MODUL PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK

NINF615

SEMESTER GASAL 2016/2017

PROGRAM STUDI S1 TEKNIK INFORMATIKA

MODUL PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK

DISUSUN OLEH:

Tim Asisten Praktikum

Jurusan Teknik Elektro

UM

JURUSAN TEKNIK ELEKTRO

PROGRAM STUDI S1 TEKNIK INFORMATIKA

	PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK	P-04	Enkapsulasi
	KODE MATAKULIAH : NINF615	SEMESTER : GASAL 2016/2017	

A. Tujuan

1. Mahasiswa mampu memahami dan menggunakan konsep *Encapsulation* dalam pemrograman java.
2. Mahasiswa mampu menerapkan konsep dasar aplikasi Java Swing.
3. Mahasiswa mampu membuat program sederhana berbasis GUI dengan Java.
4. Mahasiswa mampu menggunakan *class* JFrame dan JLabel.
5. Mahasiswa mampu menggunakan *class* JButton.

B. Dasar Teori

1. Enkapsulasi/ Pembungkusan (Encapsulation)

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

Dua hal yang mendasar dalam enkapsulasi yakni:

- ***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.

Enkapsulasi memiliki manfaat sebagai berikut:

- Modularitas

  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.

2. Assesor dan Mutator Method

Assesor method adalah *method* yang digunakan untuk membaca nilai variabel pada *class*, baik berupa *instance* maupun *static*. Sebuah *accessor method* umumnya dimulai dengan penulisan **get<namaInstanceVariable>**. *Method* ini juga mempunyai sebuah *return value*.

Mutator method adalah *method* yang digunakan untuk memberi atau mengubah nilai variabel dalam kelas, baik itu berupa *instance* maupun *static* variabel. Sebuah *mutator method* umumnya tertulis **set<namaInstanceVariabel>**. *Method* ini tidak menghasilkan balikan nilai atau *return value*.

3. Keuntungan menerapkan Encapsulasi

- 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.

4. Pengantar Swing

Banyak pengguna lebih menyukai aplikasi berbasis GUI (*Graphical User Interface*) karena memiliki tampilan lebih interaktif jika dibandingkan dengan tampilan berbasis teks atau *console* yang membosankan. Java menyediakan dua *class* untuk membangun aplikasi berbasis GUI, yaitu AWT (*Abstract Windowing Toolkit*) dan Swing. AWT terdapat dalam *package* java.awt, sedangkan Swing terdapat pada *package* javax.swing.

Komponen GUI yang terdapat dalam *package* java.awt bersifat *platform oriented*, yaitu bergantung pada suatu *platform* sistem operasi, sedangkan komponen GUI dalam *package* javax.swing memiliki sifat *lightweight*, yaitu dapat diaplikasikan dalam semua *platform* atau *multiplatform*. Hal ini merupakan kelebihan yang dimiliki oleh *package* javax.swing bila diban-dingkan dengan *package* java.awt.

Komponen-komponen Swing dapat dibagi menjadi beberapa kategori berikut:

a. *Root Container*.

   Terdiri dari JWindow, JFrame, JDialog, JApplet dan JInternalFrame.

b. *Label* dan *Button*.

   Terdiri dari JLabel, AbstractButton, JButton, JToggleButton, JCheckBox, JRadioButton, JMenuItem, JMenu, JCheckBoxMenuItem dan JRadioButtonMenuItem.

c. Komponen *Lightweight*.

   Terdiri dari JPanel, JMenuBar, JToolBar, JScrollBar, JSlider, JProgressBar, JList, JComboBox dan JSeparator.

d. Komponen *Text*.

   Terdiri dari JTextComponent, JTextField, JPasswordField, JTextArea, JEditorPane dan JTextPane.

e. Komponen *Space-saving*.

   Terdiri dari JScrollPane, JTabbedPane dan JSplitPane.

f. Komponen dengan Model Kompleks.

   Terdiri dari JTable dan JTree.

g. Komponen yang tersusun dari komponen lain.

   Terdiri dari JFileChooser, JColorChooser dan JOptionPane.

5. Pengenalan *class* JOptionPane

Class JOptionPane merupakan *class* dari *package* javax.swing yang digunakan untuk menampilkan *message dialog*. Beberapa *message dialog* yang dapat ditampilkan adalah jenis *plain message*, *information message*, *warning message*, *error message* dan *confirmation message*.

6. Pernyataan *import*

Pernyataan import merupakan mekanisme dalam program agar bisa meng-akses class yang terdapat dalam suatu package. Jika ingin meng-import semua class dari suatu package, maka bisa digunakan tanda asterisk (*) untuk mengganti nama class.

Meng-import class JOptionPane dari package javax.swing:

```
import javax.swing.JOptionPane;
```

Meng-import semua class dari package javax.swing:

```
import javax.swing.*;
```

7. Program sederhana berbasis GUI java

```
//meng-import class JOptionPane dari package javax.swing
import javax.swing.JOptionPane;

//membuat class JGui
public class JGui
{
    //membuat method main
    public static void main(String[] args)
    {
        //menampilkan teks dengan Message Dialog dari class JOptionPane
        JOptionPane.showMessageDialog(null, "Welcome to Java World... <^_^>",
                "GUI", JOptionPane.INFORMATION_MESSAGE);
    }
}
```

Kode program di atas digunakan untuk menampilkan teks di dalam tanda kutip ganda ("…"). Kode `import javax.swing.JOptionPane;` merupakan pernyataan untuk mengimport *class* `JOptionPane` dari *package* `javax.swing`. *Class* yang dibuat bernama `JGui` dengan *modifier* `public` yang berarti *class* ini bersifat dapat diakses oleh *class* lain dari *package* yang sama dan berbeda. Kode `JOptionPane.showMessageDialog` digunakan untuk menampilkan teks dalam *message dialog* dari *class* `JOptionPane`.

8. Pengenalan *class* Jframe dan *class* Jlabel

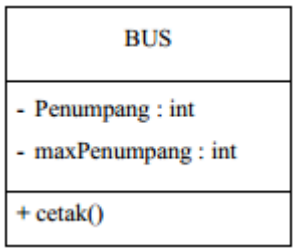
Class JFrame merupakan *class* dari *package* javax.swing yang digunakan untuk membuat *frame* sebagai window aplikasi dan sebagai tempat komponen GUI yang lain diletakkan. *Class* JLabel merupakan *class* dari dalam *package* javax.swing yang digunakan untuk membuat komponen berupa *label*. Komponen ini berfungsi untuk menampilkan teks yang pendek. Dalam praktiknya, komponen ini bisa digunakan untuk menampilkan *icon* (gambar berukuran kecil) atau pun untuk menampilkan teks dan *icon* sekaligus. Agar dapat menampilkan *icon*, maka dibutuhkan *import class* ImageIcon yang mengimplementasikan *interface* Icon dari *package* javax.swing. Selain itu, posisi relatif teks terhadap *icon* juga dapat diatur dengan menggunakan *class* SwingConstants dari *package* javax.swing. Seringkali saat *pointer mouse* diarahkan pada suatu komponen, akan muncul kotak kecil yang berisi keterangan posisi *pointer mouse* berada. Kotak kecil inilah yang disebut sebagai *tool tip*. Untuk membuatnya, maka digunakan *method* setToolTipText("keterangan") yang merupakan *subclass* dari JComponent.

9. Pengenalan Jbutton

Class JButton merupakan *class* dari *package* javax.swing yang digunakan untuk membuat komponen berupa tombol berisi teks maupun *icon*. Bila ingin membuat tombol interaktif yang *icon*-nya dapat berubah saat *pointer mouse* diarahkan pada tombol tersebut, maka dapat diatur dengan menggunakan *method* setRolloverIcon(iconObjectName).

C. Latihan

1. Latihan 1 → program tanpa encapsulation



a. Bus

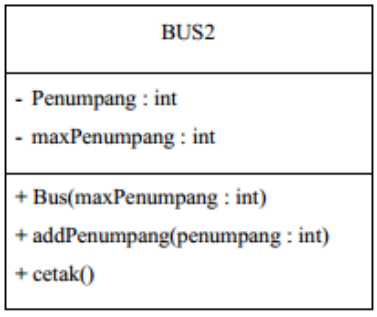
```
public class Bus {
    public int penumpang;
    public int maxPenumpang;

    public void cetak(){
        System.out.println("Penumpang Bus sekarang adalah "+penumpang);
        System.out.println("Penumpang maksimum seharusnya adalah "+maxPenumpang);
    }
}
```

b. UjiBus

```
public class UjiBus {
    public static void main(String[] args){
        //membuat objek busMini dari kelas Bus
        Bus busMini = new Bus();
        //memasukan nilai jumlah penumpang dan penumpang maksimal ke
        //dalam objek busMini
        busMini.penumpang = 5;
        busMini.maxPenumpang = 15;
        //memanggil method cetak pada kelas Bus
        busMini.cetak();

        //menambahkan penumpang pada busMini
        busMini.penumpang = busMini.penumpang + 5;
        //memanggil method cetak pada kelas Bus
        busMini.cetak();

        //mengurangi jumlah penumpang pada busMini
        busMini.penumpang = busMini.penumpang - 2;
        busMini.cetak();

        //menambahkan jumlah penumpang pad busMini
        busMini.penumpang = busMini.penumpang + 8;
        busMini.cetak();
    }
}
```

c. Jelaskan program dan output dari program !

2. Latihan 2 → program dengan encapsulation



a. Bus2

```
public class Bus2{
    private int penumpang;
    private int maxPenumpang;

    //konstruktor
    public Bus2(int maxPenumpang){
        this.maxPenumpang = maxPenumpang;
        penumpang = 0;
    }

    public void addPenumpang (int penumpang){
        int temp;
        temp = this.penumpang + penumpang;
        if (temp > maxPenumpang){
            System.out.println("Penumpang melebihi kuota");
        }
        else {
            this.penumpang = temp;
        }
    }

    public void cetak(){
        System.out.println("Penumpang bus sekarang: "+penumpang);
        System.out.println("Penumpang maks seharusnya: "+maxPenumpang);
    }
}
```

b. UjiBus2

```
public class UjiBus2{
    public static void main(String[] args) {
        Bus2 busBesar = new Bus2(40);
        busBesar.cetak();

        busBesar.addPenumpang(15);
        busBesar.cetak();

        busBesar.addPenumpang(5);
        busBesar.cetak();

        busBesar.addPenumpang(26);
        busBesar.cetak();
    }
}
```

c. Jelaskan program dan output dari program !

3. Latihan 3 → program encapsulasi dengan assesor dan mutator method

BUS2
- Penumpang : int - maxPenumpang : int - penumpangBaru : int
+ Bus(maxPenumpang : int) + addPenumpang(penumpang : int) + getPenumpang(password : int) + cetak()

Tambahkan method **getPenumpang** pada class **Bus** dan simpan dalam **Bus3.java**. Tambahkan aturan untuk mengakses data penumpang baru ke dalam method getPenumpang. Aturan yang ditambahkan memuat **kode akses (password)**. Jika password **benar**, maka data penumpang yang baru ditambahkan dan ditampilkan, jika password **salah**, maka ada peringatan bahwa password salah.

Lalu jelaskan program dan output dari program !

4. Latihan 4 → pembuatan frame

```
import javax.swing.JFrame;
public class CreateFrame
{
    public static void main(String[] args)
    {
        JFrame frame = new JFrame();
        frame.setTitle("CREATE FRAME");
        frame.setBounds(200,200,300,150);
        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

Jelaskan program dan output dari program !

5. Latihan 5 → pembuatan label

```
import javax.swing.JFrame;
import javax.swing.JLabel;

public class CreateLabel
{
    public static void main(String[] args)
    {
        JFrame frame = new JFrame();
        frame.setTitle("CREATE LABEL");
        frame.setBounds(200,200,300,150);
        JLabel label = new JLabel("S1 Pendidikan Teknik Informatika");
        label.setBounds(25,25,200,25);
        frame.add(label);
        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

a. Jelaskan program dan output dari program !

b. Adakah perbedaan jika perintah berikut dihilangkan? Jelaskan !

```
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

6. Latihan 6 → mengatur posisi relatif teks terhadap *icon*

```
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.ImageIcon;
import javax.swing.SwingConstants;

public class LabelIcon
{
    private JLabel label; //membuat objek label

    public LabelIcon()
    {
        JFrame frame = new JFrame();
        frame.setTitle("LABEL ICON");
        frame.setBounds(200,200,350,200);

        ImageIcon um = new ImageIcon("um.png");
        label = new JLabel();
        label.setBounds(25,25,325,100);
        label.setText("UNIVERSITAS NEGERI MALANG");
        label.setIcon(um);
        label.setHorizontalTextPosition(SwingConstants.LEFT);
        label.setVerticalTextPosition(SwingConstants.TOP);

        frame.add(label);
        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public static void main(String[] args)
    {
        LabelIcon frame = new LabelIcon();
    }
}
```

a. Jelaskan program dan output dari program !

b. Apa fungsi dari perintah berikut?

```
label.setHorizontalTextPosition(SwingConstants.LEFT);
label.setVerticalTextPosition(SwingConstants.TOP);
```

7. Latihan 7 → membuat button

```
import javax.swing.JFrame;
import javax.swing.JButton;

public class CreateButton
{
    public static void main(String[] args)
    {
        JFrame frame = new JFrame();
        JButton button = new JButton("TOMBOL");
        frame.setTitle("CREATE BUTTON");
        frame.setBounds(200,300,250,125);
        frame.add(button);
        button.setBounds(25,25,100,25);
        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

Jelaskan program dan output dari program !

8. Latihan 8 → membuat event pada button

```
import java.awt.event.*;
import javax.swing.*;

//class TryButton mengimplementasikan interface ActionListener
public class TryButton implements ActionListener
{
    private JButton btnEvent; //membuat objek btnEvent dari class JButton

    public TryButton()
    {
        JFrame frame = new JFrame();
        frame.setBounds(200,200,300,200);
        frame.setTitle("TRY BUTTON");

        btnEvent = new JButton("Button"); //menugaskan objek btnEvent
        frame.add(btnEvent);
        btnEvent.setBounds(50,50,175,50);

        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        btnEvent.addActionListener(this); //mendeteksi event pada button
    }
    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource()==btnEvent) //jika tombol ditekan
        {
            //aksi yang terjadi jika tombol ditekan
            JOptionPane.showMessageDialog(null, "Teks dalam tombol: \n"+e.getActionCommand());
        }
    }
    public static void main(String[] args)
    {
        TryButton frame = new TryButton();
    }
}
```

Jelaskan program dan output dari program !

D. Tugas Praktikum

1. Dari hasil program latihan 3, tambahkan method getAverage() untuk menghitung rata-rata jumlah penumpang yang ditambahkan , dan simpan dalam Bus4.java serta uji dengan UjiBus4.java.

Bus4
- penumpang : double - maxPenumpang : double - counter : double - penumpngBaru : double
+ Bus(maxPenumpang : double) + addPenumpang(penumpang : double) + getPenumpang(password : int) : double + getAverage() : double + cetak()

2. Buatlah sebuah *class* baru yang berisi *method main* untuk menjalankan program berikut!

```
class GajiPegawai
{
    //deklarasi variabel dengan modifier private
    private double gajiKotor, pajak, gajiBersih;
    private double potongan=75000;
    private String nama = "Surya";

    public void setGaji(double gaji) //mengatur nilai gajiKotor
    {
        gajiKotor=gaji;
    }
    public void hitungPajak() //menghitung nilai pajak
    {
        pajak=0.2*gajiKotor;
    }
    public void hitungGaji() //menghitung nilai gajiBersih
    {
        gajiBersih=gajiKotor-pajak-potongan;
    }
    public String namaPegawai() //menampilkan nama
    {
        return nama;
    }
    public double getGajiKotor() //menampilkan nilai gajiKotor
    {
        return gajiKotor;
    }
    public double getPajak() //menampilkan nilai pajak
    {
        return pajak;
    }
    public double getPotongan() //menampilkan nilai potongan
    {
        return potongan;
    }
    public double getGajiBersih() //menampilkan nilai gajiBersih
    {
        return gajiBersih;
    }
}
```

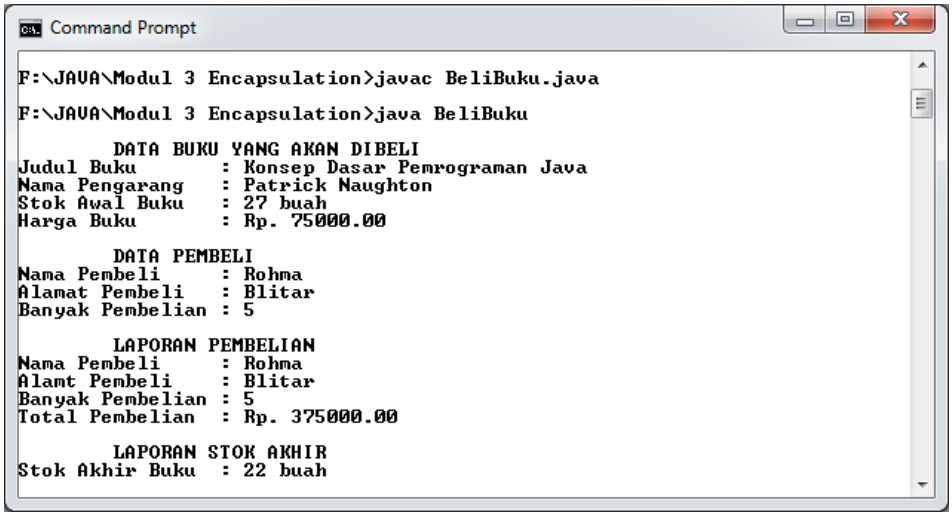
E. Tugas Rumah

1. Buatlah sebuah program berbasis GUI untuk menampilkan informasi tentang biodata Anda! Simpanlah informasinya dalam variabel! Gunakan *label* dan tambahkan *icon* untuk membuat tampilan menjadi lebih menarik! Contoh output:

2. Buatlah sebuah program berbasis GUI untuk menampilkan 3 buah *label* dengan tampilan yang berbeda! Berikan keterangan yang berbeda saat *pointer mouse* mengarah ke setiap *label* tersebut! Contoh output:

3. Lengkapi kode program berikut 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*! Berikut potongan kode program beserta contoh outputnya:

```
class Buku
{
    private String judulBuku="Konsep Dasar Pemrograman Java";
    private String pengarangBuku="Patrick Naughton";
    private int stokBuku=27;
    private int hargaBuku=75000;

    public void setPembeli(String nama)
    {
        namaPembeli=nama;
    }
    public void setAlamat(String alamat)
    {
        alamatPembeli=alamat;
    }
    public void setPembelian(int pembelian)
    {
        banyakPembelian=pembelian;
    }
    public void hitungBayar()
    {
        bayarBuku=hargaBuku*banyakPembelian;
    }
    public void hitungSisa()
    {
        sisaBuku=stokBuku-banyakPembelian;
    }
}
```


Command Prompt
F:\JAVA\Modul 3 Encapsulation>javac BeliBuku.java
F:\JAVA\Modul 3 Encapsulation>java BeliBuku
DATA BUKU YANG AKAN DIBELI
Judul Buku : Konsep Dasar Pemrograman Java
Nama Pengarang : Patrick Naughton
Stok Awal Buku : 27 buah
Harga Buku : Rp. 75000.00
DATA PEMBELI
Nama Pembeli : Rohma
Alamat Pembeli : Blitar
Banyak Pembelian : 5
LAPORAN PEMBELIAN
Nama Pembeli : Rohma
Alamt Pembeli : Blitar
Banyak Pembelian : 5
Total Pembelian : Rp. 375000.00
LAPORAN STOK AKHIR
Stok Akhir Buku : 22 buah