

# **BERORIENTASI OBJEK**

**NINF615**

**SEMESTER GASAL 2016/2017**

**MODUL PRAKTIKUM**

**PEMROGRAMAN**

**BERORIENTASI OBJEK**

**DISUSUN OLEH:**


**Tim Asisten Praktikum**

**Jurusan Teknik Elektro**

**UM**

**JURUSAN TEKNIK ELEKTRO**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

	<b>PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK</b>	<b>P-05</b>	<b>Polymorphism</b>
	<b>KODE MATAKULIAH : NINF615</b>	<b>SEMESTER : GASAL 2016/2017</b>	

### A. Tujuan

1. Mahasiswa mampu memahami dan menerapkan konsep *Polymorphism* dalam pemrograman java.
2. Mahasiswa mampu melakukan *overloading* terhadap *method*.
3. Mahasiswa mampu melakukan *overloading* terhadap *constructor method*.
4. Mahasiswa mampu menggunakan *class JTextField*.

### B. Dasar Teori

#### 1. Konsep *polymorphism*.

*Polymorphism* merupakan konsep sederhana dalam bahasa pemrograman berorientasi objek yang berarti kemampuan sebuah objek untuk menghasilkan aksi yang berbeda. Bila *method* yang sama dipanggil, maka aksi *method* yang akan dikerjakan tergantung pada tipe objeknya.

#### 2. **Overloading terhadap constructor method.**

*Overloading* terhadap konstruktor merupakan suatu mekanisme pembuatan konstruktor yang memiliki bentuk lebih dari satu. Dalam hal ini pembeda antara satu konstruktor dengan konstruktor yang lain berupa jumlah atau tipe parameter.

#### 3. **Pengenalan overloading method.**

Terkadang di dalam sebuah *class* ada lebih dari satu *method* yang namanya sama, tetapi memiliki parameter yang berbeda sehingga fungsinya pun berbeda. *Method* dengan kemampuan seperti ini disebut sebagai *overloading method*.

#### 4. **Pengenalan class JTextField.**

*Class JTextField* merupakan *class* dari *package javax.swing* yang digunakan untuk membuat komponen berupa *text field*. Komponen ini berfungsi untuk memasukkan data satu baris saja. Jika data yang dimasukkan tergolong sebagai

*password*, maka dapat digunakan `JPasswordField` dan dilengkapi dengan penggunaan *method* `getPassword()` untuk memperoleh string pada objeknya.

Data yang dimasukkan melalui *text field* memiliki tipe data `String`. Jika membutuhkan data berupa bilangan yang akan dilakukan perhitungan, maka dapat dikonversi ke dalam tipe data lain, misalnya dikonversi ke dalam tipe data integer dengan perintah `Integer.parseInt(objek.getText())`. Kemudian agar dapat menampilkan hasilnya kembali pada *text field*, maka bisa digunakan perintah `objek.setText(String.valueOf(nilai))`.

### C. Latihan

#### 1. Menerapkan *overloading* terhadap *method*.

Tulislah dan simpan kode program di bawah ini:

Kode program *class* Pilihan:

```
class Pilihan
{
    public void jurusan() //method tanpa parameter
    {
        String jrs="Pendidikan Teknik Informatika";
        System.out.println("\nPilihan 1: \nJurusan "+jrs);
    }

    public void jurusan(String jrs) //method dengan parameter
    {
        System.out.println("\nPilihan 2: \nJurusan "+jrs);
    }
}
```

Kode program *class* OverloadMethod1:

```
public class OverloadMethod1
{
    public static void main(String[] args)
    {

        System.out.println("\n*****
        *****");

        System.out.println("\n\tOVERLOAD TERHADAP METHOD #1\n");
    }
}
```

```

System.out.println("*****
****");

    Pilihan pilih = new Pilihan(); //membuat objek pilih
    pilih.jurusan(); //memanggil method tanpa parameter
    pilih.jurusan("Pendidikan Matematika"); //memanggil method
    dengan parameter
}
}

```

- a. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
- b. Berikan penjelasan terkait jalannya program ini!

## 2. Melakukan overloading terhadap method.

Tulishlah dan simpan kode program di bawah ini:

Kode program *class* Matematika:

```

class Matematika
{
    static public int kuadrat(int nilai)
    {
        return nilai*nilai;
    }

    static public double kuadrat(double nilai)
    {
        return nilai*nilai;
    }

    static public double kuadrat(String nilai)
    {
        double bilangan;
        bilangan=Double.valueOf(nilai).doubleValue();
        return bilangan*bilangan;
    }
}

```

- a. Buatlah sebuah *class* baru yang berisi *method main* untuk menjalankan program tersebut! Aturilah agar ketiga *method* di dalam *class* tersebut dapat digunakan semua! Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
- b. Berikan penjelasan terkait jalannya program ini!
- c. Jelaskan fungsi dari bagian kode program `bilangan=Double.valueOf (nilai) .doubleValue () ;!`

### 3. Menerapkan overloading terhadap constructor method.

Tuliskan dan simpan kode program di bawah ini:

Kode program *class* Mahasiswa:

```
class Mahasiswa
{
    private String nama;
    private int angkatan;
    public Mahasiswa () //konstruktor tanpa parameter
    {
        this.nama="Siwi";
        this.angkatan=2013;
    }
    public Mahasiswa (String nama, int angkatan) //kostruktor dengan parameter
    {
        this.nama=nama;
        this.angkatan=angkatan;
    }
    public void info ()
    {
        System.out.println("\nIdentitas Mahasiswa : ");
        System.out.println("Nama      : "+this.nama);
        System.out.println("Angkatan : "+this.angkatan);
    }
}
```

Kode program *class* OverloadConstructor1:

```
public class OverloadConstructor1
{
    public static void main (String [] args)
    {
```

```

System.out.println("\n*****
***");
        System.out.println("\n\tOVERLOAD TERHADAP KONSTRUKTOR
#1\n");

System.out.println("*****
*");

        Mahasiswa mhs1=new Mahasiswa("Surya", 2012); //menugaskan
objek mhs1
        mhs1.info(); //memanggil method info yang menampilkan data
dari kostruktor dengan parameter

        Mahasiswa mhs2=new Mahasiswa(); //membuat objek mhs2
        mhs2.info(); //memanggil method info yang menampilkan data
dari kostruktor tanpa parameter
    }
}

```

- a. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
- b. Berikan penjelasan terkait jalannya program ini!

#### 4. Melakukan overloading terhadap constructor method.

Tulislah dan simpan kode program di bawah ini:

Kode program *class* Lingkaran:

```

class Lingkaran
{
    double radius;

    Lingkaran() //konstruktor tanpa parameter
    {
        radius=1.0;
    }

    Lingkaran(double r) //konstruktor dengan parameter
    {
        radius=r;
    }
}

```

```

double luas() //method penghitung luas
{
    return radius*radius*Math.PI;
}
}

```

- Buatlah sebuah *class* baru yang berisi *method main* untuk menjalankan program tersebut! Aturlah agar nilai radius yang dioperasikan adalah 1, 10 dan 100. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
- Berikan penjelasan terkait jalannya program ini!

## 5. Membuat Text field.

Tulislah dan simpan kode program di bawah ini:

```

import javax.swing.JFrame;
import javax.swing.JTextField;

public class CreateTextField
{
    public static void main(String[] args)
    {
        JFrame frame = new JFrame();
        JTextField textfield = new JTextField(); //menciptakan
objek textfield dari class JTextField
        frame.setTitle("CREATE TEXT FIELD");
        frame.setBounds(200,200,300,200);
        frame.add(textfield); //menambahkan text field pada frame
        textfield.setBounds(50,50,150,25); //mengatur posisi
tampilan text field dan ukuran text field
        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}

```

- Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
- Berikan penjelasan terkait jalannya program ini!



## 6. Menampilkan kata dari Text Field.

Tuliskan dan simpan kode program di bawah ini:

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

//class TryTextField mengimplementasikan interface ActionListener
public class TryTextField implements ActionListener
{
    //membuat objek dari class yang akan digunakan
    private JTextField textfield;
    private JButton btnTampil;
    private JLabel label;

    public TryTextField()
    {
        JFrame frame = new JFrame();
        frame.setBounds(200,200,325,175);
        frame.setTitle("TRY TEXT FIELD");

        label = new JLabel("Input Kata : ");
        label.setBounds(25,25,100,25);
        frame.add(label);

        textfield = new JTextField();
        textfield.setBounds(125,25,150,25);
        frame.add(textfield);

        btnTampil = new JButton("Tampilkan Kata"); //menugaskan
objek btnTampil
        frame.add(btnTampil);
        btnTampil.setBounds(125,75,150,25);

        frame.setLayout(null);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        btnTampil.addActionListener(this); //mendeteksi event pada
btnTampil
    }

    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource()==btnTampil)
        {
            String kata = textfield.getText(); //mengambil kata
yang diinputkan oleh user
            JOptionPane.showMessageDialog(null, "Anda telah
menginput kata : \n"+kata, "INFORMASI",
JOptionPane.INFORMATION_MESSAGE);
        }
    }

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

- a. Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
- b. Berikan penjelasan terkait jalannya program ini!

#### D. Tugas Praktikum

Melakukan *overloading* terhadap *constructor method* dan *method*.

Diketahui potongan kode program berikut ini:

```
class Buku
{
    public Buku ()
    {
    }

    public void terbit (String judul)
    {
        System.out.print (judul+" ". );
    }
}

public class DaftarPustaka
{
    public static void main (String [] args)
    {
        Buku book1 = new Buku (penyusun, tahun);
        book.terbit (kota, penerbit);
    }
}
```

- a. Lengkapi kode program di atas untuk membuat sebuah program berbasis *console* di bidang kepastakaan! Lakukan *overloading* terhadap *constructor* `Buku ()` dan *method* `terbit (String judul)`! Aturlah agar *user* dapat melakukan *input* data saat menjalankan program! Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
- b. Berikan penjelasan terkait jalannya program yang Anda buat!

Contoh hasil eksekusi program:



```
Command Prompt
F:\JAUU\Modul 4 Polymorphism>javac DaftarPustaka.java
F:\JAUU\Modul 4 Polymorphism>java DaftarPustaka
*****
TAMPILAN DAFTAR PUSTAKA
*****
Masukkan Identitas Buku:
Judul Buku      : Menyusun Modul
Nama Penyusun   : Daryanto
Kota Terbit     : Yogyakarta
Nama Penerbit   : Gava Media
Tahun Terbit    : 2013
*****
Penulisan Daftar Pustaka :
Daryanto. 2013. Menyusun Modul. Yogyakarta:Gava Media.
```

Gambar 1. Contoh Tampilan Daftar Pustaka

## E. Tugas Rumah

1. Jelaskan mengapa diperlukan konsep *Polymorphism*!
2. Diketahui potongan kode program berikut ini:

```
public class FourOperations implements ActionListener
{
    private JTextField jTextField1, jTextField2, jTextFieldHasil;
    private JLabel label1, label2, labelHasil;

    public FourOperations ()
    {
        btnJumlah.addActionListener (this);
        btnKurang.addActionListener (this);
        btnBagi.addActionListener (this);
    }

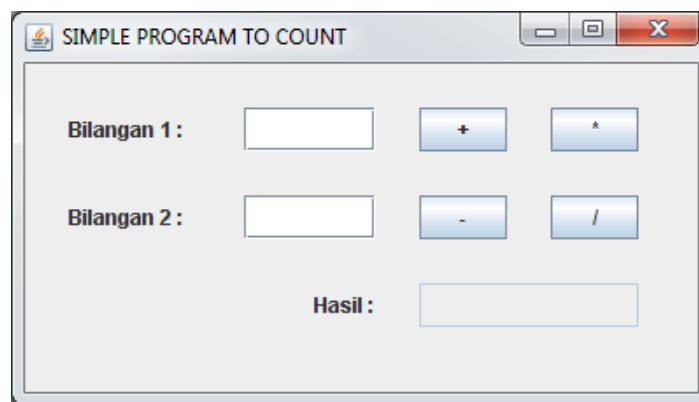
    public void actionPerformed (ActionEvent e)
    {
        if (e.getSource () == btnKali)
        {
            int
bilangan1=(Integer.parseInt (jTextField1.getText ().trim ());
            int
bilangan2=(Integer.parseInt (jTextField2.getText ().trim ());
            int hasil=bilangan1*bilangan2;
```

```
        jTextFieldHasil.setText(String.valueOf(hasil));  
    }  
}  
}
```

a. Buatlah sebuah program berbasis GUI yang minimal dapat melakukan perhitungan penjumlahan, pengurangan, perkalian dan pembagian! Gunakan *label*, *text field* dan *button*! Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!

b. Berikan penjelasan terkait jalannya program yang Anda buat!

Contoh hasil eksekusi program:



Gambar 2. Contoh Tampilan Kalkulator Sederhana