kg maxium load.”);  Vehicle1 vehicle = new Vehicle1(10000);  System.out.println(“Add box #1 (500kg) : “ +vehicle.addBox(500));  // System.out.println(“Add load(100kg) : “ + (vehicle.load=500));  System.out.println(“Add load (100kg) : “ +vehicle.load=500);  System.out.println(“Add box #2 (250kg) : “ +vehicle.addBox(250));  System.out.println(“Add box #3 (5000kg) : “ +vehicle.addBox(5000));  System.out.println(“Add box #4 (4000kg) : “ +vehicle.addBox(4000));  System.out.println(“Add box #5 (300kg) : “ + vehcle.addBox(300));  System.out.println”Vehicle load is “ +vehicle.getLoad() + “kg”);  }  } |

Output



1. **KESIMPULAN**
2. Enkapsulasi adalah suatu cara untuk menyembunyikan informasi detail dari suatu class. Dalam enkapsulasi terdapat hak akses *public, protected, dan private*.
3. Access modifier merupakan keyword yang digunakan untuk menentukan spesifikasi tingkat akses suatu variable atau method (anggota kelas). Peng-aksesan yang dimaksud bisa berupa pengaksesan dalam kelas yang sama, turunan maupun dari luar kelas di mana variable dan method dideklarasikan.
4. - default

Access modifier ini hanya menspesifikasikan kelas-kelas (classes) dipaket yang sama yang bisa mengakses variabel dan method suatu kelas. Anggota kelas dengan default access bisa melihat kelas lain dipaket yang sama. Tidak ada keyword khusus untuk mendeklarasian modifier default access. Sehingga apabila tidak ada access modifier pada pendeklarasian tersebut, berarti yang dimaksud adalah default access.

* Public

Suatu anggota kelas (class member) dideklarasikan sebagai public, maka anggota tersebut akan bisa digunakan oleh siapa saja, baik oleh kelas yang sama kelas lain maupun lingkungan luar kelas. Sehingga anggota kelas (class member) yang dideklarasikan sebagai publicakan bisa diakses oleh object lain.

* Protected

Access modifier protected menspesifikasikan atau menentukan anggota kelas (class member) yang hanya bisa diakses oleh method-method yang ada didalam kelas yang sama dan juga dapat di akses ole subclass (kelas turunan) dari kelas tersebut

* Private

Access modifier merupakan tingkatan akses yang sangat . kemudian seperti modifier modifier lain, modifier private menspesifikasikan anggota kelas (class member) yang hanya bisa diakses oleh kelas dimana anggota kelas tersebut dideklarasikan. Keadaan ini mengakibatkan bahwa tidak ada kelas lain yang bisa mengakses anggota kelas private, sekaligus termasuk subkelas-subkelas nya. Acces modifier private adalah yang paling terbatas, menspesifikasikan anggota kelas (variable dan method) hanya dapat diakses oleh kelas dimana anggota itu didefinisikan.