

LAPORAN PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK

BAB : Encapsulation
NAMA : DANI ADRIAN
NIM : 225150201111009
ASISTEN : Tengku Muhammad Rafi Rahardiansyah
Muhammad Bin Djafar Almasyhur
TGL PRAKTIKUM : 16 Maret 2023

A Encapsulation 1

Pertanyaan

1. Lakukan percobaan diatas dan benahi jika menemukan kesalahan!

Source Code

Sebelum dibenahi

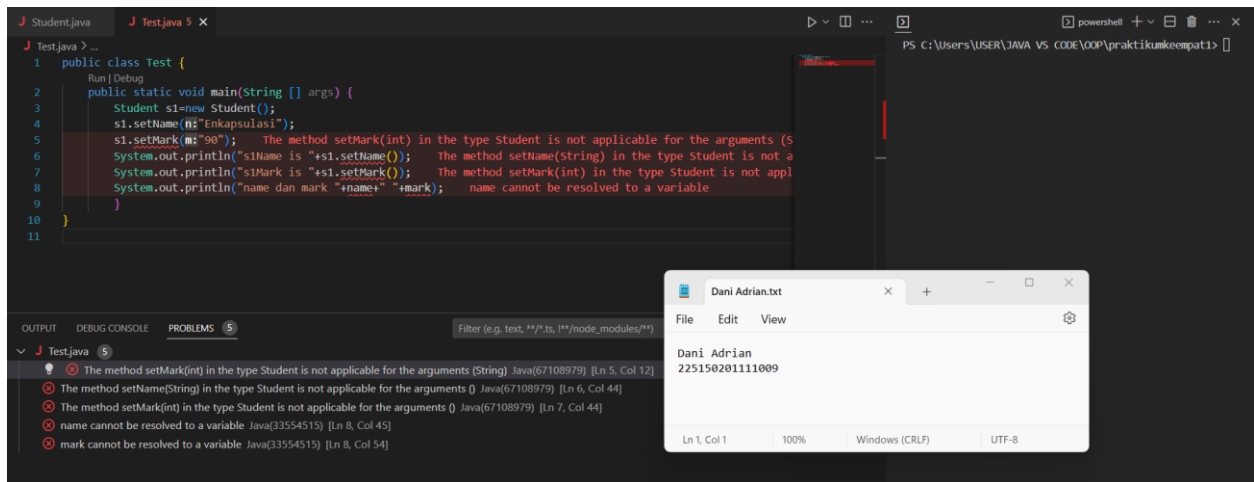
1.	public class Test {
2.	public static void main(String[] args) {
3.	Student s1 = new Student();
4.	s1.setName("Enkapsulasi");
5.	s1.setMark("90");
6.	System.out.println("s1Name is " + s1.setName());
7.	System.out.println("s1Mark is " + s1.setMark());
8.	System.out.println("name dan mark " + name + " " +
9.	mark);
9.	}
10.	}

Setelah dibenahi

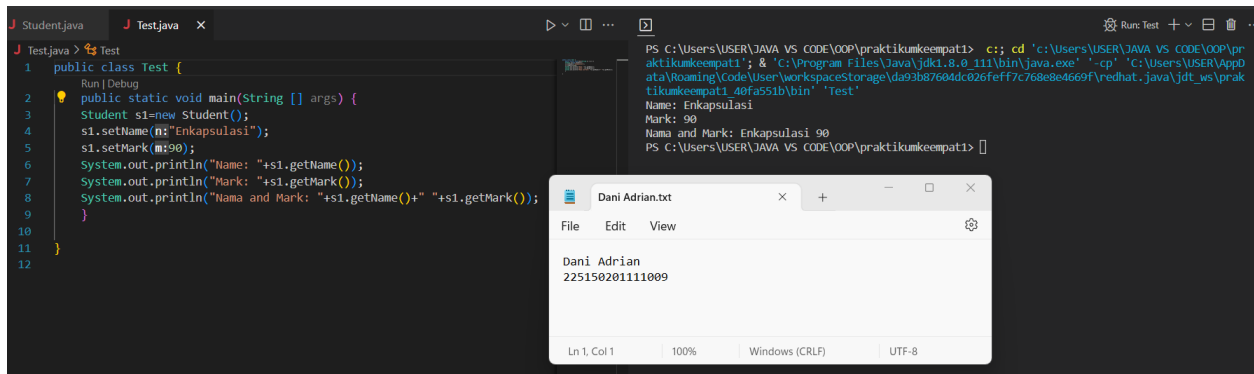
1.	public class Test {
2.	public static void main(String [] args) {
3.	Student s1=new Student();
4.	s1.setName("Enkapsulasi");
5.	s1.setMark(90);
6.	System.out.println("s1Name is "+s1.getName());
7.	System.out.println("s1Mark is "+s1.getMark());
8.	System.out.println("name dan mark "+
9.	s1.getName()+" "+s1.getMark());
9.	}
10.	}

Output

Sebelum dibenahi



Setelah dibenahi



Penjelasan

Kesalahan kode terjadi karena salah satu parameter tidak cocok dengan jenis datanya, dan terdapat pemanggilan metode yang salah untuk menampilkan nilai dari suatu variabel.

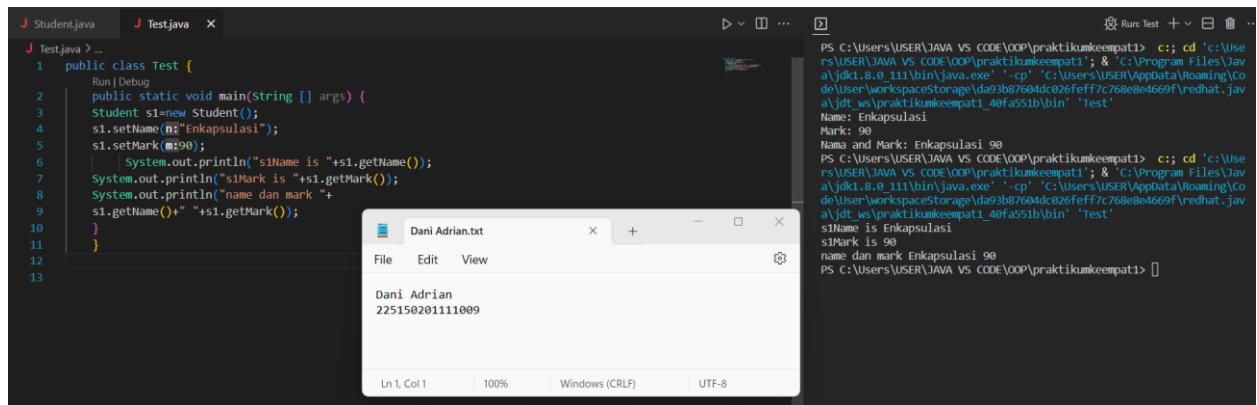
Pertanyaan

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

Source code

1.	<code>public class Test {</code>
2.	<code> public static void main(String [] args) {</code>
3.	<code> Student s1=new Student();</code>
4.	<code> s1.setName("Enkapsulasi");</code>
5.	<code> s1.setMark(90);</code>
6.	<code> System.out.println("s1Name is "+s1.getName());</code>
7.	<code> System.out.println("s1Mark is "+s1.getMark());</code>
8.	<code> System.out.println("name dan mark "+</code>
	<code> s1.getName()+" "+s1.getMark());</code>
9.	<code> }</code>
10.	<code>}</code>

Output



Penjelasan

Kesalahan pada kode tersebut sudah diperbaiki. Jika kode tersebut diganti, maka akan dicetak "s1Name is Enkapsulasi" karena getName dalam System.out.println() akan menghasilkan nilai yang dicetak.

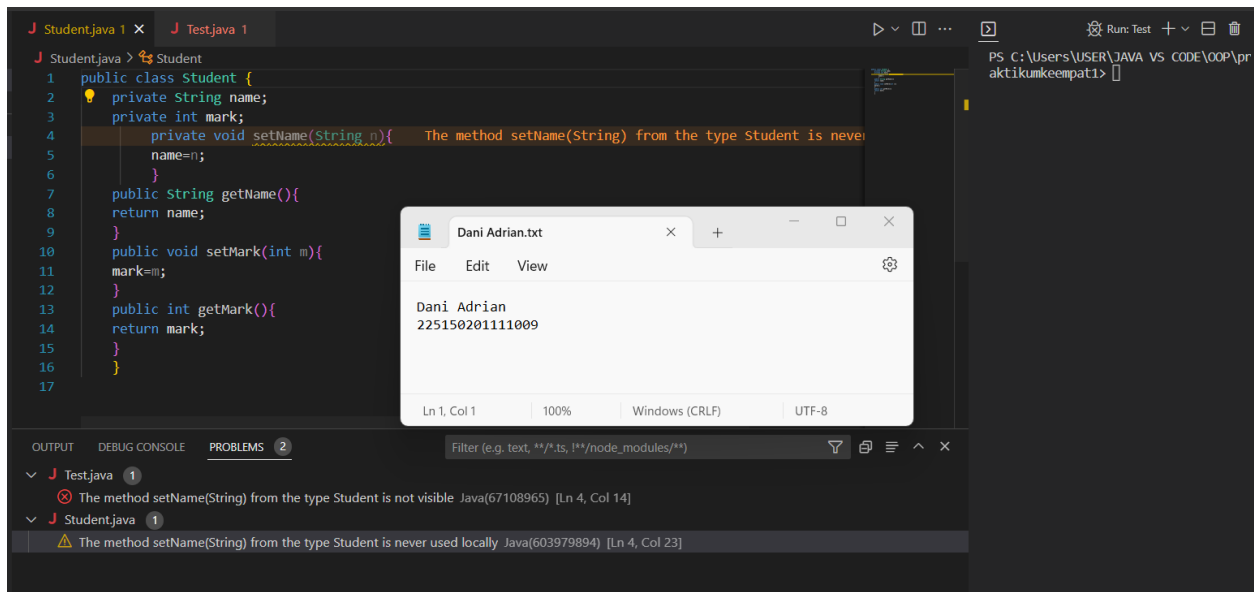
Pertanyaan

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

Source code

1.	public class Student {
2.	private String name;
3.	private int mark;
4.	private void setName(String n) {
5.	name=n;
6.	}
7.	public String getName() {
8.	return name;
9.	}
10.	public void setMark(int m){
11.	mark=m;
12.	}
13.	public int getMark(){
14.	return mark;
15.	}
16.	}

Output



Penjelasan

Tidak ada hasil yang keluar. Ketika method setName dipanggil di Class Test, akan terjadi kesalahan karena tidak dapat diakses. Modifikator akses Private hanya dapat dipanggil di kelas yang sama.

Pertanyaan

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

Source Code

1.	package praktikumkeempat1;
2.	
3.	public class Student {
4.	private String name;
5.	private int mark;
6.	public void setName(String n){
7.	name=n;
8.	}
9.	public String getName(){
10.	return name;
11.	}
12.	public void setMark(int m){
13.	mark=m;
14.	}
15.	public int getMark(){
16.	return mark;
17.	}
18.	}

1.	package praktikumkeempat1;
2.	
3.	public class Test {
4.	public static void main(String [] args) {
5.	Student s1=new Student();
6.	s1.setName("Enkapsulasi");
7.	s1.setMark(90);

8.	<code>System.out.println("s1Name is "+s1.getName());</code>
9.	<code>System.out.println("s1Mark is "+s1.getMark());</code>
10.	<code>System.out.println("name dan mark "+ s1.getName()+" "+s1.getMark());</code>
11.	<code>}</code>
12.	<code>}</code>

Output

```

package praktikumkeempat1;

public class Test {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.setName(n:"Enkapsulasi");
        s1.setMark(m: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());
    }
}
  
```

```

PS C:\Users\USER> cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat1'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\da93b87604dc026feff7c768e8e4669f\redhat.java\jdt_ws\praktikumkeempat1_40fa551b\bin' 'praktikumkeempat1.Test'
s1Name is Enkapsulasi
s1Mark is 90
name dan mark Enkapsulasi 90
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat1>
  
```

Penjelasan

Enkapsulasi adalah konsep untuk melindungi data dari akses yang tidak diinginkan. Dengan memodifikasi access modifier dari private menjadi public, data bisa diakses dari package lain. Enkapsulasi akan tetap berfungsi karena tujuannya adalah melindungi data dari akses yang tidak diinginkan.

Encapsulation 2

Pertanyaan

1. Method apakah yang menjadi accessor (getter) ?

Source Code

1.	<code>public double getLoad() {</code>
2.	<code> return this.load;</code>
3.	<code>}</code>
4.	<code>public double getMaxLoad() {</code>
5.	<code> return this.maxLoad;</code>
6.	<code>}</code>

Output

```

this.maxLoad = max;
}

public double getLoad() {
    return this.load;
}

public double getMaxLoad() {
    return this.maxLoad;
}

public boolean addBox(double weight) {
  
```

```

Dani Adrian
225150201111009
  
```

Penjelasan

Metode aksesornya adalah `getLoad()` dan `getMaxLoad()` karena keduanya mengambil nilai variabel yang bersifat privat.

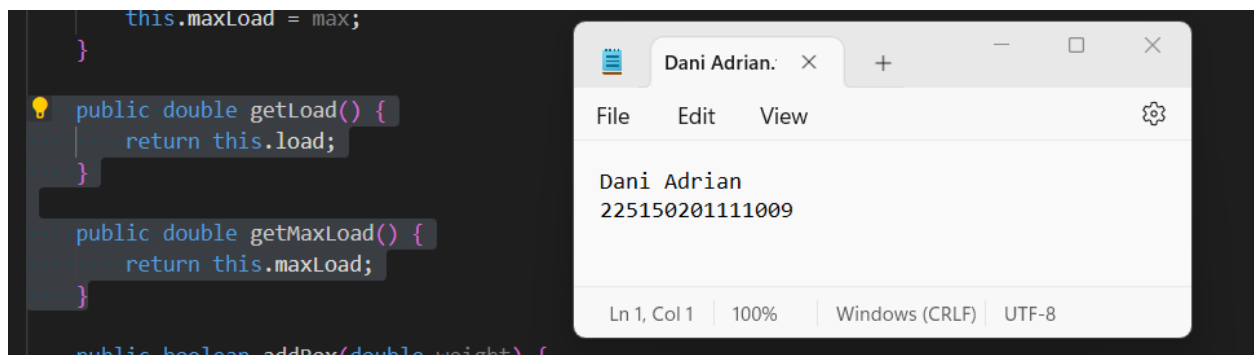
Pertanyaan

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? Kembalikan program seperti semula.

Source code

1.	<code>public class TestVehicle1{</code>
2.	<code> public static void main(String[] args){</code>
3.	<code> System.out.println("Creating a vehicle with a</code>
4.	<code> 10,000kg maximumload.");</code>
5.	<code> Vehicle1 vehicle = new Vehicle1(10000);</code>
6.	<code> System.out.println("Add box #1 (500kg) : " +</code>
7.	<code> vehicle.addBox(500));</code>
8.	<code> System.out.println("Add load(100kg) : " +</code>
9.	<code> (vehicle.load=500));</code>
10.	<code> System.out.println("Add box #2 (250kg) : " +</code>
11.	<code> vehicle.addBox(250));</code>
12.	<code> System.out.println("Add box #3 (5000kg) : " +</code>
13.	<code> vehicle.addBox(5000));</code>
14.	<code> System.out.println("Add box #4 (4000kg) : " +</code>
15.	<code> vehicle.addBox(4000));</code>
16.	<code> System.out.println("Add box #5 (300kg) : " +</code>
17.	<code> vehicle.addBox(300));</code>
18.	<code> System.out.println("Vehicle load is " +</code>
19.	<code> vehicle.getLoad() + "kg");</code>
20.	<code> }</code>
21.	<code>}</code>

Output



The screenshot shows a code editor with Java code. The code defines a `Vehicle1` class with a `load` attribute and methods `getLoad()`, `getMaxLoad()`, and `addBox(double weight)`. The `main` method creates a `Vehicle1` object and performs several operations: adding boxes of 500kg, 250kg, 5000kg, 4000kg, and 300kg, and then printing the current load. The terminal window on the right shows the output of the program, which is the name 'Dani Adrian' and the ID '225150201111009'.

Penjelasan

Terjadi kesalahan. Variabel `load` tidak dapat diakses secara langsung. Untuk mengisi nilai variabel `load`, kita harus melalui method `addBox()`.

Pertanyaan

3. Ubahlah tipe data pada atribut load dan maxload pada class Vehicle1 menjadi public. Jalankan program, apakah output dari program tersebut?
 - a. 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.
 - b. Tambahkan source code 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.

Source code

1.	public class Vehicle1
2.	{
3.	public double load, maxLoad;
4.	public Vehicle1 (double max) {
5.	this.maxLoad = max;
6.	}
7.	public double getLoad() {
8.	return this.load;
9.	}
10.	public double getMaxLoad() {
11.	return this.maxLoad;
12.	}
13.	public boolean addBox(double weight) {
14.	double temp = 0.0D;
15.	temp = this.load + weight;
16.	if(temp <= maxLoad) {
17.	this.load = this.load + weight;
18.	return true;
19.	}
20.	else
21.	{
22.	return false;
23.	}
24.	}
25.	}

A.

1.	public class TestVehicle1{
----	----------------------------

2.	<code>public static void main(String[] args){</code>
3.	<code>System.out.println("Creating a vehicle with a 10,000kg maximumload.");</code>
4.	<code>Vehicle1 vehicle = new Vehicle1(10000);</code>
5.	<code>System.out.println("Add box #1 (500kg) : " + vehicle.addBox(500));</code>
6.	<code>System.out.println("Add load(100kg) : " + (vehicle.load=500));</code>
7.	<code>System.out.println("Add box #2 (250kg) : " + vehicle.addBox(250));</code>
8.	<code>System.out.println("Add box #3 (5000kg) : " + vehicle.addBox(5000));</code>
9.	<code>System.out.println("Add box #4 (4000kg) : " + vehicle.addBox(4000));</code>
10.	<code>System.out.println("Add box #5 (300kg) : " + vehicle.addBox(300));</code>
11.	<code>System.out.println("Vehicle load is "+ vehicle.getLoad() + "kg");</code>
12.	<code>}</code>
13.	<code>}</code>

B.

1.	<code>public class TestVehicle1{</code>
2.	<code>public static void main(String[] args){</code>
3.	<code>System.out.println("Creating a vehicle with a 10,000kg maximumload.");</code>
4.	<code>Vehicle1 vehicle = new Vehicle1(10000);</code>
5.	<code>System.out.println("Add box #1 (500kg) : " + vehicle.addBox(500));</code>
6.	<code>System.out.println("Add box #2 (250kg) : " + vehicle.addBox(250));</code>
7.	<code>System.out.println("Add box #3 (5000kg) : " + vehicle.addBox(5000));</code>
8.	<code>System.out.println("Add box #4 (4000kg) : " + vehicle.addBox(4000));</code>
9.	<code>System.out.println("Add load(100kg) : " + (vehicle.load=500));</code>
10.	<code>System.out.println("Add box #5 (300kg) : " + vehicle.addBox(300));</code>
11.	<code>System.out.println("Vehicle load is "+ vehicle.getLoad() + "kg");</code>
12.	<code>}</code>
13.	<code>}</code>

Output

A.


```

1 package praktikumkeempat2;
2 public class TestVehicle1 {
3     public static void main(String[] args){
4         System.out.println("Creating a vehicle with a 10,000 kg maximumload.");
5         Vehicle1 vehicle = new Vehicle1(max:10000);
6         System.out.println("Add box #1 (500kg) : " + vehicle.addBox(weight:500));
7         System.out.println("Add load(100kg) : " + (vehicle.load-500));
8         System.out.println("Add box #2 (250kg) : " + vehicle.addBox(weight:250));
9         System.out.println("Add box #3 (5000kg) : " + vehicle.addBox(weight:5000));
10        System.out.println("Add box #4 (4000kg) : " + vehicle.addBox(weight:4000));
11        System.out.println("Add box #5 (300kg) : " + vehicle.addBox(weight:300));
12        System.out.println("Vehicle load is "+vehicle.getLoad() + "kg");
13    }
14 }
15
16
17

```

```

PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2> c:: cd 'c:\Users\USER\JAVA V
S CODE\OOP\praktikumkeempat2' & 'c:\Program Files\Java\jdk1.8.0_111\bin\java.exe'
'-cp' 'c:\Users\USER\AppData\Roaming\Code\User\workspacestorage\4544a6c988112f1fe
f11997f686986ed\redhat.java\jdt_ws\praktikumkeempat2_40fa551c\bin' 'praktikumkeemp
at2.TestVehicle1'
Creating a vehicle with a 10,000 kg maximumload.
Add box #1 (500kg) : true
Add load(100kg) : 500.0
Add box #2 (250kg) : true
Add box #3 (5000kg) : true
Add box #4 (4000kg) : true
Add box #5 (300kg) : false
Vehicle load is 9750.0kg
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2>

```

B.

```

1 package praktikumkeempat2;
2 public class TestVehicle1 {
3     public static void main(string[] args){
4         System.out.println("creating a vehicle with a 10,000 kg maximumload.");
5         Vehicle1 vehicle = new Vehicle1(max:10000);
6         System.out.println("Add box #1 (500kg) : " + vehicle.addBox(weight:500));
7         System.out.println("Add box #2 (250kg) : " + vehicle.addBox(weight:250));
8         System.out.println("Add box #3 (5000kg) : " + vehicle.addBox(weight:5000));
9         System.out.println("Add box #4 (4000kg) : " + vehicle.addBox(weight:4000));
10        System.out.println("Add load(100kg) : " + (vehicle.load-500));
11        System.out.println("Add box #5 (300kg) : " + vehicle.addBox(weight:300));
12        System.out.println("Vehicle load is "+vehicle.getLoad() + "kg");
13    }
14 }
15
16

```

```

PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2> c:: cd 'c:\Users\USER\JAVA V
S CODE\OOP\praktikumkeempat2' & 'c:\Program Files\Java\jdk1.8.0_111\bin\java.exe'
'-cp' 'c:\Users\USER\AppData\Roaming\Code\User\workspacestorage\4544a6c988112f1fe
f11997f686986ed\redhat.java\jdt_ws\praktikumkeempat2_40fa551c\bin' 'praktikumkeemp
at2.TestVehicle1'
Creating a vehicle with a 10,000 kg maximumload.
Add box #1 (500kg) : true
Add box #2 (250kg) : true
Add box #3 (5000kg) : true
Add box #4 (4000kg) : true
Add load(100kg) : 500.0
Add box #5 (300kg) : true
Vehicle load is 800.0kg
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2>

```

Penjelasan

Meskipun kode tersebut dapat dijalankan karena access modifier public memungkinkan akses di mana saja, namun menggunakan akses langsung untuk menambahkan nilai ke dalam variabel 'load' tidak memanfaatkan enkapsulasi secara optimal. Sebaiknya, nilai harus ditambahkan melalui method addBox() sehingga terdapat seleksi terlebih dahulu sebelum data dimasukkan ke dalam variabel. Hal ini penting untuk menjaga prinsip enkapsulasi dan melindungi data dari akses yang tidak diinginkan.

Pertanyaan

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

Source code

1.	public class Vehicle1{
2.	
3.	protected double load, maxLoad;
4.	
5.	public Vehicle1 (double max){
6.	this.maxLoad = max;

7.	}
8.	
9.	public double getLoad(){
10.	return this.load;
11.	}
12.	
13.	public double getMaxLoad(){
14.	return this.maxLoad;
15.	}
16.	
17.	public boolean addBox(double weight){
18.	double temp = 0.0D;
19.	temp = this.load + weight;
20.	if(temp <= maxLoad){
21.	this.load = this.load + weight;
22.	return true;
23.	}
24.	else
25.	{
26.	return false;
27.	}
28.	}
29.	}
30.	

Output

```

PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2> c++; cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2'; & 'C:\Program Files\Java\jdk-11.0.11\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\4544e4c988312f1fef11997f686986ed\re
pat2_40fa551c\bin' 'praktikumkeempat2.TestVehicle1'
Creating a vehicle with a 10,000 kg maximumload.
Add box #1 (500kg) : true
Add load(100kg) : 500.0
Add box #2 (250kg) : true
Add box #3 (5000kg) : true
Add box #4 (4000kg) : true
Add box #5 (300kg) : false
Vehicle load is 9750.0kg
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2>

```

The screenshot shows a terminal window with the following output:

```

Creating a vehicle with a 10,000 kg maximumload.
Add box #1 (500kg) : true
Add load(100kg) : 500.0
Add box #2 (250kg) : true
Add box #3 (5000kg) : true
Add box #4 (4000kg) : true
Add box #5 (300kg) : false
Vehicle load is 9750.0kg

```

Overlaid on the terminal is a Notepad window titled "Dani Adrian:" containing the same output text:

```

Dani Adrian
225150201111009

```

The Notepad window also shows a status bar at the bottom: "Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8".

```

PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2> c++; cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2';
& 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\4544e4c988312f1fef11997f686986ed\redhat.java\jdt_ws\praktikumkeempat2_40fa551c\bin' 'praktikumkeempat2.TestVehicle1'
Creating a vehicle with a 10,000 kg maximumload.
Add box #1 (500kg) : true
Add box #2 (250kg) : true
Add box #3 (5000kg) : true
Add box #4 (4000kg) : true
Add load(100kg) : 500.0
Add box #5 (300kg) : true
Vehicle load is 800.0kg
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2>

```

Dani Adrian
225150201111009

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

Penjelasan

Kode tersebut tetap akan bisa berjalan karena akses modifier protected bisa diakses oleh file itu sendiri, file yang berada dalam satu package dan subclassnya.

Pertanyaan

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

Source code

1.	public class Vehicle1{
2.	double load, maxLoad;
3.	
4.	public Vehicle1 (double max){
5.	this.maxLoad = max;
6.	}
7.	
8.	public double getLoad(){
9.	return this.load;
10.	}
11.	
12.	public double getMaxLoad(){
13.	return this.maxLoad;
14.	}
15.	
16.	public boolean addBox(double weight){
17.	double temp = 0.0D;
18.	temp = this.load + weight;
19.	if(temp <= maxLoad){
20.	this.load = this.load + weight;
21.	return true;
22.	}
23.	else
24.	{
25.	return false;
26.	}
27.	}
28.	}
29.	

Output

```
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2> c:: cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\4544e4c988312f1fef11997f686986ed\redhat.java\jdt_ws\praktikumkeempat2_40fa551c\bin' 'praktikumkeempat2.TestVehicle1'
Creating a vehicle with a 10,000 kg maximumload.
Add box #1 (500kg) : true
Add load(100kg) : 500.0
Add box #2 (250kg) : true
Add box #3 (5000kg) : true
Add box #4 (4000kg) : true
Add box #5 (300kg) : false
Vehicle load is 9750.0kg
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2> 
```

Dani Adrian: X

+

-

□

X

File

Edit

View

⚙

Dani Adrian
225150201111009

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

```
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2> c:: cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\4544e4c988312f1fef11997f686986ed\redhat.java\jdt_ws\praktikumkeempat2_40fa551c\bin' 'praktikumkeempat2.TestVehicle1'
Creating a vehicle with a 10,000 kg maximumload.
Add box #1 (500kg) : true
Add box #2 (250kg) : true
Add box #3 (5000kg) : true
Add box #4 (4000kg) : true
Add load(100kg) : 500.0
Add box #5 (300kg) : true
Vehicle load is 800.0kg
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat2> 
```

Dani Adrian: X

+

-

□

X

File

Edit

View

⚙

Dani Adrian
225150201111009

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

Penjelasan

Kode tersebut masih dapat berjalan karena access modifier default dapat diakses oleh file itu sendiri dan file yang berada dalam satu package yang sama.

Tugas Praktikum

Pertanyaan

Situasi : Apabila studi kasus KRS mempunyai hubungan dengan data mahasiswa (NIM, Nama, Alamat, No Telp, Jenis Kelamin, Program Studi, dst.)

Soal : Buatlah class diagram dan kode program agar dapat menerapkan sifat enkapsulasi untuk class Mahasiswa

Output : Data mahasiswa setelah diinputkan

Input : NIM, Nama, Alamat, No Telp, Jenis Kelamin, Program Studi, dst.

Source code

KRS.java	
1.	public class KRS {
2.	private MataKuliah[] mataKuliah;
3.	private int jumlahMataKuliah;
4.	private boolean selesai;

5.	
6.	KRS ()
7.	{
8.	mataKuliah = new MataKuliah[5];
9.	this.jumlahMataKuliah = 0;
10.	this.selesai = false;
11.	}
12.	
13.	public void tambahMataKuliah (MataKuliah mataKuliah)
14.	{
15.	if (mataKuliah.getNama ().toUpperCase ().equals ("NULL") mataKuliah.getKode ().toUpperCase ().equals ("NULL"))
16.	{
17.	selesai = true;
18.	}
19.	if (jumlahMataKuliah < 5 && selesai == false)
20.	{
21.	this.mataKuliah[jumlahMataKuliah] = mataKuliah;
22.	jumlahMataKuliah++;
23.	}
24.	else if (jumlahMataKuliah >= 5 && selesai == false)
25.	{
26.	System.out.println("Error! Maksimal 5");
27.	selesai = true;
28.	}

29.	}
30.	
31.	public int getMatkul()
32.	{
33.	return jumlahMataKuliah;
34.	}
35.	
36.	public boolean getCondition()
37.	{
38.	return selesai;
39.	}
40.	
41.	public void print(Mahasiswa ms)
42.	{
43.	System.out.println();
44.	System.out.println("Data Mahasiswa~~~~~");
45.	System.out.println("NIM : " + ms.getNIM());
46.	System.out.println("Nama : " + ms.getNama());
47.	System.out.println("Alamat : " + ms.getAlamat());
48.	System.out.println("No Telp : " + ms.getTelp());
49.	System.out.println("Gender : " + ms.getGender());
50.	System.out.println("Prodi : " + ms.getProdi());
51.	System.out.println();
52.	System.out.println("Daftar Mata Kuliah/Kode:");
53.	for (int i = 0; i < jumlahMataKuliah; i++)
54.	{
55.	System.out.println((i+1)+". " + mataKuliah[i].getNama() + "/" + mataKuliah[i].getKode());
56.	}
57.	}
58.	}

Mahasiswa.java	
1.	public class Mahasiswa {
2.	private String NIM;
3.	private String nama;
4.	private String alamat;
5.	private String telp;
6.	private String gender;
7.	private String prodi;

8.	
9.	public String getNIM() {
10.	return NIM;
11.	}
12.	public void setNIM(String NIM) {
13.	this.NIM = NIM;
14.	}
15.	public String getNama() {
16.	return nama;
17.	}
18.	public void setNama(String nama) {
19.	this.nama = nama;
20.	}
21.	public String getAlamat() {
22.	return alamat;
23.	}
24.	public void setAlamat(String alamat) {
25.	this.alamat = alamat;
26.	}
27.	public String getTelp() {
28.	return telp;
29.	}
30.	public void setTelp(String telp) {
31.	this.telp = telp;
32.	}
33.	public String getGender() {
34.	return gender;
35.	}
36.	public void setGender(String gender) {
37.	this.gender = gender;
38.	}
39.	public String getProdi() {
40.	return prodi;
41.	}
42.	public void setProdi(String prodi) {
43.	this.prodi = prodi;
44.	}
45.	
46.	}

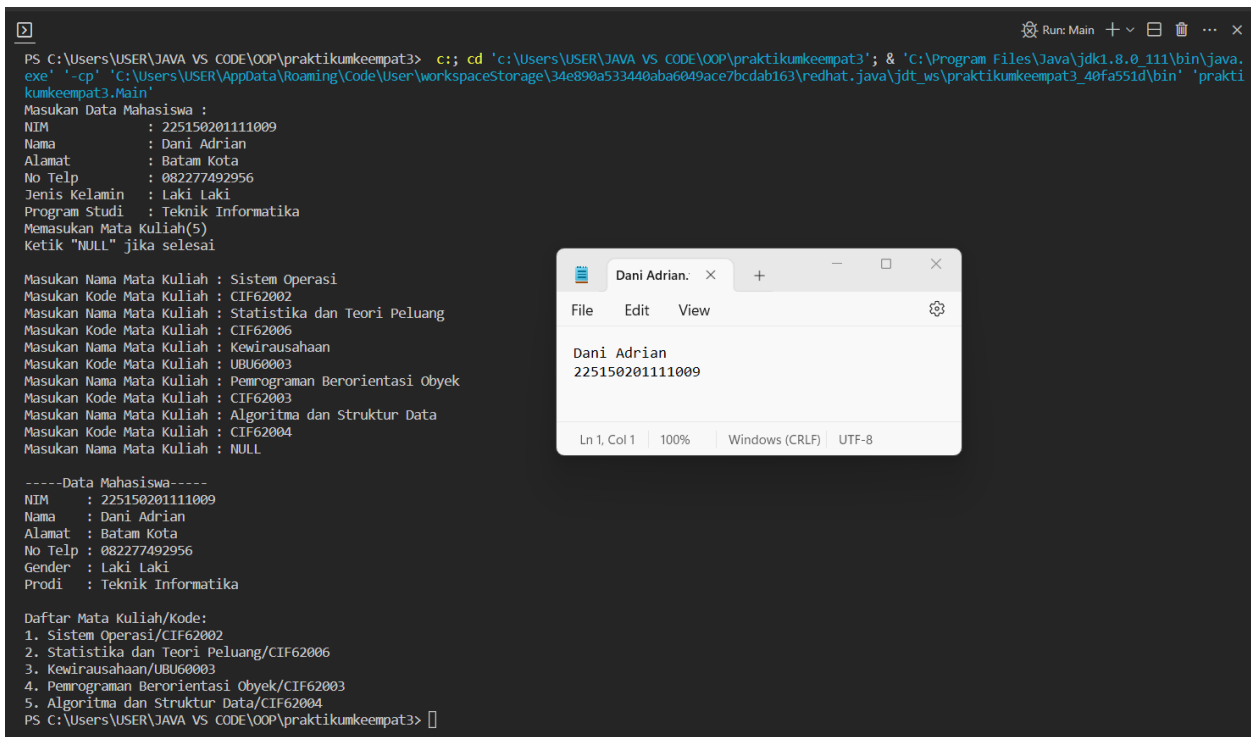
MataKuliah.java	
1.	public class MataKuliah {
2.	private String nama;
3.	private String kode;
4.	
5.	MataKuliah(String nama, String kode)
6.	{
7.	this.nama = nama;

8.	this.kode = kode;
9.	}
10.	
11.	public String getCode()
12.	{
13.	return kode;
14.	}
15.	
16.	public String getName()
17.	{
18.	return nama;
19.	}
20.	}

Main.java	
1.	public class Main {
2.	public static void main(String[] args) {
3.	Scanner sc = new Scanner(System.in);
4.	KRS krs = new KRS();
5.	Mahasiswa m = new Mahasiswa();
6.	System.out.println("Masukan Data Mahasiswa :");
7.	System.out.print("NIM : ");
8.	m.setNIM(sc.nextLine());
9.	System.out.print("nama : ");
10.	m.setName(sc.nextLine());
11.	System.out.print("alamat : ");
12.	m.setAlamat(sc.nextLine());
13.	System.out.print("no telp : ");
14.	m.setTelp(sc.nextLine());
15.	System.out.print("gender : ");
16.	m.setGender(sc.nextLine());
17.	System.out.print("prodi : ");
18.	m.setProdi(sc.nextLine());
19.	
20.	System.out.println("Memasukan Mata Kuliah (5)");
21.	System.out.println("Ketik \"NULL\" jika selesai");
22.	System.out.println();
23.	while(krs.getCondition() == false)
24.	{
25.	System.out.print("Masukan Nama Mata Kuliah : ");
26.	String nama = sc.nextLine();
27.	if (nama.toUpperCase().equals("NULL"))
28.	{
29.	break;
30.	}

31.	System.out.print("Masukan Kode Mata
32.	Kuliah : ");
33.	String kode = sc.nextLine();
34.	krs.tambahMataKuliah(new MataKuliah(nama,
35.	kode));
36.	}
37.	sc.close();
38.	krs.print(m);
39.	}

Output



```

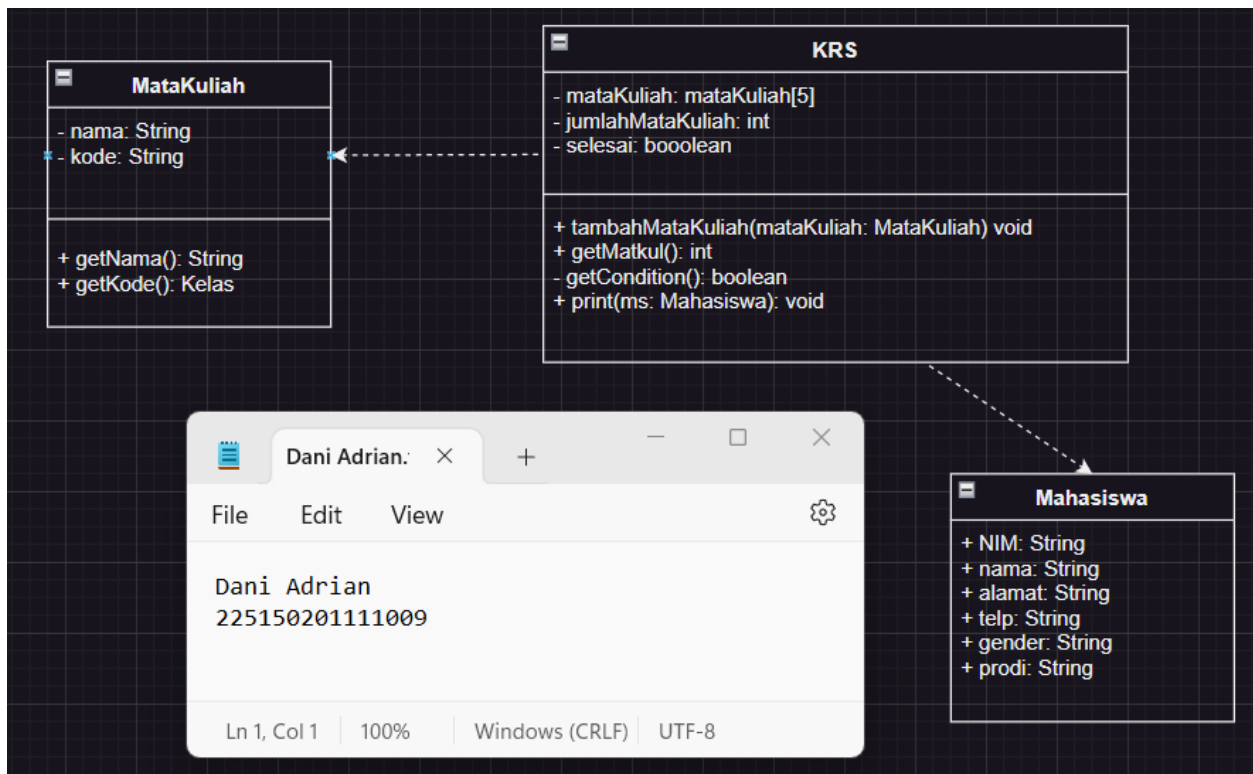
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat3> c:: cd 'c:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat3'; & 'C:\Program Files\Java\jdk1.8.0_111\bin\java.exe' '-cp' 'C:\Users\USER\AppData\Roaming\Code\User\workspaceStorage\34e890a533440aba6049ace7bcdab163\redhat.java\jdt_ws\praktikumkeempat3_40fa551d\bin' 'praktikumkeempat3.Main'
Masukan Data Mahasiswa :
NIM      : 225150201111009
Nama     : Dani Adrian
Alamat  : Batam Kota
No Telp  : 082277492956
Jenis Kelamin : Laki Laki
Program Studi : Teknik Informatika
Masukan Mata Kuliah(5)
Ketik "NULL" jika selesai

Masukan Nama Mata Kuliah : Sistem Operasi
Masukan Kode Mata Kuliah : CIF62002
Masukan Nama Mata Kuliah : Statistika dan Teori Peluang
Masukan Kode Mata Kuliah : CIF62006
Masukan Nama Mata Kuliah : Kewirausahaan
Masukan Kode Mata Kuliah : UBU60003
Masukan Nama Mata Kuliah : Pemrograman Berorientasi Obyek
Masukan Kode Mata Kuliah : CIF62003
Masukan Nama Mata Kuliah : Algoritma dan Struktur Data
Masukan Kode Mata Kuliah : CIF62004
Masukan Nama Mata Kuliah : NULL

-----Data Mahasiswa-----
NIM      : 225150201111009
Nama     : Dani Adrian
Alamat  : Batam Kota
No Telp  : 082277492956
Gender   : Laki Laki
Prodi    : Teknik Informatika

Daftar Mata Kuliah/Kode:
1. Sistem Operasi/CIF62002
2. Statistika dan Teori Peluang/CIF62006
3. Kewirausahaan/UBU60003
4. Pemrograman Berorientasi Obyek/CIF62003
5. Algoritma dan Struktur Data/CIF62004
PS C:\Users\USER\JAVA VS CODE\OOP\praktikumkeempat3>

```



Penjelasan

Kelas KRS memiliki ketergantungan terhadap kelas Mahasiswa, dikarenakan method `print` di kelas KRS (Line 41-57) membutuhkan objek Mahasiswa sebagai parameter. Dalam program utama (Main), kelas Mahasiswa dan KRS akan diinstansiasi (Line 4-5), kemudian input akan dimasukkan ke dalam atribut instance Mahasiswa yang telah dibuat menggunakan setter yang tersedia di kelas Mahasiswa. Selain itu, mata kuliah juga akan dimasukkan menggunakan method `tambahMataKuliah` yang telah disediakan di dalam kelas KRS. Setelah selesai memasukkan data mata kuliah, program akan langsung memanggil method `print` yang memiliki parameter instance dari objek Mahasiswa yang telah dibuat sebelumnya (Line 37). Pada method `print` di kelas KRS, getter yang telah tersedia di kelas Mahasiswa (Line 44-50) akan digunakan untuk mencetak data inputan yang telah dimasukkan.