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**

|  |  |
| --- | --- |
| BukuTelepon.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 | package bab4;  public class BukuTelepon {  private String Nama, Alamat, NoTelp, Email;  public BukuTelepon() {  }  public BukuTelepon(String Nama, String Alamat, String NoTelp, String Email) {  this.Nama = Nama;  this.Alamat = Alamat;  this.NoTelp = NoTelp;  this.Email = Email;  }  public BukuTelepon(String NoTelp, String Email) {  this.NoTelp = NoTelp;  this.Email = Email;  }  public void setNama(String Nama) {  this.Nama = Nama;  }  public String getNama() {  return Nama;  }  public void setAlamat(String Alamat) {  this.Alamat = Alamat;  }  public String getAlamat() {  return Alamat;  }  public void setNoTelp(String NoTelp) {  this.NoTelp = NoTelp;  }  public String getNoTelp() {  return NoTelp;  }  public void setEmail(String Email) {  this.Email = Email;  }  public String getEmail() {  return Email;  }  } |

|  |  |
| --- | --- |
| BukuTeleponMain.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  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87 | package bab4;  import java.util.ArrayList;  import java.util.List;  import java.util.Scanner;  public class BukuTeleponMain {  public static void main(String[] args) {  Scanner scn = new Scanner(System.in);  List<BukuTelepon> butel = new ArrayList<BukuTelepon>();  butel.add(new BukuTelepon("Angela", "Jalan Apel", "082265439876", "angela@gmail.com"));  butel.add(new BukuTelepon("Rama", "Jalan Anggur", "082136333348", "rama@gmail.com"));  while (true) {  System.out.print("MENU\n1.Show Contact\n2.Add Contact\n3.Edit Contact\n4.Delete Contact\n0.Exit\nPilihan:");  int pil = scn.nextInt();  scn.nextLine();  switch (pil) {  case 1:  showContact(butel);  break;  case 2:  BukuTelepon bt = new BukuTelepon();  System.out.print("Masukkan Nama : ");  bt.setNama(scn.nextLine());  System.out.print("Masukkan Alamat : ");  bt.setAlamat(scn.nextLine());  System.out.print("Masukkan No Telp : ");  bt.setNoTelp(scn.nextLine());  System.out.print("Masukkan email : ");  bt.setEmail(scn.nextLine());  butel.add(bt);  break;  case 3:  System.out.print("Masukkan nomor kontak : ");  int index = scn.nextInt();  BukuTelepon bts = butel.get(index - 1);  System.out.print("Anda yakin ingin mengubah kontak : " + bts.getNama() + " ? (Y/N) : ");  String pils = scn.next();  if (pils.equalsIgnoreCase("y")) {  scn.nextLine();  System.out.print("Masukkan nama : ");  bts.setNama(scn.nextLine());  System.out.print("Masukkan alamat : ");  bts.setAlamat(scn.nextLine());  System.out.print("Masukkan No Telp : ");  bts.setNoTelp(scn.nextLine());  System.out.print("Masukkan email : ");  bts.setEmail(scn.nextLine());  butel.set(index - 1, bts);  } else {  break;  }  break;  case 4:  System.out.print("Masukkan nomor kontak : ");  int indexs = scn.nextInt();  BukuTelepon btx = butel.get(indexs-1);  System.out.print("Anda yakin ingin menghapus kontak : " + btx.getNama() + " ? (Y/N) : ");  String pilss = scn.next();  if (pilss.equalsIgnoreCase("y")) {  butel.remove(indexs-1);  } else {  break;  }  break;  case 0:  System.exit(0);  break;  }  }  }  static void showContact(List<BukuTelepon> butel) {  for (int i = 0; i < butel.size(); i++) {  BukuTelepon bt = butel.get(i);  System.out.println("===== CONTACT NO " + (i + 1) + " =====");  System.out.println("Nama \t\t: " + bt.getNama());  System.out.println("Alamat \t\t: " + bt.getAlamat());  System.out.println("No Telp \t: " + bt.getNoTelp());  System.out.println("Email \t\t: " + bt.getEmail());  System.out.println("===== CONTACT NO " + (i + 1) + " =====");  System.out.println("");  }  }  } |

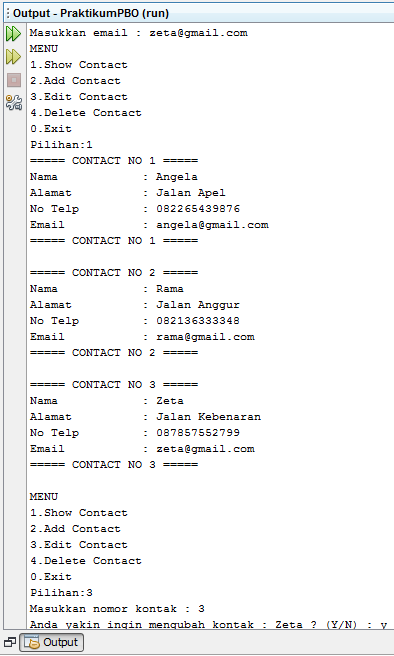
1. **PEMBAHASAN**

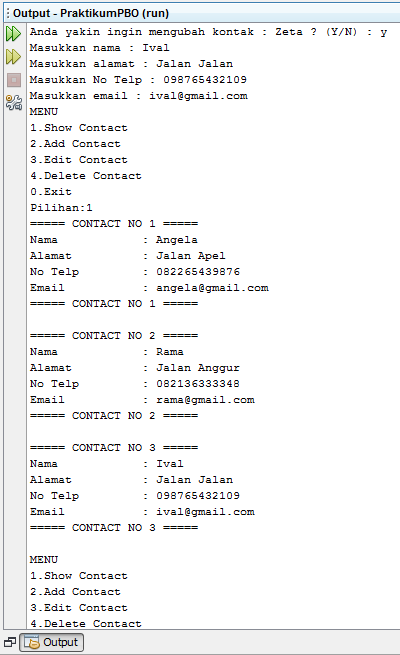
|  |  |
| --- | --- |
| BukuTelepon.java | |
| 1  2  3  4  6-7  8  9  10  11  13  14  15  17  18  20  21  23  24  26  27  29  30  32  33  35  36  38  39 | Deklarasi package bab4  Deklarasi nama clas BukuTelepon  Deklarasi variabel Nama, Alamat, NoTelp, Email bertipe String dan memiliki hak akses private  Constructor kosong  Constructor dengan parameter  Inisialisasi variabel Nama bernilai variabel Nama pada parameter  Inisialisasi variabel Alamat bernilai variabel Alamat pada parameter  Inisialisasi variabel NoTelp bernilai variabel NoTelp pada parameter  Inisialisasi variabel Email bernilai variabel Email pada parameter  Constructor dengan parameter  Inisialisasi variabel NoTelp bernilai variabel NoTelp pada parameter  Inisialisasi variabel Email bernilai variabel Email pada parameter  Mutator setNama  Inisialisasi variabel Nama bernilai variabel Nama pada parameter  Accessor getNama  Mengembalikan nilai variabel Nama  Mutator setAlamat  Inisialisasi variabel Alamat bernilai variabel Alamat pada parameter  Accessor getAlamat  Mengembalikan nilai variabel Alamat  Mutator setNoTelp  Inisialisasi variabel NoTelp bernilai variabel NoTelp pada parameter  Accessor getNoTelp  Mengembalikan nilai variabel NoTelp  Mutator setEmail  Inisialisasi variabel Email bernilai variabel Email pada parameter  Accessor getEmail  Mengembalikan nilai variabel Email |

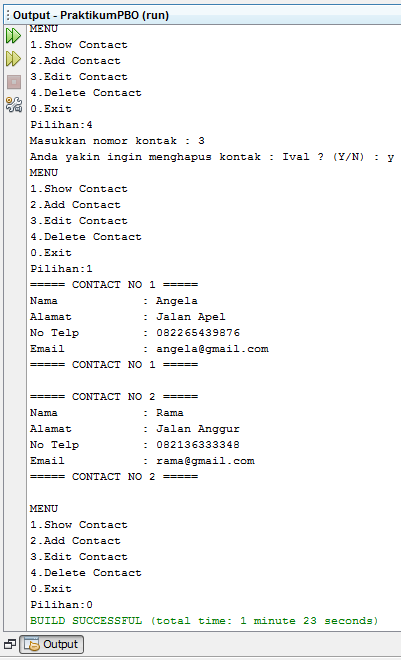
|  |  |
| --- | --- |
|  | |
|  |  |

1. **SCREENSHOT PROGRAM**







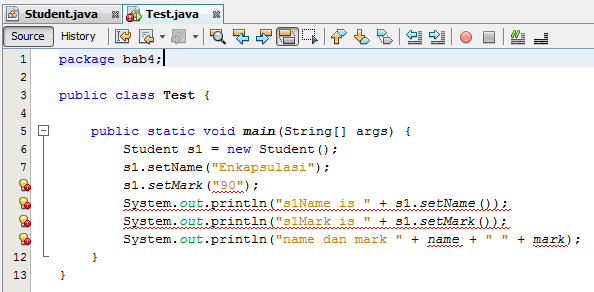


1. **PRAKTIKUM**
2. **Encapsulation 1**

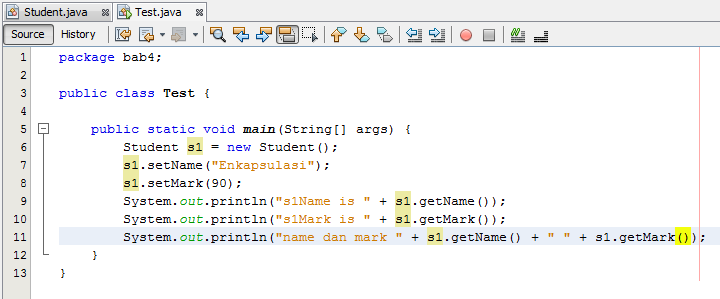
Pertanyaan

1. Lakukan percobaan diatas dan benahi jika menemukan kesalahan!

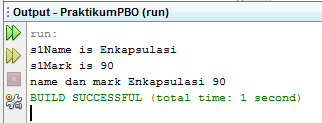
Jawab: Error pada class Test line ke 8,9,10,11.



Setelah dibenahi :



Outputnya :



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

Jawab: Baris 6 akan mencetak nilai dari variabel Name yang sebelumnya sudah di set oleh method setName

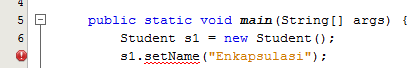
1. Lakukan perubahan pada baris 5 dengan menghilangkan String di tanda dalam kurung lalu ubah baris 7 seperti no. 3 apa yang terjadi? jelaskan!

Jawab: Baris 5 sudah diperbaiki pada nomor 1 dengan menghilangkan String di dalam kurung, dan hasilnya tidak ada error.

Jika baris 7 dengan code System.out.println("s1Mark is "+s1.setMark()); diubah seperti no 3 dengan code Student s1=new Student(); akan menjadi error karena terjadi 2 kali instansiasi dengan nama objek yang sama.

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

Jawab: Setelah diubah, akan menjadi error



Karena hak akses diubah menjadi private yang tidak bisa diakses secara langsung oleh class lain.

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

Jawab: Tetap berfungsi, karena walaupun di package yang sama namun jika suatu class menggunakan hak akses private, maka atribut/method tersebut hanya bisa diakses dalam classnya sendiri.

1. **Encapsulation 2**

Pertanyaan

1. Method apakah yang menjadi accessor (getter) ?

Jawab: Method return value.

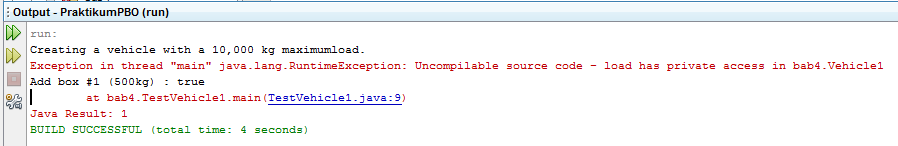
1. Tambahkan source code berikut dibawah baris ke 6 pada class TestVehicle1.

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

Jalankan program, apakah output dari program tersebut?

Kembalikan program seperti semula.

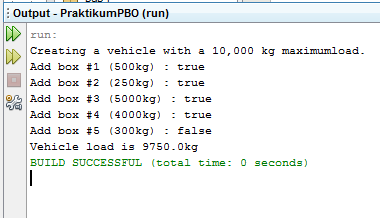
Jawab:



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

Jalankan program, apakah output dari program tersebut?

Jawab:



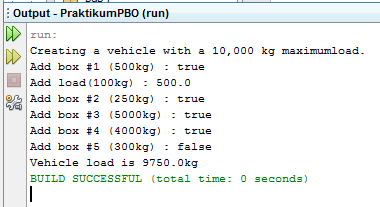
1. Tambahkan source kode berikut dibawah baris ke 6 pada class TestVehicle1.

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

Jalankan program, apakah output dari program tersebut?

Kembalikan program seperti semula.

Jawab:



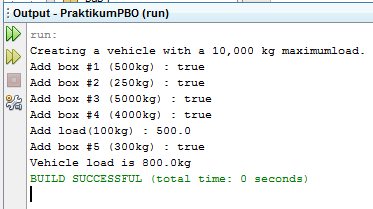
1. Tambahkan source kode berikut dibawah baris ke 12 pada class TestVehicle1.

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

Jalankan program, apakah output dari program tersebut?

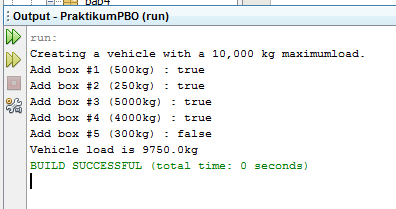
Kembalikan program seperti semula.

Jawab:



1. Ulangi instruksi pada nomer 3 dengan mengubah tipe data pada atribut load dan maxload pada class Vehicle1 menjadi protected.

Jawab: Saat program dijalankan maka outputnya adalah



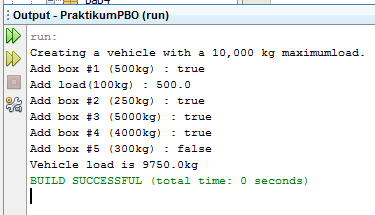
1. Tambahkan source kode berikut dibawah baris ke 6 pada class TestVehicle1.

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

Jalankan program, apakah output dari program tersebut?

Kembalikan program seperti semula.

Jawab:



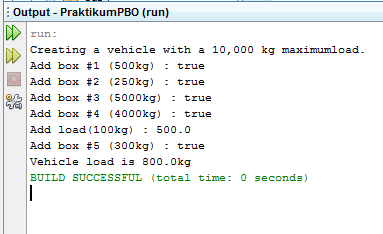
1. Tambahkan source kode berikut dibawah baris ke 12 pada class TestVehicle1.

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

Jalankan program, apakah output dari program tersebut?

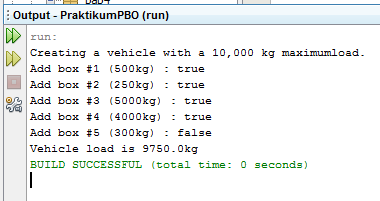
Kembalikan program seperti semula.

Jawab:



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

Jawab: Saat program dijalankan maka outputnya adalah



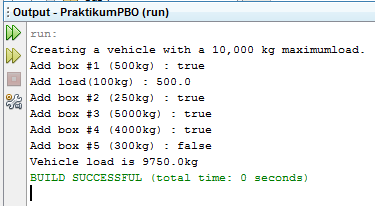
1. Tambahkan source kode berikut dibawah baris ke 6 pada class TestVehicle1.

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

Jalankan program, apakah output dari program tersebut?

Kembalikan program seperti semula.

Jawab:



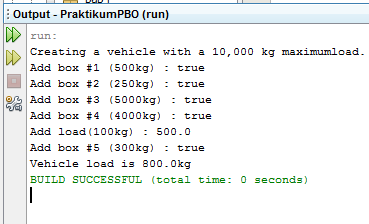
1. Tambahkan source kode berikut dibawah baris ke 12 pada class TestVehicle1.

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

Jalankan program, apakah output dari program tersebut?

Kembalikan program seperti semula.

Jawab:



1. **KESIMPULAN**

Encapsulation adalah proses menyembunyikan detil implementasi sebuah objek atau pembungkusan attribut (field atau variabel) dan tingkah laku (metode) di dalam sebuah kelas atau bisa juga memiliki pengertian suatu cara untuk menyembunyikan informasi detail dari suatu class.

Access modifier atau hak akses adalah keyword yang digunakan untuk menentukan spesifikasi tingkat akses suatu vaiable atau method (anggota kelas). Pengaksesan yang dimaksud bisa berupa pengaksesan dalam class yang sama, turunan maupun dari luar kelas di mana variable dan method dideklarasikan.

Dalam enkapsulasi terdapat hak akses public, protected, default dan private.

1. Public

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

Contoh :

class Mahasiswa {

public String nama;

public String kata() {

return “SAYA ANGELA”;

}

}

1. Protected

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

Contoh :

class Mahasiswa {

protected String nama;

protected String kata() {

return “SAYA ANGELA”;

}

}

1. Default

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

Contoh :

class Mahasiswa {

String nama;

String kata() {

return “SAYA ANGELA”;

}

}

1. Private

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

Contoh :

class Mahasiswa {

private String nama;

private String kata() {

return “SAYA ANGELA”;

}

}