

# BAB 1

## PENGANTAR

### PEMROGRAMAN BERORIENTASI OBYEK

---

**Tujuan:**

1. *Mahasiswa dapat menjelaskan konsep pemrograman berbasis obyek.*
  2. *Mahasiswa dapat menerapkan proses enkapsulasi, inheritance dan polimorfisme pada pemrograman berbasis obyek*
  3. *Mahasiswa dapat membuat program berbasis menggunakan bahasa Java secara sederhana*
- 

Bahasa Java adalah bahasa pemrograman berorientasi obyek, sehingga untuk memperoleh pemahaman yang lebih baik terhadap materi pada bab-bab selanjutnya, perlu disampaikan beberapa topik mengenai pemrograman berorientasi obyek yang berhubungan langsung dengan topik-topik yang akan dibahas, diantaranya mengenai konsep pemrograman berorientasi obyek, *inheritance* dan *constructor*. Topik-topik lain dalam pemrograman berorientasi obyek akan dibahas lebih lanjut bila memang diperlukan.

#### 1.1. Pemrograman Berorientasi Obyek

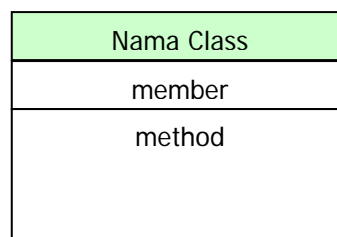
Pemrograman berorientasi obyek merupakan suatu konsep pemrograman dengan mengambil konsep obyek sebagai komponen dasar dari pemrogramannya. Obyek merupakan suatu kesatuan komponen dan struktur yang di dalamnya berisi atribut yang selanjutnya dinamakan dengan *member* dan *method* yang merupakan kumpulan fungsional dari suatu obyek. Sebagai suatu analogi obyek, kita ambil obyek mobil mempunyai member berupa roda, kemudi, body, pintu, lampu, dashborad

dan lainnya. Obyek mobil ini mempunyai method berupa maju, mundur, jalan, berhenti, dan berputar.

Dengan demikian dapat dikatakan bahwa obyek mempunyai sifat-sifat, yaitu:

- *Member* atau sering juga disebut dengan *attribut* yang menjelaskan variable, parameter atau keadaan (*state*) dari suatu obyek, misalkan pada obyek mobil terdapat member berupa roda, kemudi, seperti yang disebutkan di atas
- *Method* atau sering juga disebut dengan *behavior* yang menjelaskan perilaku, kegiatan atau kerja dari suatu obyek, misalkan pada obyek mobil terdapat method maju, mundur, berhenti, seperti yang disebutkan diatas.

Menulis program berbasis obyek menggunakan bahasa Java dilakukan dengan cara membentuk sebuah **class**, menentukan variabel member dan menentukan method. Pembentukan class dari suatu obyek ini dinamakan dengan **enkapsulasi**. Untuk menggambarkan suatu obyek digunakan suatu diagram yang dinamakan dengan class diagram. Model class di dalam class diagram seperti terlihat pada gambar 1.1.



Gambar 1.1 Model Class

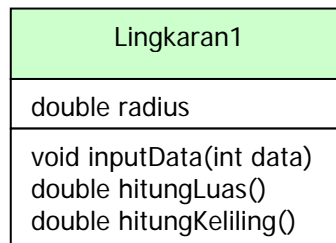
Class merupakan pendefinisian suatu bentuk obyek dengan menyebutkan definisi member dan method dari obyek tersebut. Bisa dikatakan bahwa class merupakan suatu tipe data dari obyek, sedangkan obyek adalah variabel yang menggunakan tipe data yang didefinisikan dalam class.

Misalnya kita ingin membuat program untuk menghitung luas dan keliling lingkaran maka kita dapat membuat sebuah class mengenai

lingkaran dengan nama **Lingkaran1.java**. Seperti yang telah diketahui bahwa lingkaran mempunyai parameter berupa jari-jari dan perhitungan yang bisa dilakukan adalah luas lingkaran dan keliling lingkaran. Sehingga class lingkaran1.java memiliki variabel member **radius** dan beberapa method seperti

- **void inputData(int data),**
- **double hitungLuas(),**
- **double hitungKeliling()** dan
- **void main(String args[]).**

Class tersebut dapat dinyatakan dengan model seperti pada gambar 1.2.



Gambar 1.2. Model class untuk lingkaran

Dari Class Diagram tersebut di atas dapat dituliskan program Java sebagai berikut :

```
class Lingkaran1
{
    double radius;

    public void inputData(int data)
    {
        radius=data;
    }

    public double hitungLuas()
    {
        double Luas;
        Luas=Math.PI*radius*radius;
        return Luas;
    }

    public double hitungKeliling()
    {
```

```

        double Keliling;
        Keliling=2*Math.PI*radius;
        return Keliling;
    }

    public void cetak()
    {
        System.out.println("Radius Lingkaran : "+radius);
        System.out.println("Luas Lingkaran : "+hitungLuas());
        System.out.println("Keliling Lingkaran : "+hitungKeliling());
    }

    public static void main(String args[])
    {
        Lingkaran1 ling1=new Lingkaran1();
        ling1.inputData(10);
        ling1.cetak();
    }
}

```

**Hasil dari program di atas adalah:**

```

Radius Lingkaran : 10.0
Luas Lingkaran : 314.1592653589793
Keliling Lingkaran : 62.83185307179586

```

## 1.2. Inheritance

*Inheritance* atau pewarisan sifat adalah sebuah class yang memiliki properti (*variabel member* dan *method*) dari class induknya. Dengan inheritance ini maka suatu class anak akan mempunyai semua member dan semua method yang ada di class induk.

Berikut ini adalah sebuah contoh penggunaan inheritance dimana class **Lingkaran1.java** di atas merupakan class induk dan class **Tabung.java** merupakan class turunannya. Untuk menyatakan suatu class sebagai turunan class yang lain dapat dilakukan dengan menambahkan pernyataan extends pada definisi class tersebut.

**class classAnak extends classInduk**

Pernyataan ini digunakan untuk membuat suatu class yang bernama classAnak, class ini menjadi class turunan dari classInduk. Dengan demikian semua member dan method dari classInduk akan berlaku juga pada classAnak.

```
class Tabung extends Lingkaran1
{
    public double hitungSelimut(int tinggi)
    {
        double Selimut;
        Selimut=2*hitungLuas()+hitungKeliling()*tinggi;
        return Selimut;
    }

    public static void main(String args[])
    {
        Tabung t1=new Tabung();
        t1.inputData(10);
        t1.cetak();
        System.out.println("Luas Selimut : "+
            t1.hitungSelimut(10));
    }
}
```

**Hasil dari program di atas adalah:**

```
Radius Lingkaran : 10.0
Luas Lingkaran : 314.1592653589793
Keliling Lingkaran : 62.83185307179586
Luas Selimut : 1256.6370614359173
```

Kata kunci *extends* pada class Tabung.java merupakan kata yang menunjukkan bahwa class Tabung.java merupakan turunan dari Lingkaran1.java. Dengan demikian class Tabung memiliki variabel member radius, method hitungLuas() dan method hitungKeliling() yang dimiliki oleh class Lingkaran.java. Oleh karenanya variabel member dan method tersebut dapat kita gunakan untuk perhitungan dalam method hitungSelimut(int tinggi) meskipun variabel member dan .method tersebut tidak ada dalam class Tabung.java.

### 1.3. Constructor

*Constructor* adalah method yang memiliki nama sama dengan nama classnya. Berfungsi sebagai inisialisasi ketika sebuah obyek dibuat dari sebuah class. Class **Lingkaran2.java** berikut ini adalah class yang memiliki constructor yang diperoleh dengan melakukan modifikasi terhadap method inputData(int data) pada class **Lingkaran1.java**.

```
class Lingkaran2
{
    double radius;

    Lingkaran2(int data)
    {
        radius=data;
    }

    public double hitungLuas()
    {
        double Luas;
        Luas=Math.PI*radius*radius;
        return Luas;
    }

    public double hitungKeliling()
    {
        double Keliling;
        Keliling=2*Math.PI*radius;
        return Keliling;
    }

    public void cetak()
    {
        System.out.println("Radius Lingkaran : "+radius);
        System.out.println("Luas Lingkaran : "+hitungLuas());
        System.out.println("Keliling Lingkaran : "+hitungKeliling());
    }

    public static void main(String args[])
    {
        Lingkaran2 ling2=new Lingkaran2(10);
        ling2.cetak();
    }
}
```

**Hasil dari program di atas adalah:**

```
Radius Lingkaran : 10.0  
Luas Lingkaran : 314.1592653589793  
Keliling Lingkaran : 62.83185307179586
```

Perbedaan program Lingkaran1.java dengan Lingkaran2.java adalah pada pemberian argumen untuk variabel member radius. Pada class Lingkaran1.java terdapat sintaks :

```
public static void main(String args[])  
{  
    Lingkaran1 ling1=new Lingkaran1();  
    ling1.inputData(10);  
    ling1.cetak();  
}
```

Dimana nilai dari variabel radius dimasukkan melalui method inputData(10). Pada class Lingkaran2.java nilai dari radius langsung dimasukkan ketika obyek ling2 dibuat menggunakan sintaks:

```
public static void main(String args[])  
{  
    Lingkaran2 ling2=new Lingkaran2(10);  
    ling2.cetak();  
}
```

## BAB 2

### KOMPONEN-KOMPONEN VISUAL

---

**Tujuan:**

1. *Mahasiswa dapat membangun komponen visual menggunakan bahasa Java.*
  2. *Mahasiswa dapat menerapkan komponen-komponen visual tersebut ke dalam contoh-contoh sederhana*
- 

Komponen visual adalah berbagai objek yang digunakan dalam membangun program berbasis windows. Dengan komponen-komponen ini diharapkan pengguna akhir dapat menggunakan program yang kita buat dengan lebih mudah dan nyaman. Dalam bab ini tidak semua komponen visual dibahas, hanya komponen yang dianggap sering digunakan saja yang ditampilkan sebagai bahan eksplorasi lebih lanjut bagi anda terhadap berbagai komponen java.

#### 2.1. JFRAME

JFrame adalah komponen dasar dalam pemrograman visual dengan java. Dalam frame inilah komponen lain diletakkan. Berikut adalah kode program yang diperlukan untuk membuat sebuah frame kosong. Program ini menggunakan constructor tunggal tanpa argumen yang digunakan untuk menampung properti frame yang dikehendaki.

```
import javax.swing.*;

class AplikasiPenilaian extends JFrame
{
    AplikasiPenilaian()
    {
        setTitle("Lembar Penilaian");
    }
}
```



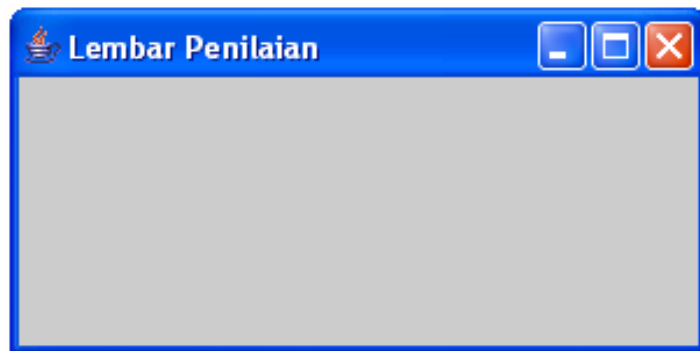
```

        setLocation(300,100);
        setSize(300,150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiPenilaian ap=new AplikasiPenilaian();
    }
}

```

Hasil dari program di atas adalah :



Gambar 2.1. Frame kosong

Penjelasan kode program untuk membuat fram kosong di atas adalah sebagai berikut:

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

Pernyataaan ini merupakan perintah yang digunakan untuk menyiapkan class-class yang diperlukan oleh program kita. Komponen visual yang kita perlukan dalam pemrograman visual berada dalam paket javax.swing.\* ini.

```
class AplikasiPenilaian extends JFrame
```

Pernyataan ini menunjukkan bahwa class AplikasiPenilaian merupakan turunan dari class JFrame yang berarti bahwa class ini memiliki atribut-atribut dari class JFrame.

```
AplikasiPenilaian()
{
    setTitle("Lembar Penilaian");
    setLocation(300,100);
    setSize(300,150);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setVisible(true);
}
```

merupakan constructor bagi class AplikasiPenilaian sehingga pada saat kita membentuk sebuah obyek, semua kode di dalamnya akan dilaksanakan.

- setTitle digunakan untuk membuat judul frame.
- setLocation(300,100) digunakan untuk menentukan posisi frame di layar dimana 300 menunjukkan posisi x dan 100 menunjukkan posisi y,
- setSize(300,150) digunakan untuk menentukan besar frame dimana 300 menunjukkan lebar frame dan 150 menunjukkan tinggi frame.
- setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE) digunakan untuk mengakhiri jalannya program bila frame ditutup. Bila perintah tersebut tidak ada, maka penutupan frame tidak akan menghentikan jalannya program.
- setVisible(true) digunakan untuk menampilkan frame.

```
public static void main(String args[])
```

merupakan metode utama agar program dapat dijalankan.

```
AplikasiPenilaian ap=new AplikasiPenilaian();
```

Membentuk obyek dari class AplikasiPenilaian, yang secara otomatis menjalankan constructor yang berisi sintaks-sintaks untuk membuat frame kosong.

## 2.2. JLabel

JLabel adalah komponen yang digunakan untuk membuat tulisan atau gambar pada frame sebagai suatu informasi untuk pengguna program. Untuk menggunakan JLabel, sebagai suatu class maka sebelumnya perlu dibuat suatu obyek menggunakan class JLabel. Program berikut akan menambahkan komponen JLabel pada program **AplikasiPenilaian** seperti yang dijelaskan di atas (catatan: penulisan huruf tebal pada listing program adalah modifikasi untuk menambahkan komponen JLabel).

```
import javax.swing.*;

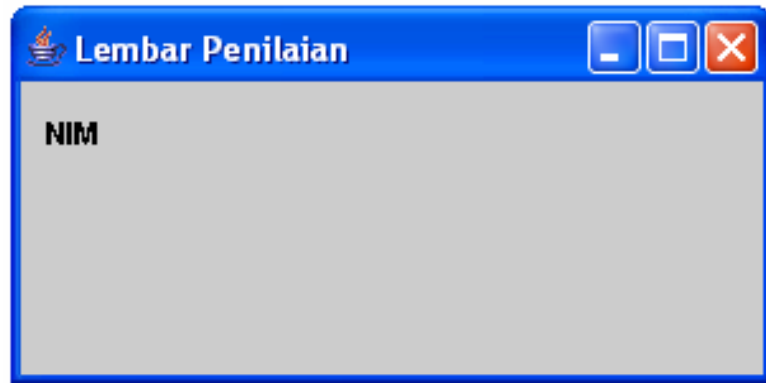
class AplikasiPenilaian extends JFrame
{
    JLabel lblnim=new JLabel("NIM");

    AplikasiPenilaian()
    {
        setTitle("Lembar Penilaian");
        setLocation(300,100);
        setSize(300,150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void komponenVisual()
    {
        getContentPane().setLayout(null);
        getContentPane().add(lblnim);
        lblnim.setBounds(10,10,70,20);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiPenilaian ap=new AplikasiPenilaian();
        ap.komponenVisual();
    }
}
```

Hasil dari program di atas adalah :



Gambar 2.2. Label pada frame

Bagian yang dicetak tebal merupakan perubahan dan penambahan yang perlu dilakukan untuk memperoleh Hasil dari program di atas adalah: di atas. Penjelasannya adalah sebagai berikut :

```
JLabel lblnim=new JLabel("NIM");
```

Merupakan cara untuk membentuk obyek JLabel yang kita beri nama lblnim dan bertuliskan "NIM". Obyek ini akan kita gunakan sebagai informasi kepada user untuk memasukkan NIM mahasiswa.

```
void komponenVisual()
{
    getContentPane().setLayout(null);
    getContentPane().add(lblnim);
    lblnim.setBounds(10,10,70,20);
    setVisible(true);
}
```

method void komponenVisual() digunakan untuk meletakkan berbagai komponen visual yang kita gunakan dalam program.

```
getContentPane().setLayout(null)
```

Sintaks di atas digunakan untuk mengatur tata letak komponen dalam frame, dimana layout null berarti bahwa koordinat tiap komponen dalam frame harus ditentukan sendiri posisinya oleh programmer.

```
getContentPane().add(lblnim)
```

adalah perintah yang digunakan untuk menempelkan obyek JLabel ke frame.

```
lblnim.setBounds(10,10,70,20);
```

perintah di atas berhubungan dengan getContentPane().setLayout(null) di atas yang berguna untuk mengatur posisi dari komponen JLabel yang bernama lblnim. Pengaturan posisi dilakukan dengan menggunakan setBounds(10,10,70,20) dimana argumen pertama menunjukkan koordinat x dari ujung kiri atas obyek, argumen kedua menunjukkan koordinat y dari ujung kiri atas obyek, argumen ketiga menunjukkan lebar dari obyek dan argumen keempat menunjukkan tinggi dari obyek.

```
setVisible(true);
```

digunakan untuk menampilkan frame beserta semua obyek yang ada di dalamnya. Pada program sebelumnya method ini kita letakkan pada constructor karena kita belum membuat method komponenVisual().

Pada main, kita perlu memanggil method komponenVisual() agar semua semua pengaturan yang telah kita lakukan dapat tampil ke layar, dengan cara:

**ap.komponenVisual();**

Hasil dari program di atas adalah: program adalah tampak seperti di atas, dimana frame awal yang tadinya masih kosong, sekarang telah kita beri label NIM. Selanjutnya kita perlu menambahkan komponen JTextField agar kita dapat memasukkan data ke dalam program.

## 2.3. JTEXTFIELD

JTextField adalah komponen yang digunakan untuk memasukkan sebaris string yang selanjutnya dapat digunakan sebagai input bagi proses selanjutnya. Pembuatan JTextField dilakukan dengan membuat obyek berdasarkan class JTextField, seperti terlihat pada program berikut. Perhatikan tulisan tebal menyatakan pemakaian JTextField.

```
import javax.swing.*;

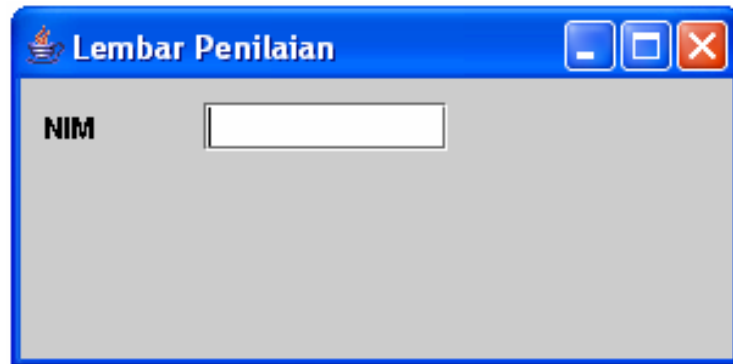
class AplikasiPenilaian extends JFrame
{
    JLabel lblnim=new JLabel("NIM ");
    JTextField txnim=new JTextField(20);

    AplikasiPenilaian()
    {
        setTitle("Lembar Penilaian");
        setLocation(300,100);
        setSize(300,150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void komponenVisual()
    {
        getContentPane().setLayout(null);
        getContentPane().add(lblnim);
        lblnim.setBounds(10,10,70,20);
        getContentPane().add(txnim);
        txnim.setBounds(75,10,100,20);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiPenilaian ap=new AplikasiPenilaian();
        ap.komponenVisual();
    }
}
```

Hasil dari program di atas adalah :



Gambar 2.3. Hasil Jtextfield

Baris kode yang dicetak tebal adalah perubahan dan penambahan yang dilakukan terhadap program sebelumnya. Baris perintah :

```
TextField txnim=new TextField(20);
```

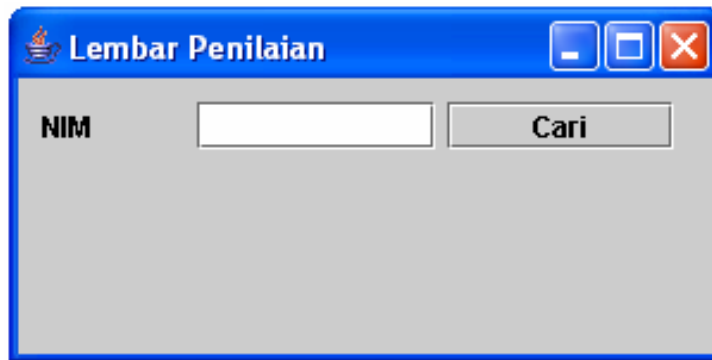
Adalah membentuk obyek dari komponen `TextField` yang akan digunakan untuk menampung nilai dari NIM mahasiswa.

```
getContentPane().add(txnim);  
txnim.setBounds(75,10,100,20);
```

adalah perintah untuk menempelkan dan mengatur posisi `txnim` pada frame.

## 2.4. JBUTTON

`JButton` adalah komponen berbentuk tombol. Komponen ini banyak digunakan sebagai eksekusi terhadap tindakan yang diinginkan. Pada aplikasi komputer, biasanya dibutuhkan tombol untuk mengeksekusi sebuah perintah. Misalnya program di atas kita tambahkan sebuah tombol untuk mencari data mahasiswa sehingga tampilannya akan tampak seperti berikut:



Gambar 2.4. Hasil JButton

Maka program sebelumnya dapat kita tambah (tercetak tebal) untuk memasukkan komponen JButton dalam frame seperti berikut:

```
import javax.swing.*;

class AplikasiPenilaian extends JFrame
{
    JLabel lblnim=new JLabel("NIM ");
    JTextField txnim=new JTextField(20);
    JButton tblcari=new JButton("Cari");

    AplikasiPenilaian()
    {
        setTitle("Lembar Penilaian");
        setLocation(300,100);
        setSize(300,150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void komponenVisual()
    {
        getContentPane().setLayout(null);
        getContentPane().add(lblnim);
        lblnim.setBounds(10,10,70,20);
        getContentPane().add(txnim);
        txnim.setBounds(75,10,100,20);
        getContentPane().add(tblcari);
        tblcari.setBounds(180,10,95,20);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiPenilaian app = new AplikasiPenilaian();
        app.setVisible(true);
    }
}
```



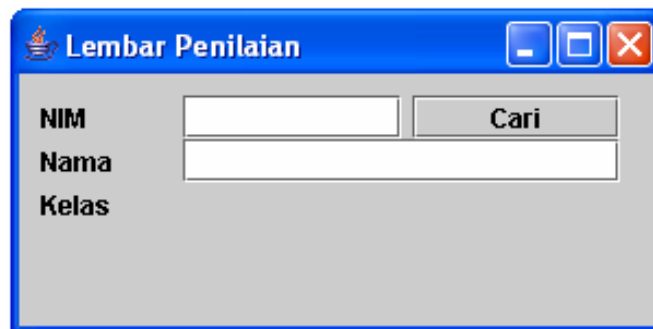
```

{
    AplikasiPenilaian ap=new AplikasiPenilaian();
    ap.komponenVisual();
}
}

```

### **Latihan 2.1**

Sampai disini, tambahkan sendiri komponen yang diperlukan agar tampilan aplikasi tampak seperti berikut :



Gambar 2.5. Contoh form latihan

Bila tampilan telah sesuai, untuk pembahasan berikutnya adalah bagaimana cara menambahkan komponen `JRadioButton` untuk memilih kelas dari masing-masing mahasiswa.

## **2.5. JRADIOBUTTON**

`JRadioButton` adalah komponen yang digunakan ketika pengguna perlu memilih satu diantara beberapa pilihan. Program untuk menentukan kelas mahasiswa adalah sebagai berikut :

```

import javax.swing.*;

class AplikasiPenilaian extends JFrame
{
    JLabel lblnim=new JLabel("NIM ");
    JTextField txnim=new JTextField(20);
    JButton tblcari=new JButton("Cari");
    JLabel lblnama=new JLabel("Nama");
    JTextField txnama=new JTextField(20);
    JLabel lblkelas=new JLabel("Kelas ");
    JRadioButton kelasA=new JRadioButton("A");
    JRadioButton kelasB=new JRadioButton("B");

```

```

JRadioButton kelasC=new JRadioButton("C");
ButtonGroup grupkelas=new ButtonGroup();

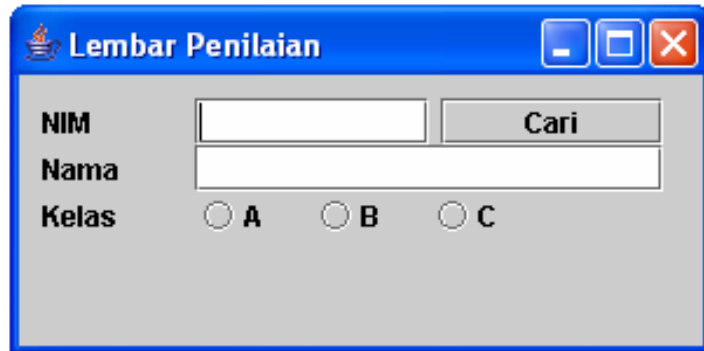
AplikasiPenilaian()
{
    setTitle("Lembar Penilaian");
    setLocation(300,100);
    setSize(300,150);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

void komponenVisual()
{
    getContentPane().setLayout(null);
    getContentPane().add(lblnim);
    lblnim.setBounds(10,10,70,20);
    getContentPane().add(txnim);
    txnim.setBounds(75,10,100,20);
    getContentPane().add(tblcari);
    tblcari.setBounds(180,10,95,20);
    getContentPane().add(lblnama);
    lblnama.setBounds(10,30,70,20);
    getContentPane().add(txnama);
    txnama.setBounds(75,30,200,20);
    getContentPane().add(lblkelas);
    lblkelas.setBounds(10,50,100,20);
    getContentPane().add(kelasA);
    kelasA.setBounds(75,50,50,20);
    getContentPane().add(kelasB);
    kelasB.setBounds(125,50,50,20);
    getContentPane().add(kelasC);
    kelasC.setBounds(175,50,50,20);
    grupkelas.add(kelasA);
    grupkelas.add(kelasB);
    grupkelas.add(kelasC);
    setVisible(true);
}

public static void main(String args[])
{
    AplikasiPenilaian ap=new AplikasiPenilaian();
    ap.komponenVisual();
}
}

```

Hasil dari program di atas adalah :



Gambar 2.6. Contoh JRadioButton

Bagian yang dicetak tebal merupakan kode yang perlu kita tambahkan dari proram sebelumnya untuk menggunakan komponen JRadioButton. Penjelasan komponen JradioButton adalah sebagai berikut:

```
JRadioButton kelasA=new JRadioButton("A");  
JRadioButton kelasB=new JRadioButton("B");  
JRadioButton kelasC=new JRadioButton("C");
```

Sintaks di atas menunjukkan bahwa kita memiliki tiga kelas yang dibuat dengan membentuk tiga buah obyek dari class JRadioButton.

```
ButtonGroup grupkelas=new ButtonGroup();
```

Merupakan obyek yang digunakan untuk menyatukan semua obyek JRadioButton, sehingga hanya akan terpilih satu diantara obyek JRadioButton yang ada.

```
getContentPane().add(kelasA);  
kelasA.setBounds(75,50,50,20);  
getContentPane().add(kelasB);  
kelasB.setBounds(125,50,50,20);  
getContentPane().add(kelasC);  
kelasC.setBounds(175,50,50,20);
```

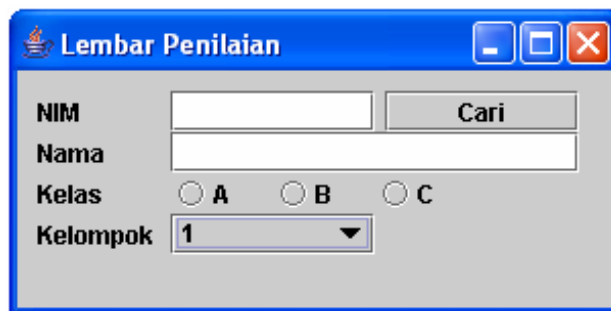
digunakan untuk menambahkan dan mengatur posisi masing-masing obyek dalam frame.

```
grupkelas.add(kelasA);  
grupkelas.add(kelasB);  
grupkelas.add(kelasC);
```

adalah cara untuk menyatukan ketiga obyek menjadi satu kesatuan, sehingga hanya dapat dipilih satu diantara ketiganya.

## 2.6. JCOMBOBOX

JComboBox juga merupakan komponen yang digunakan untuk memilih satu diantara sekian banyak pilihan yang berbentuk semacam TextField dan ada panah ke bawah. Bila program kita dikembangkan agar diperoleh hasil berikut:



Gambar 2.6. Contoh JComboBox

Dimana terdapat komponen ComboBox yang digunakan untuk memilih seorang mahasiswa tergabung dalam kelompok berapa, maka programnya adalah sebagai berikut:

```
import javax.swing.*;  
  
class AplikasiPenilaian extends JFrame  
{  
    JLabel lblnim=new JLabel("NIM ");  
    JTextField txnim=new JTextField(20);  
    JLabel lblnama=new JLabel("Nama");  
    JTextField txnama=new JTextField(20);  
    JButton tblcari=new JButton("Cari");  
    JLabel lblkelas=new JLabel("Kelas ");  
    JRadioButton kelasA=new JRadioButton("A");  
    JRadioButton kelasB=new JRadioButton("B");  
    JRadioButton kelasC=new JRadioButton("C");  
    ButtonGroup grupkelas=new ButtonGroup();
```

```

JLabel lblkelompok=new JLabel("Kelompok");
String[] jeniskelompok={"1","2","3","4","5","6","7"};
JComboBox cbkelompok=new JComboBox(jeniskelompok);

```

```

AplikasiPenilaian()
{
    setTitle("Lembar Penilaian");
    setLocation(300,100);
    setSize(300,150);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

```

```

void komponenVisual()
{
    getContentPane().setLayout(null);
    getContentPane().add(lblnim);
    lblnim.setBounds(10,10,70,20);
    getContentPane().add(txnim);
    txnim.setBounds(75,10,100,20);
    getContentPane().add(tblcari);
    tblcari.setBounds(180,10,95,20);
    getContentPane().add(lblnama);
    lblnama.setBounds(10,30,70,20);
    getContentPane().add(txnama);
    txnama.setBounds(75,30,200,20);
    getContentPane().add(lblkelas);
    lblkelas.setBounds(10,50,100,20);
    getContentPane().add(kelasA);
    kelasA.setBounds(75,50,50,20);
    getContentPane().add(kelasB);
    kelasB.setBounds(125,50,50,20);
    getContentPane().add(kelasC);
    kelasC.setBounds(175,50,50,20);
    grupkelas.add(kelasA);
    grupkelas.add(kelasB);
    grupkelas.add(kelasC);
    getContentPane().add(lblkelompok);
    lblkelompok.setBounds(10,70,100,20);
    getContentPane().add(cbkelompok);
    cbkelompok.setBounds(75,70,100,20);
    setVisible(true);
}

```

```

public static void main(String args[])
{

```

```

    AplikasiPenilaian ap=new AplikasiPenilaian();
    ap.komponenVisual();
}
}

```

Penjelasan sintaks di atas adalah sebagai berikut:

```
String[] jeniskelompok={"1","2","3","4","5","6","7"};
```

Adalah variabel array yang akan digunakan sebagai alternatif pilihan bagi ComboBox.

```
JComboBox cbkelompok=new JComboBox(jeniskelompok);
```

Merupakan deklarasi pembentukan obyek ComboBox menggunakan argumen array yang sebelumnya telah disediakan yaitu jeniskelompok.

```

getContentPane().add(cbkelompok);
cbkelompok.setBounds(75,70,100,20);

```

adalah cara yang sudah biasa kita gunakan untuk memasang komponen pada frame.

## **Latihan 2.2**

Buatlah aplikasi untuk menghasilkan form sebagai berikut :

Gambar 2.7. Contoh hasil lembar penilaian

## 2.7. JTextArea

JTextArea merupakan komponen yang mirip dengan JTextField tetapi dapat menampung lebih dari satu baris. Sebagai contoh penggunaan dari JTextArea adalah seperti pada program berikut:

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

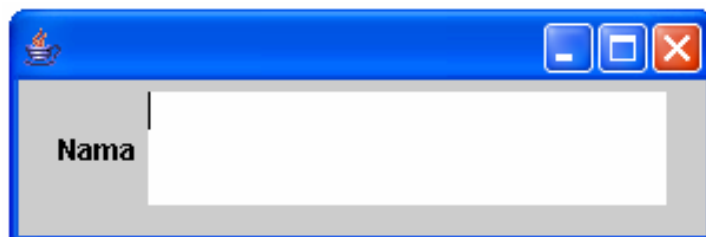
class FrameTextArea extends JFrame
{
    private JLabel label=new JLabel("Nama");
    private JTextArea area1=new JTextArea(3,20);

    FrameTextArea()
    {
        setLocation(200,100);
        setSize(300,100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        getContentPane().add(label);
        getContentPane().add(area1);
        getContentPane().setLayout(new FlowLayout());
        show();
    }
}

class AplikasiTextArea
{
    public static void main(String args[])
    {
        FrameTextArea tf=new FrameTextArea();
    }
}
```

Hasil dari program di atas adalah: berikut:



Gambar 2.8. Contoh JTextArea

## 2.8. JCheckBox

JCheckBox adalah komponen yang digunakan ketika pengguna memerlukan komponen untuk melakukan satu atau banyak pilhan sekaligus. Contoh penggunaan JCheckBox adalah sebagai berikut:

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

class FrameCheckBox extends JFrame
{
    JCheckBox cek1=new JCheckBox("Pilihan 1");
    JCheckBox cek2=new JCheckBox("Pilihan 2");
    JCheckBox cek3=new JCheckBox("Pilihan 3");

    FrameCheckBox()
    {
        setLocation(200,100);
        setSize(300,100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        getContentPane().add(cek1);
        getContentPane().add(cek2);
        getContentPane().add(cek3);
        getContentPane().setLayout(new FlowLayout());
        setVisible(true);
    }
}

class AplikasiCheckBox
{
    public static void main(String args[])
    {
        FrameCheckBox cb=new FrameCheckBox();
        cb.KomponenVisual();
    }
}
```



Hasil dari program di atas adalah: berikut:



Gambar 2.9. Contoh JCheckBox

## 2.9. JTable

JTable digunakan untuk menampilkan data dalam bentuk tabel, suatu bentuk yang banyak digunakan dalam pemrograman database. Contoh penggunaan JTable seperti listing program berikut.

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

class FrameTabel extends JFrame
{
    String[] header={"Senin","Selasa","Rabu"};
    String[][] data =
        {"100","300","150"}, {"500","600","450"}, {"290","690","360"};

    JTable tabel1=new JTable(data,header);

    FrameTabel()
    {
        setLocation(200,100);
        setSize(300,100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        getContentPane().add(tabel1);
        setVisible(true);
    }
}

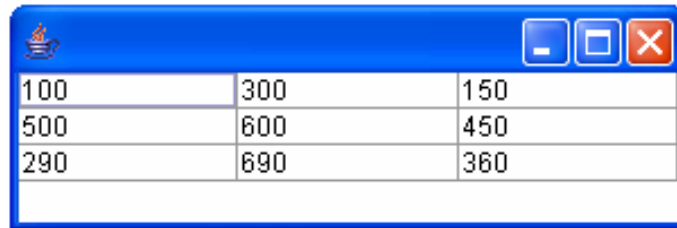
class AplikasiTabel
{
    public static void main(String args[])
```

```

{
    FrameTabel t=new FrameTabel();
    t.KomponenVisual();
}
}

```

Hasil dari program di atas adalah: berikut:



100	300	150
500	600	450
290	690	360

Gambar 2.10. Contoh JTable

## 2.10. JScrollPane

JScrollPane adalah komponen yang digunakan untuk menggerakkan obyek ke atas, ke bawah atau ke samping agar semua sebuah obyek terlihat di layar. Contoh penggunaan JScrollPane adalah sebagai berikut.

```

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

class FrameScrollPane2 extends JFrame
{
    String[] header={"Senin","Selasa","Rabu"};
    String[][] data={{ "100","300","150"}, {"500","600","450"}, {"290","690","360"};

    FrameScrollPane2()
    {
        setLocation(200,100);
        setSize(300,100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        JTable tabel1=new JTable(data,header);
        JScrollPane scrollPane = new JScrollPane(tabel1);
        getContentPane().add(scrollPane, BorderLayout.CENTER);
    }
}

```

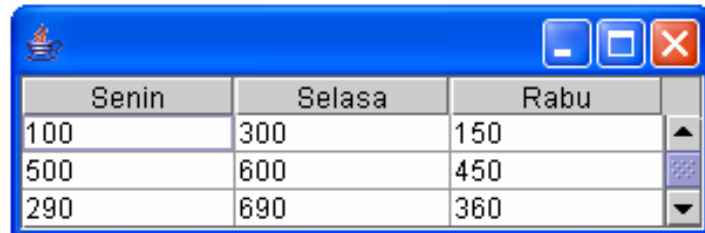
```

        setVisible(true);
    }
}

class AplikasiScrollPane2
{
    public static void main(String args[])
    {
        FrameScrollPane2 t=new FrameScrollPane2();
        t.KomponenVisual();
    }
}

```

Hasil dari program di atas adalah: berikut:



Senin	Selasa	Rabu
100	300	150
500	600	450
290	690	360

Gambar 2.11. Contoh JScrollPane

## 2.11. JMenu

JMenu adalah komponen yang digunakan untuk membuat menu. Menu membuat program kita menjadi lebih sederhana dan mudah digunakan. Contoh penggunaan JMenu adalah sebagai berikut.

```

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

class AplikasiMenu extends JFrame
{
    JMenuBar mb=new JMenuBar();
    JMenu file=new JMenu("File");
    JMenu help=new JMenu("Help");
    JMenuItem open=new JMenuItem("Open");
    JMenuItem close=new JMenuItem("Close");
    JMenuItem quit=new JMenuItem("Quit");
    JMenuItem about=new JMenuItem("About");
}

```

```

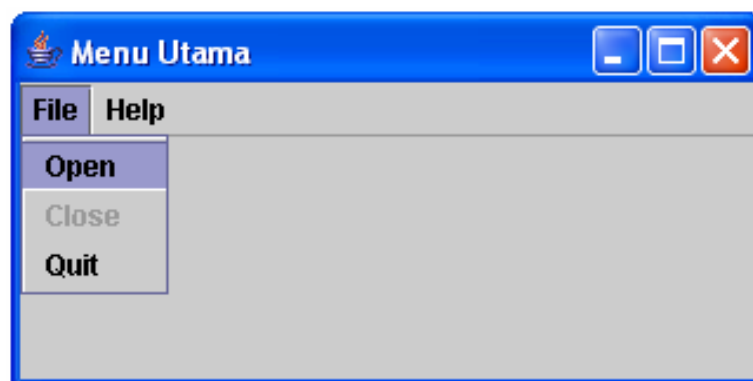
AplikasiMenu()
{
    setTitle("Menu Utama");
    setSize(320,160);
    setLocation(300,200);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

void KomponenVisual()
{
    setJMenuBar(mb);
    mb.add(file);
    mb.add(help);
    file.add(open);
    file.add(close);
    close.setEnabled(false);
    file.add(quit);
    help.add(about);
    setVisible(true);
}

public static void main(String args[])
{
    AplikasiMenu m1=new AplikasiMenu();
    m1.KomponenVisual();
}
}

```

Hasil dari program di atas adalah: berikut:



Gambar 2.12. Contoh JMenu

## 2.12. JInternalFrame

JInternalFrame menyebabkan sebuah frame hanya dapat berada dalam frame lain. Kondisi ini akan membantu tampilan menjadi lebih rapi dan teratur. Contoh penggunaan JInternalFrame adalah sebagai berikut.

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

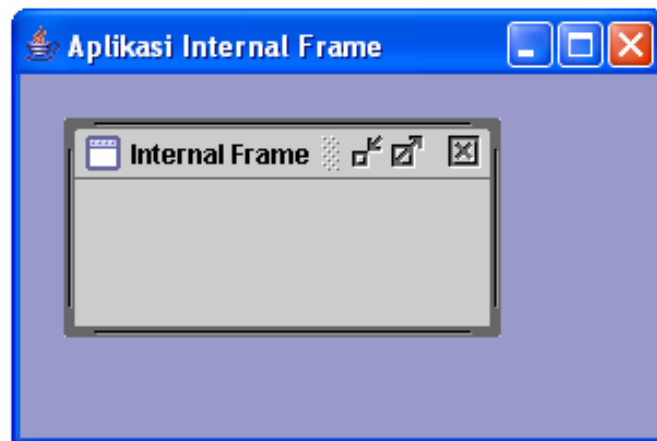
public class AplikasiFrameInternal extends JFrame
{
    JDesktopPane desktop=new JDesktopPane();
    JInternalFrame iframe=new
        JInternalFrame("Internal Frame",true,true,true,true);

    AplikasiFrameInternal()
    {
        setTitle("Aplikasi Internal Frame");
        setLocation(100,200);
        setSize(300,200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void komponenVisual()
    {
        iframe.setLocation(20,20);
        iframe.setSize(200,100);
        iframe.setVisible(true);
        desktop.add(iframe);
        setContentPane(desktop);
        setVisible(true);
    }

    public static void main(String[] args)
    {
        AplikasiFrameInternal if1=new AplikasiFrameInternal();
        if1.komponenVisual();
    }
}
```

Hasil dari program di atas adalah: berikut:



Gambar 2.13. COntoh JInternalFrame

## BAB 3

### EVENT HANDLER

---

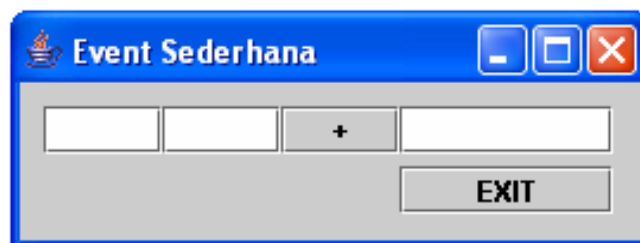
**Tujuan:**

1. Mahasiswa dapat menjelaskan konsep even handler pada pemrograman bahasa Java.
  2. Mahasiswa dapat membuat program sederhana untuk even handler pada komponen-komponen dasar dengan contoh-contoh yang mudah dipahami
  3. Mahasiswa dapat membuat program menu menggunakan bahasa Java
- 

*Event Handler* adalah proses yang diperlukan untuk melakukan reaksi bila diberikan sebuah aksi. *Event handler* ini dapat diberikan pada semua obyek yang kita gunakan bergantung pada keperluan. Berikut ini adalah beberapa contoh penerapan *event handler* sebagai bahan bagi anda untuk melakukan eksplorasi lebih lanjut.

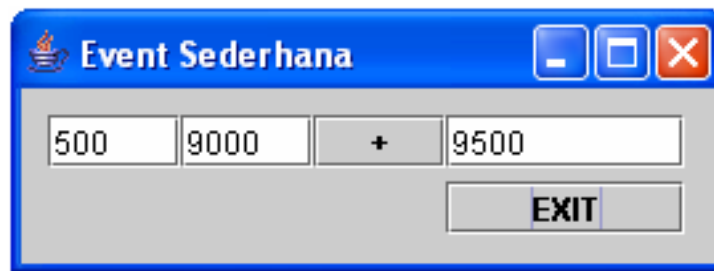
#### 3.1. Event Handler Untuk Menangani Klik Mouse Pada JButton

Misalnya kita memiliki sebuah aplikasi sederhana untuk menjumlahkan dua buah bilangan dengan tampilan sebagai berikut :



Gambar 3.1. Contoh event sederhana pada button plus  
Setelah program dijalankan, kita dapat memasukkan data pada TextField yang tersedia dan melakukan klik pada button plus untuk mendapatkan

hasil. Keluar dari aplikasi dapat dilakukan dengan klik pada tombol exit atau klik pada icon close.



Gambar 3.2. Contoh pengisian pada event

Kode program untuk aplikasi tersebut adalah sebagai berikut :

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

class AplikasiEvent1 extends JFrame
{
    JTextField data1=new JTextField(6);
    JTextField data2=new JTextField(6);
    JButton operasi=new JButton("+");
    JTextField hasil=new JTextField(6);
    JButton exit=new JButton("EXIT");

    AplikasiEvent1()
    {
        setTitle("Event Sederhana");
        setLocation(200,100);
        setSize(270,100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void komponenVisual()
    {
        getContentPane().setLayout(null);
        getContentPane().add(data1);
        data1.setBounds(10,10,50,20);
        getContentPane().add(data2);
        data2.setBounds(60,10,50,20);
        getContentPane().add(operasi);
        operasi.setBounds(110,10,50,20);
        getContentPane().add(hasil);
        hasil.setBounds(160,10,90,20);
    }
}
```



```

        getContentPane().add(exit);
        exit.setBounds(160,35,90,20);
        setVisible(true);
    }

    void AksiReaksi()
    {
        operasi.addActionListener(new ActionListener()
        {
            public void actionPerformed(ActionEvent e)
            {
                int x=Integer.parseInt(data1.getText());
                int y=Integer.parseInt(data2.getText());
                String z=String.valueOf(x+y);
                hasil.setText(z);
            }
        });

        exit.addActionListener(new ActionListener()
        {
            public void actionPerformed(ActionEvent e)
            {
                System.exit(0);
            }
        });
    }

    public static void main(String args[])
    {
        AplikasiEvent1 e1=new AplikasiEvent1();
        e1.komponenVisual();
        e1.AksiReaksi();
    }
}

```

Penjelasan program di atas untuk kode-kode yang belum pernah dijelaskan di depan:

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

Adalah tempat dari class-class yang diperlukan untuk memberikan event pada komponen visual.

```

JTextField data1=new JTextField(6);
JTextField data2=new JTextField(6);
JButton operasi=new JButton("+");
JTextField hasil=new JTextField(6);
JButton exit=new JButton("EXIT");

```

Merupakan deklarasi dari obyek-obyek yang akan digunakan dalam aplikasi, terdiri dari JTextField dan JButton.

```

AplikasiEvent1()
{
    setTitle("Event Sederhana");
    setLocation(200,100);
    setSize(270,100);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

```

adalah constructor tanpa argumen dari class AplikasiEvent1.

```

void komponenVisual()
{
    getContentPane().setLayout(null);
    getContentPane().add(data1);
    data1.setBounds(10,10,50,20);
    getContentPane().add(data2);
    data2.setBounds(60,10,50,20);
    getContentPane().add(operasi);
    operasi.setBounds(110,10,50,20);
    getContentPane().add(hasil);
    hasil.setBounds(160,10,90,20);
    getContentPane().add(exit);
    exit.setBounds(160,35,90,20);
    setVisible(true);
}

```

Merupakan method yang digunakan untuk mengatur posisi masing-masing komponen visual dalam frame.

```

void AksiReaksi()
{
    operasi.addActionListener(new ActionListener()
    {

```

```

    public void actionPerformed(ActionEvent e)
    {
        int x=Integer.parseInt(data1.getText());
        int y=Integer.parseInt(data2.getText());
        String z=String.valueOf(x+y);
        hasil.setText(z);
    }
});

exit.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        System.exit(0);
    }
});
}

```

method AksiReaksi() inilah yang akan kita gunakan untuk menempatkan kode-kode program event handler. Pada contoh di atas, terdapat dua event handler yaitu penekanan mouse pada tombol operasi dan penekanan mouse pada tombol exit. Reaksi yang timbul akibat klik mouse pada tombol operasi dan exit diimplementasikan dengan kode berikut :

```

operasi.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        ...
        ...
    }
});

exit.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        ...
        ...
    }
});

```

Dalam method public void actionPerformed(ActionEvent e) inilah efek yang timbul akibat penekanan klik mouse pada tombol operasi dan exit kita tuliskan.

```
int x=Integer.parseInt(data1.getText());  
int y=Integer.parseInt(data2.getText());
```

Dua baris kode tersebut di atas mempunyai tugas yang sama yaitu mengambil nilai dari JTextField dengan cara `data1.getText()` , mengubahnya dari String menjadi integer dengan `Integer.parseInt` dan menyimpan nilainya ke dalam variabel `x` dan `y`.

```
String z=String.valueOf(x+y);
```

Baris perintah di atas digunakan untuk melakukan proses penjumlahan kedua input data `x` dan `y`, kemudian mengubah tipe datanya menjadi String agar hasilnya dapat ditampilkan melalui JTextField hasil dengan menggunakan sintaks :

```
hasil.setText(z);
```

Pada tombol exit diberikan sintaks `System.exit(0)` agar aplikasi berhenti bila tombol exit di klik.

### **3.2. Event Handler Untuk Menangani Penekanan Enter Pada Keyboard**

Kadang pengguna lebih menyukai penekanan tombol enter dibanding menggunakan tombol klik mouse, program di atas dapat ditambahkan event handler agar operasi penjumlahan dapat dilakukan dengan klik mouse maupun penekanan tombol enter pada keyboard. Kode program yang perlu ditambahkan adalah:

```
operasi.addKeyListener(new KeyAdapter()  
{  
    public void keyPressed(KeyEvent e)  
    {  
        if(e.getKeyCode()==e.VK_ENTER)  
        {  
            int x=Integer.parseInt(data1.getText());
```

```

        int y=Integer.parseInt(data2.getText());
        String z=String.valueOf(x+y);
        hasil.setText(z);
    }
}
});

```

Yang diletakkan pada method AksiReaksi().Baris-baris program di atas berhubungan dengan event yang berhubungan dengan tombol-tombol keyboard. Dan sintaks :

```

if(e.getKeyCode()==e.VK_ENTER)
{
    .....
    .....
}

```

adalah sintaks yang memungkinkan suatu reaksi terjadi bila kita menekan tombol enter di keyboard.

Dengan cara yang sama, program aplikasi sederhana di atas dapat kita sempurnakan dengan memberikan event enter pada JTextField data1 dan JTextField data2 sehingga setelah data pertama dimasukkan dan kita tekan enter kursor akan pindah ke JTextField data2 siap untuk mengisi data kedua. Setelah data kedua diisi dan ditekan tombol enter maka fokus akan menuju ke tombol operasi, siap untuk ditekan. Untuk lebih jelasnya silahkan dicoba program berikut:

```

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

class AplikasiEvent1 extends JFrame
{
    JTextField data1=new JTextField(6);
    JTextField data2=new JTextField(6);
    JButton operasi=new JButton("+");
    JTextField hasil=new JTextField(6);
    JButton exit=new JButton("EXIT");

    AplikasiEvent1()
    {
        setTitle("Event Sederhana");
        setLocation(200,100);
        setSize(300,100);
    }
}

```

```

    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

void komponenVisual()
{
    getContentPane().setLayout(null);
    getContentPane().add(data1);
    data1.setBounds(10,10,50,20);
    getContentPane().add(data2);
    data2.setBounds(70,10,50,20);
    getContentPane().add(operasi);
    operasi.setBounds(130,10,50,20);
    getContentPane().add(hasil);
    hasil.setBounds(190,10,90,20);
    getContentPane().add(exit);
    exit.setBounds(190,35,90,20);
    setVisible(true);
}

void AksiReaksi()
{
    data1.addKeyListener(new KeyAdapter()
    {
        public void keyPressed(KeyEvent e)
        {
            if(e.getKeyCode()==e.VK_ENTER)
            {
                data2.requestFocus();
            }
        }
    });

    data2.addKeyListener(new KeyAdapter()
    {
        public void keyPressed(KeyEvent e)
        {
            if(e.getKeyCode()==e.VK_ENTER)
            {
                operasi.requestFocus();
            }
        }
    });

    operasi.addActionListener(new ActionListener()
    {

```

```

        public void actionPerformed(ActionEvent e)
        {
            int x=Integer.parseInt(data1.getText());
            int y=Integer.parseInt(data2.getText());
            String z=String.valueOf(x+y);
            hasil.setText(z);
        }
    });

    operasi.addKeyListener(new KeyAdapter()
    {
        public void keyPressed(KeyEvent e)
        {
            if(e.getKeyCode()==e.VK_ENTER)
            {
                int x=Integer.parseInt(data1.getText());
                int y=Integer.parseInt(data2.getText());
                String z=String.valueOf(x+y);
                hasil.setText(z);
                exit.requestFocus();
            }
        }
    });

    exit.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            System.exit(0);
        }
    });
}

public static void main(String args[])
{
    AplikasiEvent1 e1=new AplikasiEvent1();
    e1.komponenVisual();
    e1.AksiReaksi();
}
}

```

Bagian yang dicetak tebal adalah kode tambahan yang digunakan untuk memberikan event handler bila terjadi penekanan enter pada JTextField. Penjelasan sintaks-sintaks di atas adalah sebagai berikut:

```
data2.requestFocus()
```

adalah kode yang memungkinkan terjadinya perpindahan kursor dari JTextField pertama ke JTextField kedua.

### 3.3. Event Handling Yang Berkaitan Dengan Komponen Jradiobutton Dan Jtextarea

Sebelum masuk ke pemberian event handling pada JRadioButton kita berkenalan dulu dengan komponen visual JTextArea yang akan kita gunakan dalam program-program selanjutnya. JTextArea merupakan komponen yang mirip dengan JTextField tetapi dapat menampung lebih dari satu baris.

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

class AplikasiTextArea extends JFrame
{
    private JLabel label=new JLabel("Nama");
    private JTextArea area1=new JTextArea(3,20);

    AplikasiTextArea()
    {
        setTitle("Komponen JTextArea");
        setLocation(200,100);
        setSize(300,100);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

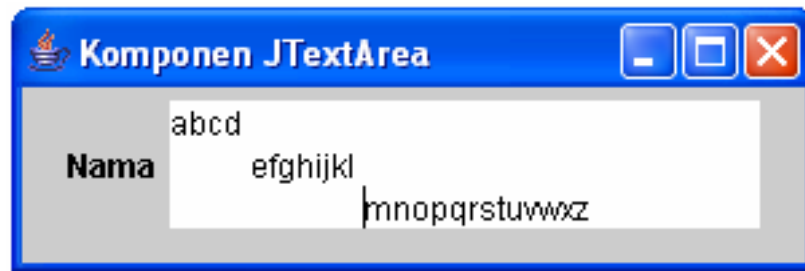
    void komponenVisual()
    {
        getContentPane().add(label);
        getContentPane().add(area1);
        getContentPane().setLayout(new FlowLayout());
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiTextArea ta=new AplikasiTextArea();
        ta.komponenVisual();
    }
}
```



```
}
}
```

Hasil dari program di atas adalah:



Gambar 3.3. Contoh hasil program even handling jradiobutton dan jtextarea

Penjelasan sintaks sebagai berikut :

```
private JTextArea area1=new JTextArea(3,20);
```

merupakan deklarasi dan pembentukan obyek JTextArea dimana angka 3 menunjukkan banyaknya baris dan 20 menunjukkan banyaknya kolom dari JTextArea yang ingin ditampilkan.

Selanjutnya mari kita perhatikan program berikut yang merupakan pemberian event handling untuk menangani pemilihan option menggunakan radio button dan hasilnya dicetak ke text area melalui klik mouse pada tombol cetak.

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

class AplikasiEvent5 extends JFrame
{
    JLabel lblnama=new JLabel("Nama");
    JTextField txnama=new JTextField(20);
    JLabel lblnim=new JLabel("NIM");
    JTextField txnim=new JTextField(7);
    JLabel lblkelamin=new JLabel("Jenis Kelamin");
    JRadioButton pria=new JRadioButton("Pria");
    JRadioButton wanita=new JRadioButton("Wanita");
    ButtonGroup kelompok=new ButtonGroup();
    JButton cetak=new JButton("Cetak");
    JTextArea hasil=new JTextArea();
```

```

AplikasiEvent5()
{
    setTitle("Event Sederhana");
    setLocation(300,100);
    setSize(300,320);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

void KomponenVisual()
{
    getContentPane().setLayout(null);
    getContentPane().add(lblnama);
    lblnama.setBounds(10,10,80,20);
    getContentPane().add(txnama);
    txnama.setBounds(105,10,175,20);
    getContentPane().add(lblnim);
    lblnim.setBounds(10,33,80,20);
    getContentPane().add(txnim);
    txnim.setBounds(105,33,70,20);
    getContentPane().add(lblkelamin);
    lblkelamin.setBounds(10,56,80,20);
    kelompok.add(pria);
    kelompok.add(wanita);
    getContentPane().add(pria);
    pria.setBounds(105,56,50,20);
    getContentPane().add(wanita);
    wanita.setBounds(160,56,70,20);
    getContentPane().add(cetak);
    cetak.setBounds(10,80,270,20);
    getContentPane().add(hasil);
    hasil.setBounds(10,105,270,150);
    setVisible(true);
}

void AksiReaksi()
{
    cetak.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            hasil.append(txnama.getText()+"\n");
            hasil.append(txnim.getText()+"\n");
            if(pria.isSelected()==true)

```

```

        {
            hasil.append(pria.getText()+"\n");
        }
        else
        {
            hasil.append(wanita.getText()+"\n");
        }
    }
});
}

public static void main(String args[])
{
    AplikasiEvent5 e5=new AplikasiEvent5();
    e5.KomponenVisual();
    e5.AksiReaksi();
}
}

```

Hasil program di atas adalah:



Gambar 3.4. contoh event click mouse

Pada program di atas, setelah Nama dan NIM diisi, pilihan pria atau wanita dipilih, serta tombol cetak ditekan, maka hasilnya akan ditampilkan di JTextArea. Yang perlu diperhatikan dalam baris kode di

atas adalah method yang berisi event handler untuk menangani kejadian ketika tombol cetak di klik.

```
void AksiReaksi()
{
    cetak.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            hasil.append(txnama.getText()+"\n");
            hasil.append(txnim.getText()+"\n");
            if(pria.isSelected()==true)
            {
                hasil.append(pria.getText()+"\n");
            }
            else
            {
                hasil.append(wanita.getText()+"\n");
            }
        }
    });
}
```

Penjelasan sintaks-sintaks di atas adalah sebagai berikut :

```
hasil.append(txnama.getText()+"\n");
hasil.append(txnim.getText()+"\n");
```

digunakan untuk mengambil text yang ada dalam txnama dan txnim, kemudian menuliskannya di JTextArea. Penggunaan hasil.append akan menyebabkan teks yang kita masukkan ke JTextArea akan terus bertambah tanpa penghapusan teks yang sudah ada. Bila kita menggunakan sintaks hasil.setText maka tulisan yang sebelumnya telah ada dalam JTextArea akan hilang digantikan oleh teks yang baru.

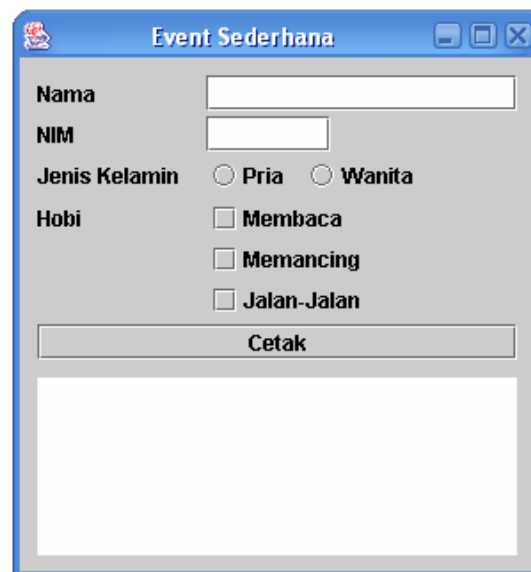
```
if(pria.isSelected()==true)
{
    hasil.append(pria.getText()+"\n");
}
else
{
    hasil.append(wanita.getText()+"\n");
}
```

```
}
```

merupakan baris-baris kode yang digunakan untuk menangkap pilihan dari pengguna terhadap RadioButton yang disediakan, dimana bila radiobutton pria yang dipilih maka teks pada JRadioButton pria akan tercetak pada JTextArea karena sintaks pria.getText() begitu juga sebaliknya..

### 3.4. Event Handling Yang Berkaitan Dengan Komponen Jcheckbox Dan JComboBox

Bila program di atas kita kembangkan untuk memberikan kesempatan kepada pengguna memasukkan informasi hobi yang diminati, maka dapat kita tambahkan obyek JCheckBox sehingga tampilan aplikasi akan tampak sebagai berikut:



Gambar 3.5. Contoh checkbox dan combobox

JCheckBox adalah komponen visual yang digunakan ketika pengguna memerlukan komponen untuk memilih satu atau banyak pilhan sekaligus. Berikut ini adalah kode program lengkap dari aplikasi di atas, dimana bagian yang dicetak tebal merupakan penambahan yang perlu dilakukan terhadap program sebelumnya.

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

```

import java.awt.event.*;

class AplikasiEvent5 extends JFrame
{
    JLabel lblnama=new JLabel("Nama");
    JTextField txnama=new JTextField(20);
    JLabel lblnim=new JLabel("NIM");
    JTextField txnim=new JTextField(7);
    JLabel lblkelamin=new JLabel("Jenis Kelamin");
    JRadioButton pria=new JRadioButton("Pria");
    JRadioButton wanita=new JRadioButton("Wanita");
    ButtonGroup kelompok=new ButtonGroup();
    JLabel lblhobi=new JLabel("Hobi");
    JCheckBox baca=new JCheckBox("Membaca");
    JCheckBox mancing=new JCheckBox("Memancing");
    JCheckBox jalan=new JCheckBox("Jalan-Jalan");
    JButton cetak=new JButton("Cetak");
    JTextArea hasil=new JTextArea();

    AplikasiEvent5()
    {
        setTitle("Event Sederhana");
        setLocation(300,100);
        setSize(300,320);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        getContentPane().setLayout(null);
        getContentPane().add(lblnama);
        lblnama.setBounds(10,10,80,20);
        getContentPane().add(txnama);
        txnama.setBounds(105,10,175,20);
        getContentPane().add(lblnim);
        lblnim.setBounds(10,33,80,20);
        getContentPane().add(txnim);
        txnim.setBounds(105,33,70,20);
        getContentPane().add(lblkelamin);
        lblkelamin.setBounds(10,56,80,20);
        kelompok.add(pria);
        kelompok.add(wanita);
        getContentPane().add(pria);
        pria.setBounds(105,56,50,20);
        getContentPane().add(wanita);
    }
}

```

```

wanita.setBounds(160,56,70,20);
getContentPane().add(lblhobi);
lblhobi.setBounds(10,80,70,20);
getContentPane().add(baca);
baca.setBounds(105,80,100,20);
getContentPane().add(mancing);
mancing.setBounds(105,103,100,20);
getContentPane().add(jalan);
jalan.setBounds(105,126,100,20);
getContentPane().add(cetak);
cetak.setBounds(10,150,270,20);
getContentPane().add(hasil);
hasil.setBounds(10,180,270,100);
setVisible(true);
}

void AksiReaksi()
{
    cetak.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            hasil.append(txnama.getText()+"\n");
            hasil.append(txnim.getText()+"\n");
            if(pria.isSelected()==true)
            {
                hasil.append(pria.getText()+"\n");
            }
            else
            {
                hasil.append(wanita.getText()+"\n");
            }

            if(baca.isSelected()==true)
            {
                hasil.append(baca.getText()+"\n");
            }
            if(mancing.isSelected()==true)
            {
                hasil.append(mancing.getText()+"\n");
            }
            if(jalan.isSelected()==true)
            {
                hasil.append(jalan.getText()+"\n");
            }
        }
    });
}

```

```

    }
    });
}

public static void main(String args[])
{
    AplikasiEvent5 e5=new AplikasiEvent5();
    e5.KomponenVisual();
    e5.AksiReaksi();
}
}

```

Yang perlu diperhatikan dalam program di atas adalah penggunaan sintak-sintaks :

```

if(baca.isSelected()==true)
{
    hasil.append(baca.getText()+"\n");
}

```

Yang digunakan untuk memberikan reaksi ketika komponen CheckBox dipilih. Pada persoalan di atas, ketika CheckBox baca dipilih maka teks dari CheckBox tersebut akan dituliskan ke TextArea.

Bila kita jalankan program diatas maka tampilannya adalah sebagai berikut:

Gambar 3.6. Contoh hasil JcheckBox

### 3.5. JMenu



Menu digunakan untuk mengatur berbagai program yang telah kita buat agar menjadi lebih sederhana dan mudah digunakan dengan memberikan pilihan-pilihan yang disusun secara rapi dan terstruktur. Berikut ini adalah contoh program bagaimana caranya membuat komponen menu di Java menggunakan JMenu.

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

class AplikasiMenu extends JFrame
{
    JMenuBar mb=new JMenuBar();
    JMenu file=new JMenu("File");
    JMenu help=new JMenu("Help");
    JMenuItem open=new JMenuItem("Open");
    JMenuItem close=new JMenuItem("Close");
    JMenuItem quit=new JMenuItem("Quit");
    JMenuItem about=new JMenuItem("About");

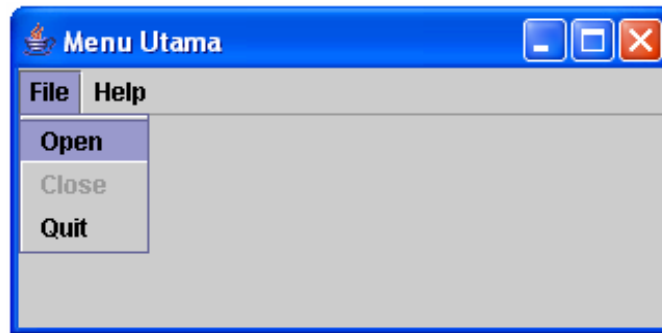
    AplikasiMenu()
    {
        setTitle("Menu Utama");
        setSize(320,160);
        setLocation(300,200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        setJMenuBar(mb);
        mb.add(file);
        mb.add(help);
        file.add(open);
        file.add(close);
        close.setEnabled(false);
        file.add(quit);
        help.add(about);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiMenu m1=new AplikasiMenu();
        m1.KomponenVisual();
    }
}
```

```
}  
}
```

Hasil dari program di atas adalah:



Gambar 3.7. Contoh hasil JMenu

Penjelasan dari sintak-sintak di atas adalah sebagai berikut:

```
JMenuBar mb=new JMenuBar();
```

Digunakan untuk membuat tempat bagi penempatan menu-menu yang akan dibuat.

```
JMenu file=new JMenu("File");  
JMenu help=new JMenu("Help");
```

Merupakan cara yang digunakan untuk membuat menu. Dalam kode di atas menu yang akan dibuat adalah menu File dan Help.

```
JMenuItem open=new JMenuItem("Open");  
JMenuItem close=new JMenuItem("Close");  
JMenuItem quit=new JMenuItem("Quit");  
JMenuItem about=new JMenuItem("About");
```

Adalah sintaks untuk membuat submenu dari masing-masing menu yang telah dibuat. Pada bagian ini belum ada informasi submenu mana yang menjadi bagian dari menu tertentu, baris kode untuk itu dapat kita letakkan pada method `komponenVisual()`.

```
void KomponenVisual()  
{
```

```

    setJMenuBar(mb);
    mb.add(file);
    mb.add(help);
    file.add(open);
    file.add(close);
    close.setEnabled(false);
    file.add(quit);
    help.add(about);
    setVisible(true);
}

```

Pada method inilah kita mengatur letak menu, dan submenu mana yang merupakan bagian dari menu File dan submenu mana yang menjadi menu Help.

Sekarang mari kita perhatikan contoh lain sebagai ilustrasi bagaimana memberikan event handling pada menu dan submenu untuk melakukan suatu reaksi.

```

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

class AplikasiEvent8 extends JFrame
{
    JMenuBar mb=new JMenuBar();
    JMenu data=new JMenu("Data");
    JMenu help=new JMenu("Help");
    JMenuItem input=new JMenuItem("Input Data");
    JMenuItem edit=new JMenuItem("Edit Data");
    JMenuItem quit=new JMenuItem("Keluar");
    JMenuItem about=new JMenuItem("About");
    AplikasiEvent5 isidata=new AplikasiEvent5();

    AplikasiEvent8()
    {
        setTitle("Menu Utama");
        setSize(320,160);
        setLocation(300,200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()

```

```

{
    setJMenuBar(mb);
    mb.add(data);
    mb.add(help);

    data.add(input);
    data.add(edit);
    data.addSeparator();
    data.add(quit);
    help.add(about);

    setVisible(true);
}

void AksiReaksi()
{
    input.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            JOptionPane.showInputDialog(null,"Masukkan nama anda","Data
            Personalia",JOptionPane.INFORMATION_MESSAGE);
        }
    });

    edit.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            isidata.KomponenVisual();
            isidata.AksiReaksi();
        }
    });

    quit.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            System.exit(0);
        }
    });

    about.addActionListener(new ActionListener()
    {

```

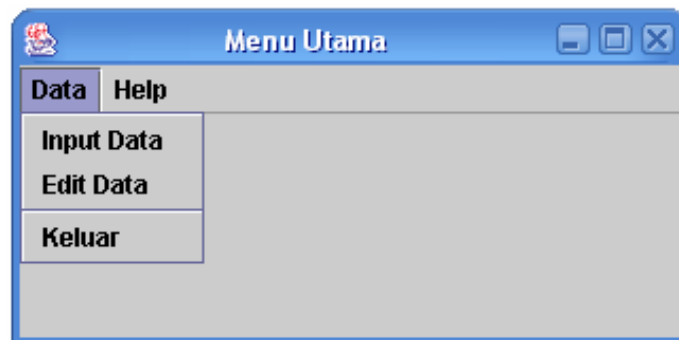
```

    public void actionPerformed(ActionEvent e)
    {
        new FrameAbout();
    }
});
}

public static void main(String args[])
{
    AplikasiEvent8 e8=new AplikasiEvent8();
    e8.KomponenVisual();
    e8.AksiReaksi();
}
}

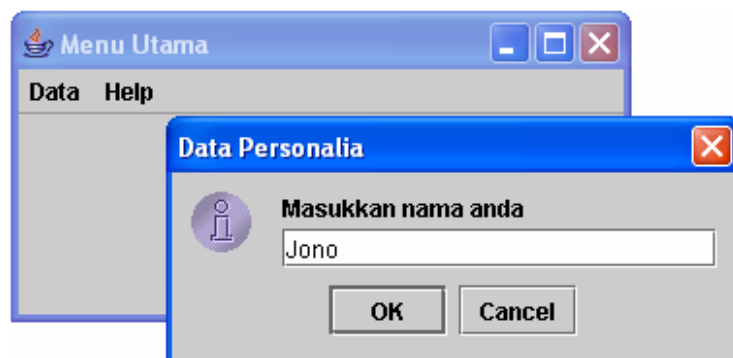
```

Hasil dari program di atas adalah seperti dibawah ini



Gambar 3.8. Even handling pada menu

Ketika menu Data dipilih maka akan tampil submenu yang ada dalam menu tersebut. Tampilan ketika Menu Input Data di Klik adalah sebagai berikut:



Gambar 3.9. Event click pada menu

Ketika Menu Edit Data di Klik maka akan tampil form dari aplikasi yang telah kita buat sebelumnya karena kita telah membentuk sebuah obyek dari class AplikasiEvent5.

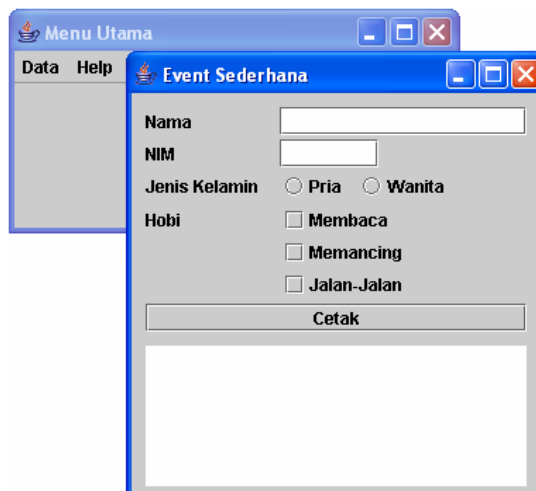
```
AplikasiEvent5 isidata=new AplikasiEvent5();
```

Dan dengan menggunakan sintak:

```
edit.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        isidata.KomponenVisual();
        isidata.AksiReaksi();
    }
});
```

Maka kita memanggil method KomponenVisual() dan method AksiReaksi() dari class AplikasiEvent5. Tampilannya adalah seperti terlihat pada gambar 3.10.

Dalam program ini, tampilan ketika Menu Help About di Klik akan muncul sebuah form yang hanya berisi sebuah gambar. Tentu saja form ini dapat kita isi dengan informasi lain misalnya versi dari program kita, tahun pembuatan, tim pembuat atau informasi lain. Class yang digunakan untuk membuat form about adalah sebagai berikut :



Gambar 3.11. Penggambungan form

Berikutnya membuat aplikasi menu dengan hasil berupa view gambar, bila pilihan [Data] pada menu dipilih maka akan ditampilkan gambar, seperti pada listing program di bawah ini.

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

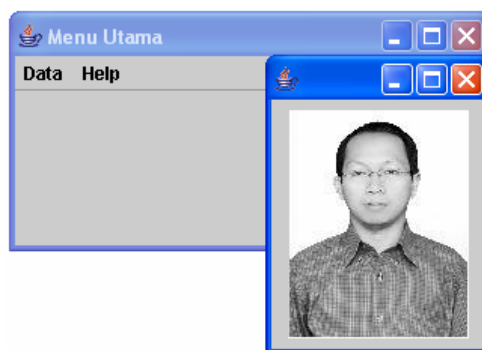
class FrameAbout extends JFrame
{
    JLabel picture=new JLabel(new ImageIcon("foto.jpg"));

    FrameAbout()
    {
        picture.setPreferredSize(new Dimension(200,200));
        setLocation(500,300);
        setSize(150,200);
        getContentPane().add(picture);
        setDefaultCloseOperation(JFrame.HIDE_ON_CLOSE);
        setVisible(true);
    }
}

class tersebut kita tempatkan pada method aksiReaksi() pada class
AplikasiEvent8 dengan cara :
```

```
        about.addActionListener(new ActionListener()
        {
            public void actionPerformed(ActionEvent e)
            {
                new FrameAbout();
            }
        });
```

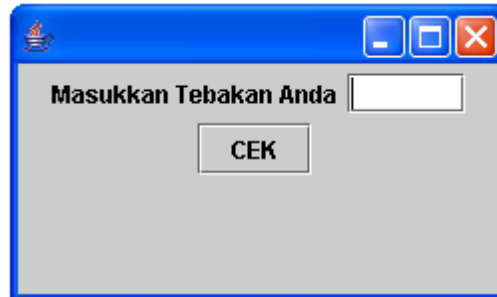
Hasilnya outputnya adalah sebagai berikut:



Gambar 3.12. Contoh hasil form about

### 3.6. LATIHAN

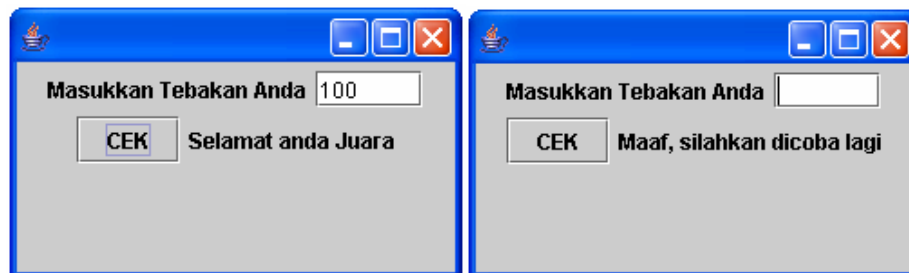
- (1) Bangunlah sebuah program tebak angka, dengan tampilan sebagai berikut:



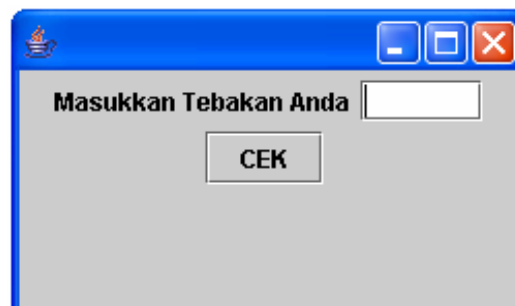
Gambar 3.13. Program tebak angka

Bila kita masukkan sebuah bilangan ke dalam textfield dan bilangan tersebut sesuai dengan bilangan yang kita tentukan, maka akan muncul tampilan gambar 3.14a

Bila angka yang kita masukkan tidak sesuai, maka akan muncul tampilan gambar 3.14b



Gambar 3.14. (a) Tebak angka benar (b)Tebak angka salah  
Dan ketika kita memasukkan sebuah bilangan baru ke dalam textfield maka tampilan akan kembali ke kondisi awal.



Gambar 3.15. Refresh



(2) Buatlah contoh-contoh program sederhana terhadap Listener-listener berikut:

FocusListener	FocusAdapter	focusGained(FocusEvent) focusLost(Focus Event)
ItemListener	none	itemStateChanged(ItemEvent)
KeyListener	KeyAdapter	keyPressed(KeyEvent) keyReleased(KeyEvent) keyTyped(KeyEvent)
MouseListener	MouseAdapter	mouseClicked(MouseEvent) mouseEntered(MouseEvent) mouseExited(MouseEvent) mousePressed(MouseEvent) mouseReleased(MouseEvent)
WindowListener	WindowAdapter	windowActivated(WindowEvent) windowClosed (WindowEvent) windowClosing (WindowEvent) windowDeactivated(WindowEvent) windowOpened(WindowEvent)

## **BAB 4**

### **PENGATURAN LAYOUT**

---

**Tujuan:**

1. Mahasiswa dapat mengatur layout dari form yang dibuat menggunakan pemrograman bahasa Java.
  2. Mahasiswa dapat menjelaskan perbedaan *FormLayout*, *BorderLayout*, *GridLayout* dan *NoneLayout*
  3. Mahasiswa dapat menerapkan *JPanel*.
- 

Pengaturan layout digunakan untuk mengatur posisi dari komponen visual penyusun program sesuai dengan desain *user interface*. Beberapa pilihan layout telah disediakan java, dimana keputusan untuk menggunakan jenis layout tertentu bergantung pada jenis aplikasi yang ingin dibuat serta tingkat kerapian yang diinginkan. Pada bab-bab terdahulu, manajemen layout yang digunakan adalah none layout, dimana pengaturan posisi komponen dalam frame dilakukan sendiri oleh programmer. Beberapa jenis layout yang lain dapat digunakan untuk berbagai keperluan. Berikut ini adalah contoh program yang terdiri dari beberapa komponen dan diatur dengan menggunakan beberapa jenis manajemen layout agar dapat dilihat karakteristik masing-masing.

#### **4.1. FlowLayout**

*FlowLayout* adalah jenis pengaturan layout yang paling sederhana, dimana semua komponen akan tersusun dari kiri ke kanan sepanjang frame, dan akan pindah ke bawah bila telah sampai batas kanan frame. Kode program yang digunakan untuk melakukan pengaturan *flowlayout* adalah:

```
getContentPane().setLayout(new FlowLayout())
```

yang dapat kita letakkan pada method `komponenVisual()` bersama dengan komponen-komponen visual. Berikut ini adalah contoh program sederhana dimana komponen visual yang digunakan diatur menggunakan `FlowLayout`.

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

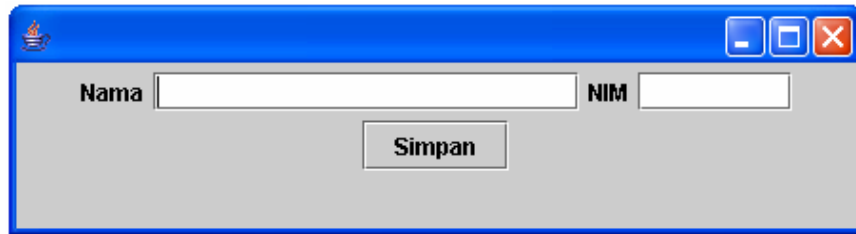
class AplikasiFlowLayout extends JFrame
{
    JLabel nama=new JLabel("Nama");
    JLabel nim=new JLabel("NIM");
    JTextField txnama=new JTextField(20);
    JTextField txnim=new JTextField(7);
    JButton tombolSimpan=new JButton("Simpan");

    AplikasiFlowLayout()
    {
        setLocation(200,100);
        setSize(450,120);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        getContentPane().setLayout(new FlowLayout());
        getContentPane().add(nama);
        getContentPane().add(txnama);
        getContentPane().add(nim);
        getContentPane().add(txnim);
        getContentPane().add(tombolSimpan);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiFlowLayout fl=new AplikasiFlowLayout();
        fl.KomponenVisual();
    }
}
```

Hasil program di atas adalah :



Gambar 4.1. Contoh FlowLayout

Dalam program di atas komponen visual yang digunakan adalah JLabel, JTextField dan JButton. Terlihat bahwa komponen visual secara berurutan menempati posisi dari kiri ke kanan dan selalu berada di tengah frame.

## 4.2. BorderLayout

BorderLayout merupakan jenis layout yang bekerja dengan membagi frame menjadi lima bagian yaitu NORTH, EAST, SOUTH, WEST dan CENTER. Komponen visual dapat diletakkan pada bagian-bagian tersebut. Bila program sebelumnya diganti layout-nya menggunakan BorderLayout maka programnya adalah sebagai berikut:

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

class AplikasiBorderLayout extends JFrame
{
    JLabel nama=new JLabel("Nama");
    JLabel nim=new JLabel("NIM");
    JTextField txnama=new JTextField(20);
    JTextField txnim=new JTextField(7);
    JButton tombolSimpan=new JButton("Simpan");

    AplikasiBorderLayout()
    {
        setLocation(200,100);
        setSize(450,120);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
```

```

        getContentPane().setLayout(new BorderLayout());
        getContentPane().add(nama, "North");
        getContentPane().add(txnama, "East");
        getContentPane().add(nim, "Center");
        getContentPane().add(txnim, "West");
        getContentPane().add(tombolSimpan, "South");
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiBorderLayout fl=new AplikasiBorderLayout();
        fl.KomponenVisual();
    }
}

```

Hasil program di atas adalah :



Gambar 4.2. Contoh BorderLayout

Kode program yang digunakan untuk pengaturan BorderLayout adalah :

```

        getContentPane().setLayout(new BorderLayout());
        getContentPane().add(nama, "North");

```

method add membutuhkan argumen nama obyek dan salah satu dari posisi layout yang telah disediakan dalam BorderLayout().

### 4.3. GRIDLAYOUT

GridLayout adalah jenis layout yang bekerja berdasar baris dan kolom. Dengan layout ini kita dapat memberikan argumen banyaknya baris dan kolom sesuai dengan kebutuhan. Berikut ini adalah program sebelumnya menggunakan pengaturan GridLayout.

```

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

class AplikasiGridLayout extends JFrame
{
    JLabel nama=new JLabel("Nama");
    JLabel nim=new JLabel("NIM");
    JTextField txnama=new JTextField(20);
    JTextField txnim=new JTextField(7);
    JButton tombolSimpan=new JButton("Simpan");

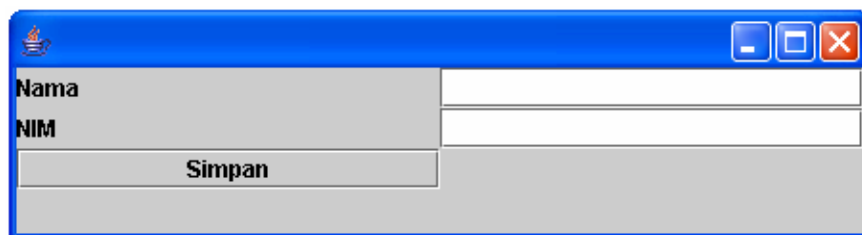
    AplikasiGridLayout()
    {
        setLocation(200,100);
        setSize(450,120);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        getContentPane().setLayout(new GridLayout(4,2));
        getContentPane().add(nama);
        getContentPane().add(txnama);
        getContentPane().add(nim);
        getContentPane().add(txnim);
        getContentPane().add(tombolSimpan);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiGridLayout fl=new AplikasiGridLayout();
        fl.KomponenVisual();
    }
}

```

Hasil dari program di atas:



Gambar 4.3. Contoh GridLayout

Kode program yang digunakan untuk pengaturan GridLayout adalah :

```
getContentPane().setLayout(new GridLayout(4,2));
```

dimana angka 4 pada argumen GridLayout menunjukkan banyaknya baris dan 2 menunjukkan banyaknya kolom.

#### 4.4. NoneLayout

NoneLayout merupakan jenis layout yang dapat menghasilkan tampilan yang rapi karena kita dapat mengatur posisi komponen secara detil berdasar koordinatnya. Konsekuensinya dengan layout ini waktu yang diperlukan relatif lebih banyak dibanding layout yang lain karena kita perlu menentukan posisi koordinat tiap komponen.

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

class AplikasiNoneLayout extends JFrame
{
    JLabel nama=new JLabel("Nama");
    JLabel nim=new JLabel("NIM");
    JTextField txnama=new JTextField(20);
    JTextField txnim=new JTextField(7);
    JButton tombolSimpan=new JButton("Simpan");

    AplikasiNoneLayout()
    {
        setLocation(200,100);
        setSize(450,120);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        getContentPane().setLayout(null);
        getContentPane().add(nama);
        nama.setBounds(10,10,100,25);
        getContentPane().add(txnama);
        txnama.setBounds(150,10,200,20);
        getContentPane().add(nim);
        nim.setBounds(10,30,100,25);
        getContentPane().add(txnim);
        txnim.setBounds(150,30,100,20);
    }
}
```

```

        getContentPane().add(tombolSimpan);
        tombolSimpan.setBounds(150,50,100,25);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiNoneLayout fl=new AplikasiNoneLayout();
        fl.KomponenVisual();
    }
}

```

Hasil dari program di atas adalah:



Gambar 4.4. Contoh NoneLayout

Kode program yang digunakan untuk pengaturan NoneLayout adalah :

```

        getContentPane().setLayout(null);
        getContentPane().add(nama);
        nama.setBounds(10,10,100,25);

```

Pengaturan ini membutuhkan method `setBounds` yang digunakan untuk meletakkan komponen visual pada posisi tertentu. Method ini membutuhkan argumen koordinat x, koordinat y, lebar obyek dan tinggi obyek. Pada contoh di atas, komponen label yang kita letakkan pada posisi `x=10, y=10` lebar 100 dan tinggi 25.

## 4.5. JPANEL

JPanel adalah komponen visual yang digunakan untuk membantu mengatur letak komponen lain agar terlihat lebih tertata rapi dan nyaman. Berikut ini adalah contoh penggunaan JPanel dalam sebuah program.



```

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

class AplikasiPanel1 extends JFrame
{
    JPanel panel1=new JPanel();
    JLabel label1=new JLabel("Nama");
    JTextField text1=new JTextField(20);

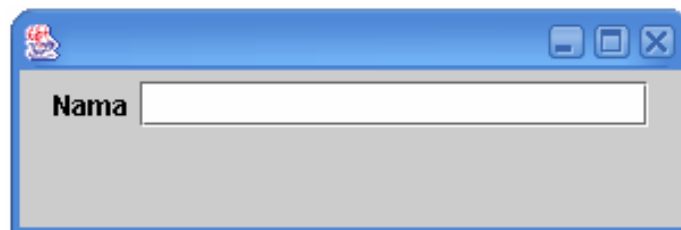
    AplikasiPanel1()
    {
        setLocation(300,100);
        setSize(300,150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        panel1.add(label1);
        panel1.add(text1);
        getContentPane().add(panel1);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiPanel1 p1=new AplikasiPanel1();
        p1.KomponenVisual();
    }
}

```

Hasil program di atas adalah :



Gambar 4.5. Contoh JPanel

Seperti biasa, sebelum digunakan komponen JPanel perlu dideklarasikan menggunakan sintaks :

```
JPanel panel1=new JPanel();
```

Selanjutnya dalam komponen JPanel ini kita letakkan komponen visual lain sesuai keperluan, kemudian semua komponen tersebut kita letakkan ke dalam method komponenVisual() sebagai berikut.

```
void KomponenVisual()
{
    panel1.add(label1);
    panel1.add(text1);
    getContentPane().add(panel1);
    setVisible(true);
}
```

Bila contoh program di atas belum terlalu terlihat peran dari JPanel, program berikut diharapkan bisa memberikan gambaran yang lebih baik bagaimana JPanel digunakan.

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

class AplikasiPanel2 extends JFrame
{
    JPanel panel1=new JPanel();
    JLabel label1=new JLabel("Nama");
    JTextField text1=new JTextField(20);
    JPanel panel2=new JPanel();
    JButton tombol1=new JButton("Simpan");
    JButton exit=new JButton("Exit");

    AplikasiPanel2()
    {
        setLocation(300,100);
        setSize(300,150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        panel1.add(label1);
        panel1.add(text1);
        getContentPane().add(panel1, BorderLayout.NORTH);
        panel2.add(tombol1);
        panel2.add(exit);
        getContentPane().add(panel2, BorderLayout.SOUTH);
    }
}
```

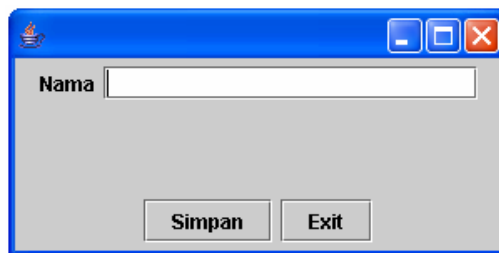
```

    setVisible(true);
}

public static void main(String args[])
{
    AplikasiPanel2 p2=new AplikasiPanel2();
    p2.KomponenVisual();
}
}

```

Hasil program di atas adalah :



Gambar 4.6. Contoh Jpanel 2

Pada contoh di atas terdapat dua JPanel. Layout yang digunakan adalah BorderLayout sehingga komponen visual dapat diletakkan dalam lima posisi. JPanel pertama diletakkan pada posisi North dan JPanel kedua diletakkan pada posisi South.

```

void KomponenVisual()
{
    panel1.add(label1);
    panel1.add(text1);
    getContentPane().add(panel1, BorderLayout.NORTH);
    panel2.add(tombol1);
    panel2.add(exit);
    getContentPane().add(panel2, BorderLayout.SOUTH);
    setVisible(true);
}

```

Panel pertama diisi dengan JLabel dan JTextField, sementara panel kedua berisi JButton. Kedua Panel tersebut kemudian diletakkan dalam Frame menggunakan sintaks :

```

getContentPane().add(panel1, BorderLayout.NORTH);

```

Bila diperlukan dapat juga dilakukan pengaturan dengan menggunakan salah satu diantara beberapa layout terhadap komponen visual dalam sebuah JPanel. Berikut ini adalah contohnya.

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

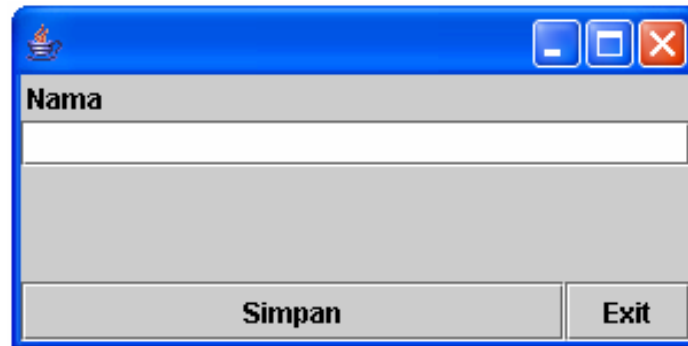
class AplikasiPanel3 extends JFrame
{
    JPanel panel1=new JPanel();
    JLabel label1=new JLabel(" Nama");
    JTextField text1=new JTextField(20);
    JPanel panel2=new JPanel();
    JButton tombol1=new JButton("Simpan");
    JButton exit=new JButton("Exit");

    AplikasiPanel3()
    {
        setLocation(300,100);
        setSize(300,150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        panel1.add(label1);
        panel1.add(text1);
        panel1.setLayout(new GridLayout(2,1));
        getContentPane().add(panel1, BorderLayout.NORTH);
        panel2.setLayout(new BorderLayout());
        panel2.add(tombol1,"Center");
        panel2.add(exit,"East");
        getContentPane().add(panel2, BorderLayout.SOUTH);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AplikasiPanel3 p3=new AplikasiPanel3();
        p3.KomponenVisual();
    }
}
```

Hasil program di atas adalah:



Gambar 4.7. Contoh pengaturan dalam Jpanel

Pada program di atas, panel pertama diatur menggunakan GridLayout 2 baris 1 kolom.

```
panel1.add(label1);  
panel1.add(text1);  
panel1.setLayout(new GridLayout(2,1));  
getContentPane().add(panel1, BorderLayout.NORTH);
```

Sementara itu panel kedua menggunakan BorderLayout yang diisi dengan Tombol simpan pada posisi Center dan tombol exit pada posisi East.

## 4.6. LATIHAN

- (1) Buatlah sebuah form pemesanan tiket kereta api dengan layout seperti terlihat pada gambar 4.8 berikut:

A screenshot of a Java Swing window titled 'Pemesanan Tiket Kereta Api'. The window has a blue title bar with standard Windows controls. The main area is a light gray panel. It contains several input fields and buttons. At the top, there is a label 'Nama Customer' followed by a text input field containing 'Abdullah'. Below this, there are three labels: 'Nama Kereta', 'Dari', and 'Menuju'. Each label is followed by a dropdown menu. The 'Nama Kereta' dropdown shows 'Mutiaras Selatan'. The 'Dari' dropdown shows 'Surabaya'. The 'Menuju' dropdown shows 'Bandung'. Below these, there is a label 'Kuantitas Pesanan' followed by a text input field containing '3'. Below that, there is a label 'Tanggal Berlaku' followed by a text input field containing '21/09/2006'. Below that, there is a label 'Petugas' followed by a dropdown menu showing 'Yuni'. At the bottom of the panel, there are three buttons: 'Simpan', 'Edit', and 'Exit'.

Gambar 4.8. Form pemesanan tiket KA

(2) Bangunlah Form untuk simulasi aplikasi Transaksi Mesin ATM dengan layout seperti terlihat pada gambar 4.9. berikut:



**Transaksi Bank**

Masukkan No PIN Anda

100.000 ▼

**Jenis Transaksi**

☐ Penarikan Tunai

☐ Transfer Uang

☐ Rekening Koran

☐ Pembayaran Tunai

**Next** **Finish**

Gambar 4.9. Form simulasi aplikasi ATM

## BAB 5

### PEMROGRAMAN GRAFIS

---

**Tujuan:**

1. *Mahasiswa dapat menjelaskan konsep dasar pemrograman grafis pada bahasa Java.*
  2. *Mahasiswa dapat membuat program untuk menggambar bentuk-bentuk grafik primitive seperti garis, kotak, segitiga dan lain-lain.*
  3. *Mahasiswa dapat membuat program untuk menampilkan dan mengolah data gambar (image).*
- 

Pemrograman grafis digunakan dalam banyak hal mulai dari upaya untuk memperoleh tampilan yang indah, aplikasi animasi, aplikasi simulasi, aplikasi pengolahan citra sampai visi komputer. Berikut ini adalah beberapa pemrograman dasar di bidang grafis dalam java yang diharapkan dapat menjadi bekal awal untuk memahami berbagai topik seputar grafis yang begitu luas. Pembahasan pemrograman grafis akan dimulai dari menggambar obyek-obyek sederhana menggunakan garis, menggambar fungsi dan bagaimana dapat menampilkan dan mengolah data gambar (image processing) dengan pembahasan yang sangat sederhana.

Bahasa Java merupakan salah satu bahasa dengan fasilitas berupa komponen-komponen grafis yang lengkap. Karena itulah banyak program grafik, dan permainan yang berbasis grafik dibuat menggunakan bahasa Java, khususnya untuk permainan di perangkat-perangkat mobile. Hal ini akan dibahas dalam pembahasan J2ME yang merupakan materi lanjut dari pemrograman Java dan tidak dibahas dalam buku ini

## 5.1. DASAR PEMROGRAMAN GRAFIS

Salah satu cara untuk membuat aplikasi grafis dengan java adalah dengan membuat class yang merupakan turunan dari class canvas. Dalam class inilah kita meletakkan gambar atau animasi, dan membentuk sebuah obyek untuk mengakses gambar kita tersebut. Berikut ini adalah contoh program dan penjelasan langkah demi langkah pembuatan program yang berhubungan dengan grafis dalam Java.

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

class Kanvas1 extends Canvas
{
    public void paint(Graphics g)
    {
    }
}

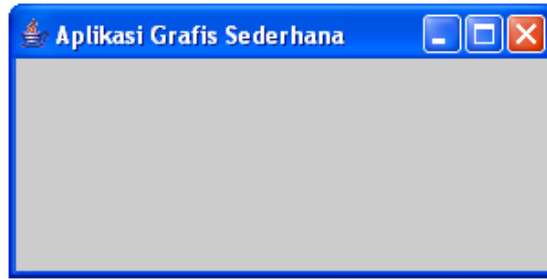
class Menggambar extends JFrame
{
    Kanvas1 gambar=new Kanvas1();
    Menggambar()
    {
        super("Aplikasi Grafis ");
        setLocation(100,100);
        setSize(250,150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        getContentPane().setLayout(new BorderLayout());
        getContentPane().add(gambar,BorderLayout.CENTER);
        setVisible(true);
    }

    public static void main (String[] args)
    {
        Menggambar g1=new Menggambar();
        g1.KomponenVisual();
    }
}
```

Hasil program di atas adalah sebagai berikut:





Gambar 5.1. Contoh aplikasi grafis sederhana

Penjelasan dari sintak-sintak program di atas adalah sebagai berikut:

```
class Kanvas1 extends Canvas
{
    public void paint(Graphics g)
    {
    }
}
```

Sintaks ini merupakan class tempat menggambar berbagai obyek gambar. Class inilah yang digunakan untuk meletakkan gambar yang kita bangun menggunakan kode-kode tertentu. Kode-kode tersebut diletakkan dalam method paint. Pada method paint() di atas tidak terdapat baris kode apapun sehingga output yang muncul adalah frame kosong tanpa gambar apapun.

```
class Menggambar extends JFrame
{
    Kanvas1 gambar=new Kanvas1();
    .....
    .....
}
```

Class Menggambar adalah program aplikasi grafis kita. Pada class ini kita membentuk obyek menggunakan class Kanvas1 dengan nama gambar.

```
Kanvas1 gambar=new Kanvas1();
```

Obyek gambar ini kemudian dimasukkan dalam method komponenVisual() untuk menampilkan di frame dengan cara:

```
getContentPane().setLayout(new BorderLayout());
```

```
getContentPane().add(gambar,BorderLayout.CENTER);
```

Layout diatur menggunakan BorderLayout dan obyek gambar diletakkan dalam salah satu diantara lima posisi yang ada.

Output dari program di atas berupa frame kosong tanpa gambar apapun. Method paint() pada Class Kanvas1 dapat diisi dengan berbagai sintaks untuk membuat aplikasi grafis, misalnya kita isi dengan sintaks berikut :

```
class Kanvas1 extends Canvas
{
    public void paint(Graphics g)
    {
        g.setColor(Color.red);
        g.drawLine(10,10,250,10);
        g.setColor(Color.blue);
        g.drawRect(10,20,100,50);
        g.setColor(Color.red);
        g.drawOval(120,20,50,50);
        g.drawString("Selamat Belajar", 100,100);
        g.setColor(Color.black);
    }
}
```

Penjelasan sintaks-sintaks di atas adalah sebagai berikut :

```
g.setColor(Color.red);
```

Digunakan untuk mengatur warna dari obyek gambar setelah baris tersebut dituliskan..

```
g.drawLine(10,10,250,10);
```

Digunakan untuk menggambar bentuk garis dimana argumen pertama dan kedua menunjukkan koordinat titik awal garis, argumen ketiga dan keempat menunjukkan koordinat titik akhir garis.

```
g.drawRect(10,20,100,50);
```

Digunakan untuk menggambar bentuk segi empat dimana argumen pertama dan kedua merupakan ujung kiri atas, argumen ketiga dan keempat menunjukkan ujung kanan bawah dari segi empat.

```
g.drawOval(120,20,50,50);
```

Digunakan untuk menggambar bentuk lingkaran dimana argumen pertama dan kedua merupakan koordinat pusat, argumen ketiga jarak horizontal dan argumen keempat adalah jarak vertikal.

```
g.drawString("Selamat Belajar", 100,100);
```

Digunakan untuk menggambar bentuk string. Argumen pertama merupakan string yang akan ditampilkan, argumen kedua dan ketiga menunjukkan posisi dari tulisan.

Program lengkap setelah penambahan baris-baris kode di atas adalah sebagai berikut:

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

class Kanvas1 extends Canvas
{
    public void paint(Graphics g)
    {
        g.setColor(Color.red);
        g.drawLine(10,10,250,10);
        g.setColor(Color.blue);
        g.drawRect(10,20,100,50);
        g.setColor(Color.red);
        g.drawOval(120,20,50,50);
        g.drawString("Selamat Belajar", 100,100);
        g.setColor(Color.black);
    }
}

class Menggambar extends JFrame
{
    Kanvas1 gambar=new Kanvas1();
}
```

```

Menggambar()
{
    super("Aplikasi Grafis Sederhana");
    setLocation(100,100);
    setSize(300,150);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

void KomponenVisual()
{
    getContentPane().setLayout(new BorderLayout());
    getContentPane().add(gambar,BorderLayout.CENTER);
    setVisible(true);
}

public static void main (String[] args)
{
    Menggambar g1=new Menggambar();
    g1.KomponenVisual();
}
}

```

Dengan isi tersebut maka ketika program menggambar.java dijalankan Hasil dari program di atas akan berubah menjadi:



Gambar 5.2. Contoh menggambar obyek dasar

Masih banyak method lain yang berhubungan dengan bentuk-bentuk grafis ini, silahkan ber-eksplorasi untuk menambah wawasan anda. Salah satu kegunaan dari pembentukan gambar-gambar di atas adalah untuk keperluan pembuatan grafik dan animasi. Berikut ini adalah program visualisasi grafik sinus dan cosinus sederhana yang didasarkan pada pengetahuan di atas.

```

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

class KanvasAnimasi extends Canvas
{
    public void paint(Graphics g)
    {
        g.setColor(Color.black);
        g.drawLine(10,10,10,210);
        g.drawLine(5,110,380,110);
        g.drawString("1",2,14);
        g.drawString("-1",14,210);
        g.drawString("0",13,123);
        g.drawString("90",100,123);
        g.drawString("180",190,123);
        g.drawString("270",280,123);
        g.drawString("360",370,123);

        g.setColor(Color.red);
        for(int i=0; i<360; i+=1)
        {
            int y=(int) (Math.sin(i*Math.PI/180)*100)*-1;
            g.drawOval(i+10,y+110,1,1);
        }

        g.setColor(Color.blue);
        for(int i=0; i<360; i+=1)
        {
            int y=(int) (Math.cos(i*Math.PI/180)*100)*-1;
            g.drawOval(i+10,y+110,1,1);
        }
    }
}

class Animasi extends JFrame
{
    KanvasAnimasi gambar=new KanvasAnimasi();

    Animasi()
    {
        setTitle("Plotting Fungsi Sinus dan Cosinus ");
        setLocation(200,100);
        setSize(400,260);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}

```

```

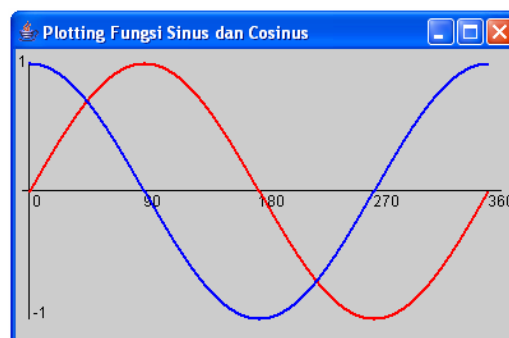
    }

    void KomponenVisual()
    {
        getContentPane().setLayout(new BorderLayout());
        getContentPane().add(gambar, BorderLayout.CENTER);
        setVisible(true);
    }

    public static void main (String[] args)
    {
        Animasi anime=new Animasi();
        anime.KomponenVisual();
    }
}

```

Hasil dari program di atas adalah sebagai berikut:



Gambar 5.3. Contoh gambar gelombang yang bergerak

Seperti pada program sebelumnya, program ini juga terdiri dari dua class yaitu class `KanvasAnimasi` yang berisi gambar atau bentuk-bentuk grafis dan class `Animasi` yang digunakan untuk menampilkan gambar tersebut. Perubahan terjadi pada class `KanvasAnimasi` sementara class `Animasi` tidak mengalami perubahan.

Class `KanvasAnimasi` terdiri dari dua bagian yaitu bagian yang digunakan untuk menggambar sumbu x dan y, serta bagian yang digunakan untuk menampilkan grafik sinus dan cosinus. Bagian untuk menggambar sumbu x dan y adalah :

```

g.setColor(Color.black);
g.drawLine(10,10,10,210);

```

```
g.drawLine(5,110,380,110);
g.drawString("1",2,14);
g.drawString("-1",14,210);
g.drawString("0",13,123);
g.drawString("90",100,123);
g.drawString("180",190,123);
g.drawString("270",280,123);
g.drawString("360",370,123);
```

Penjelasan masing-masing baris sintaks di atas adalah sebagai berikut,

```
g.drawString("1",2,14);
```

Adalah sintaks yang digunakan untuk menuliskan string pada argumen pertama ke dalam gambar, argumen kedua dan ketiga digunakan untuk menentukan posisinya pada frame.

Bagian untuk menampilkan grafik sinus adalah :

```
g.setColor(Color.red);
for(int i=0; i<360; i+=1)
{
    int y=(int) (Math.sin(i*Math.PI/180)*100)*-1;
    g.drawOval(i+10,y+110,1,1);
}
```

Perulangan `for(int i=0; i<360; i+=1)` diperlukan karena kita ingin menampilkan mulai grafik mulai dari  $0^\circ$  sampai  $360^\circ$ . Method `sin()` memiliki argumen dalam bentuk radian sehingga perlu dilakukan konversi ke bentuk derajat dengan mengalikan setiap bilangan dengan  $\pi/180$ . Hasil dari perkalian ini dikalikan dengan 100 supaya skalanya lebih besar. Selanjutnya hasil perhitungan tersebut dikalikan dengan -1 agar gambar grafik menemukan posisi yang benar. Perlu diingat bahwa koordinat grafis agak berbeda dengan koordinat kartesius yang biasa. Titik (0,0) pada koordinat grafis terletak di ujung kiri atas dari layar monitor. Bila tidak kita kalikan dengan -1 maka tampilan yang diperoleh posisinya akan terbalik.

Proses terakhir adalah type casting yang dilakukan untuk memaksa hasil perhitungan menjadi bilangan bulat sehingga dapat digunakan sebagai argumen bagi method `g.drawOval()` yang membutuhkan argumen bertipe integer.

Program di atas akan langsung menampilkan grafik sinus ketika program dijalankan. Kita dapat menambahkan satu baris kode di dalam proses plotting untuk memunculkan efek animasi. Baris kode berikut :

```
try
{
    Thread.sleep(10);
}
catch(Exception e)
{
}
```

Perlu ditambahkan pada saat plotting grafik sinus sehingga kode pada bagian proses menampilkan gambar akan berubah menjadi :

```
g.setColor(Color.red);
for(int i=0; i<360; i+=1)
{
    int y=(int) (Math.sin(i*Math.PI/180)*100)*-1;
    g.drawOval(i+10,y+110,1,1);

    try
    {
        Thread.sleep(10);
    }
    catch(Exception e)
    {
    }
}
```

Untuk plotting grafik cosinus dilaksanakan dengan cara yang sama, hanya mengubah sin menjadi cos.

Perubahan akibat penambahan kode di atas akan terlihat pada saat program dijalankan. Bila sebelumnya gambar grafik langsung muncul, maka sekarang akan muncul satu-satu yaitu grafik sinus terlebih dulu yang berjalan dari  $0^\circ$  sampai  $360^\circ$  dilanjutkan dengan grafik cosinus yang juga bergerak dari  $0^\circ$  sampai  $360^\circ$ . Program lengkapnya adalah sebagai berikut:

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



```

class KanvasAnimasi extends Canvas
{
    public void paint(Graphics g)
    {
        g.setColor(Color.black);
        g.drawLine(10,10,10,210);
        g.drawLine(5,110,380,110);
        g.drawString("1",2,14);
        g.drawString("-1",14,210);
        g.drawString("0",13,123);
        g.drawString("90",100,123);
        g.drawString("180",190,123);
        g.drawString("270",280,123);
        g.drawString("360",370,123);

        g.setColor(Color.red);
        for(int i=0; i<360; i+=1)
        {
            int y=(int) (Math.sin(i*Math.PI/180)*100)*-1;
            g.drawOval(i+10,y+110,1,1);
            try
            {
                Thread.sleep(10);
            }
            catch(Exception e)
            {
            }
        }

        g.setColor(Color.blue);
        for(int i=0; i<360; i+=1)
        {
            int y=(int) (Math.cos(i*Math.PI/180)*100)*-1;
            g.drawOval(i+10,y+110,1,1);
            try
            {
                Thread.sleep(10);
            }
            catch(Exception e)
            {
            }
        }
    }
}

```

```

}

class Animasi extends JFrame
{
    KanvasAnimasi gambar=new KanvasAnimasi();

    Animasi()
    {
        setTitle("Plotting Fungsi Sinus dan Cosinus ");
        setLocation(200,100);
        setSize(400,260);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void KomponenVisual()
    {
        getContentPane().setLayout(new BorderLayout());
        getContentPane().add(gambar, BorderLayout.CENTER);
        setVisible(true);
    }

    public static void main (String[] args)
    {
        Animasi anime=new Animasi();
        anime.KomponenVisual();
    }
}

```

Contoh berikut adalah sebuah permainan sederhana tentang bola yang dapat memantul dan permainan akan berhenti ketika kita tidak mampu menahannya jatuh. Gerakan bola akan semakin cepat ketika kita telah berhasil menahan jatuh bola pada point tertentu.

```

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

class GambarBolaAnimasi4 extends Canvas
{
    int x;
    int y;
    int dX;
    int dY;
}

```

```

boolean move=true;
int posisi;
int point;
String pointanda="";
int level=0;

public GambarBolaAnimasi4()
{
    x=200;
    y=100;
    dX=5;
    dY=5;
    setBackground(Color.white);
}

public void paint(Graphics g)
{
    g.setColor(Color.blue);
    g.drawOval(x,y,10,10);
    if(move)
    {
        x+=dX;
        y+=dY;
        if(x<0)
        {
            dX=5+level;
        }
        else if(x+10>getWidth())
        {
            dX=-5-level;
        }
        if(y<0)
        {
            dY=5+level;
        }
        else if(y+10>getHeight())
        {
            dY=-5-level;
        }
        if(posisi==1 && y>130 && x>150)
        {
            move=false;
            JOptionPane.showMessageDialog(null,"Selamat, Point anda "+
            point,"Informasi",JOptionPane.INFORMATION_MESSAGE);
        }
    }
}

```

```

        System.exit(0);
    }
    if(posisi==2 && y>130 && x<150)
    {
        move=false;
        JOptionPane.showMessageDialog(null,"Selamat, Point anda "+
            point,"Informasi",JOptionPane.INFORMATION_MESSAGE);
        System.exit(0);
    }
    if((posisi==1 || posisi==2) && y>130)
    {
        point=point+1;
    }

    }
    String pointanda=Integer.toString(point);
    System.out.println(pointanda);
    if(point>5)
    {
        level=5;
    }
}

class ProsesBolaAnimasi4 extends Thread
{
    GambarBolaAnimasi4 picture;

    public ProsesBolaAnimasi4(GambarBolaAnimasi4 obyekgambar)
    {
        this.picture=obyekgambar;
    }

    public void run()
    {
        while(true)
        {
            picture.repaint();
            try
            {
                sleep(50);
            }
            catch(Exception e)

```

```

    {
    }
}
}

public class Animasi4 extends JFrame
{
    GambarBolaAnimasi4 gambar=new GambarBolaAnimasi4();
    ProsesBolaAnimasi4 proses=new ProsesBolaAnimasi4(gambar);
    JButton kiri=new JButton("<");
    JButton kanan=new JButton(">");
    JLabel lblPoint=new JLabel("");

    public Animasi4()
    {
        super("Game Sederhana dengan Level");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocation(100,100);
        setSize(310,220);
        proses.start();
    }

    void komponenVisual()
    {
        getContentPane().setLayout(null);
        getContentPane().add(gambar);
        gambar.setBounds(0,0,300,140);
        getContentPane().add(kiri);
        kiri.setBounds(0,140,150,25);
        getContentPane().add(kanan);
        kanan.setBounds(150,140,150,25);
        getContentPane().add(lblPoint);
        lblPoint.setBounds(150,165,150,20);
        setVisible(true);
    }

    void aksiReaksi()
    {
        kanan.addKeyListener(new KeyAdapter()
        {
            public void keyPressed(KeyEvent k)
            {
                if(k.getKeyCode()==k.VK_RIGHT)

```

```

        {
            gambar.move=true;
            kiri.setVisible(true);
            kiri.requestFocus(true);
            kanan.setVisible(false);
            gambar.posisi=1;
        }
    }
});

kiri.addKeyListener(new KeyAdapter()
{
    public void keyPressed(KeyEvent k)
    {
        if(k.getKeyCode()==k.VK_LEFT)
        {
            gambar.move=true;
            kanan.setVisible(true);
            kanan.requestFocus(true);
            kiri.setVisible(false);
            gambar.posisi=2;
        }
    }
});
}

public static void main(String args[])
{
    Animasi4 a4=new Animasi4();
    a4.komponenVisual();
    a4.aksiReaksi();
}
}

```

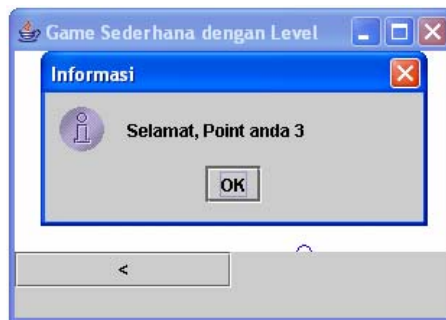
Program pada saat dijalankan akan tampil gambar berikut :

Pada kondisi ini bola akan teris memantul dan permainan belum dimulai. Permainan baru dimulai ketika kita memilih salah satu tombol kiri atau kanan dan pada saat itu keberhasilan kita dalam menahan jatuhnya bola mulai dihitung.



Gambar 5.4 Contoh permainan sederhana

Ketika pada suatu saat kita tidak berhasil menahan jatuhnya bola, maka akan muncul pesan keberhasilan kita.



Gambar 5.5. Contoh permainan selesai

Agar permainan berhenti perlu diberikan constraint atau aturan. Dalam program di atas aturan dinyatakan dengan baris-baris kode program berikut :

```

if(posisi==1 && y>130 && x>150)
{
    move=false;
    JOptionPane.showMessageDialog(null,"Selamat, Point anda "+
    point,"Informasi",JOptionPane.INFORMATION_MESSAGE);
    System.exit(0);
}

if(posisi==2 && y>130 && x<150)
{
    move=false;
    JOptionPane.showMessageDialog(null,"Selamat, Point anda "+
    point,"Informasi",JOptionPane.INFORMATION_MESSAGE);
    System.exit(0);
}

```

yang menunjukkan bahwa bila posisi batang penahan disebelah kiri yang disimbolkan dengan angka 1 dan koordinat  $y > 130$  dan koordinat  $x > 150$  maka program akan berhenti. Aturan yang lain adalah bila posisi batang penahan berada di kanan dan koordinat  $y > 130$  dan koordinat  $x < 150$  maka program juga akan berhenti.

Kode program untuk menghitung point keberhasilan menahan jatuhnya bola dinyatakan dengan :

```
if((posisi==1 || posisi==2) && y>130)
{
    point=point+1;
}
```

Yang berarti bahwa ketika batang penahan berada di posisi 1 atau posisi 2 dan  $y > 30$  maka kondisi tersebut akan dihitung sebagai keberhasilan dan menjadi point dalam permainan ini. Selanjutnya point ini dapat digunakan untuk mengetahui seberapa jauh penguasaan pemain terhadap permainan ini dan dapat kita gunakan untuk meningkatkan derajat kesulitan dari permainan. Dalam contoh ini, bila pemain telah dapat menahan jatuhnya bola sebanyak lima kali maka otomatis gerakan bola akan menjadi lebih cepat. Pekerjaan ini ditangani oleh baris kode berikut :

```
String pointanda=Integer.toString(point);
System.out.println(pointanda);

if(point>5)
{
    level=5;
}
```

Dimana variabel level ini akan diberikan sebagai faktor penjumlahan pada kode program yang berurusan dengan gerakan bola, yaitu :

```
x+=dX;
y+=dY;

if(x<0)
{
    dX=5+level;
}
else if(x+10>getWidth())
```



```

    {
        dX=-5-level;
    }

    if(y<0)
    {
        dY=5+level;
    }
    else if(y+10>getHeight())
    {
        dY=-5-level;
    }

```

## 5.2. BEKERJA DENGAN CITRA

Dalam pengolahan citra dan visi komputer kita akan bekerja dengan gambar atau citra. Pengolahan citra berhubungan dengan proses untuk mengubah citra menjadi citra lain sesuai dengan keperluan, sementara visi komputer berupaya mendapatkan informasi tertentu dari sebuah citra.

Langkah pertama dalam pengolahan citra adalah menangkap gambar dan menyimpannya sebagai data. Setelah proses tersebut, proses pengolahan citra dapat dilakukan dengan menggunakan berbagai algoritma pengolahan citra. Berikut ini adalah cara menangkap dan menampilkan citra dalam pemrograman Java.

```

import javax.swing.*;
import java.awt.*;
import java.awt.image.*;

class KanvasCitra3 extends Canvas
{
    public void paint(Graphics grafis)
    {
        Graphics2D g = (Graphics2D) grafis;

        Image image=new ImageIcon("bungaku.jpg").getImage();
        g.drawImage(image,0,2,this);
    }
}

```

```

class AmbilCitra3 extends JFrame
{
    KanvasCitra3 gambar=new KanvasCitra3();

    public AmbilCitra3()
    {
        setTitle("Dasar Pengolahan Citra");
        setLocation(0,0);
        setSize(176,317);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void komponenVisual()
    {
        getContentPane().setLayout(new BorderLayout());
        getContentPane().add(gambar,BorderLayout.CENTER);
        setVisible(true);
    }

    public static void main(String args[])
    {
        AmbilCitra3 c3=new AmbilCitra3();
        c3.komponenVisual();
    }
}

```

Hasil dari program di atas adalah sebagai berikut:



Gambar 5.6. Contoh menampilkan citra

Penjelasan program di atas adalah sebagai berikut:

Seperti juga program-program grafis sebelumnya, program ini terdiri dari dua class yaitu class KanvasCitra3 dan class AmbilCitra3. Dalam class KanvasCitra3 terdapat sintaks :

```
Image image=new ImageIcon("bungaku.jpg").getImage()
```

Yang digunakan untuk mengambil citra dari sebuah file.

```
g.drawImage(image,0,2,this)
```

Digunakan untuk menampilkan gambar ke dalam frame.

Program berikutnya adalah contoh pengolahan citra untuk melakukan proses deteksi tepi dan penajaman citra.

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

class KanvasCitra7 extends Canvas
{
    public void paint(Graphics grafis)
    {
        Graphics2D g = (Graphics2D) grafis;

        BufferedImage bi=null;
        try
        {
            Image image=new ImageIcon("bungaku.jpg").getImage();
            g.drawImage(image,0,2,this);

            bi=new BufferedImage(image.getWidth(null),
                image.getHeight(null), BufferedImage.TYPE_INT_RGB);
            Graphics2D g2=bi.createGraphics();
            g2.drawImage(image,null,null);

            float[] edgeKernel={0,-1,0,
                                -1,4,-1,
                                0,-1,0};
            BufferedImageOp edgeop=new ConvolveOp (new Kernel(3, 3,
                edgeKernel), ConvolveOp.EDGE_NO_OP, null);
            BufferedImage image3=edgeop.filter(bi,null);
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

```

        g.drawImage(image3,170,2,this);

        float[] sharpKernel={0,-1,0,
                               -1,5,-1,
                               0,-1,0};
        BufferedImageOp sharpop=new ConvolveOp(new Kernel(3, 3,
        sharpKernel), ConvolveOp.EDGE_NO_OP, null);
        BufferedImage image4=sharpop.filter(bi,null);
        g.drawImage(image4,340,2,this);
    }
    catch (Exception e)
    {
        System.out.println(e);
    }
}

class Citra7 extends JFrame
{
    KanvasCitra7 gambar=new KanvasCitra7();

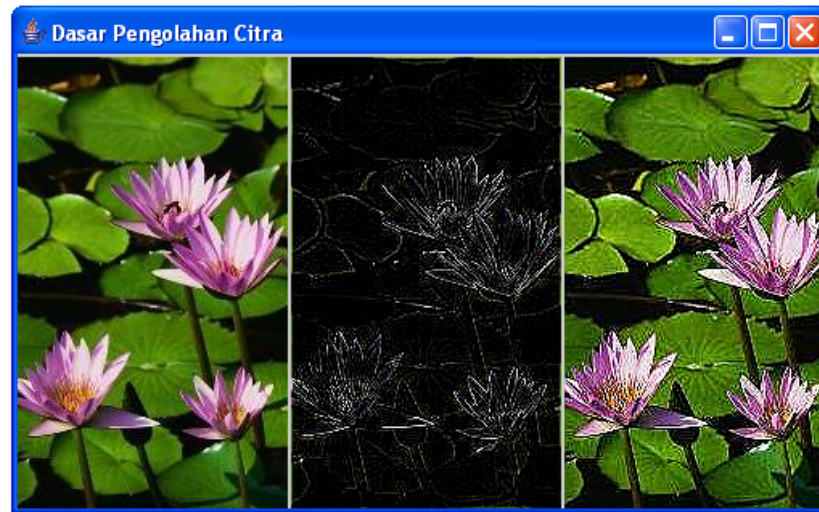
    public Citra7()
    {
        setTitle("Dasar Pengolahan Citra");
        setLocation(0,0);
        setSize(510,317);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    void komponenVisual()
    {
        getContentPane().setLayout(new BorderLayout());
        getContentPane().add(gambar,BorderLayout.CENTER);
        setVisible(true);
    }

    public static void main(String args[])
    {
        Citra7 c7=new Citra7();
        c7.komponenVisual();
    }
}

```

Hasil dari program di atas adalah sebagai berikut:

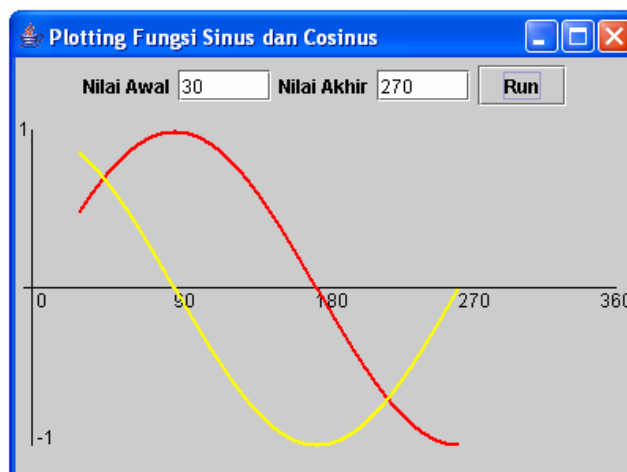


Gambar 5.7. Contoh pengolahan citra

Dasar pengolahan citra pada buku ini sifatnya dasar dan hanya berupa contoh-contoh dari pengolahan citra. Untuk mempelajari lebih lanjut dapat dibaca pada materi pengolahan citra.

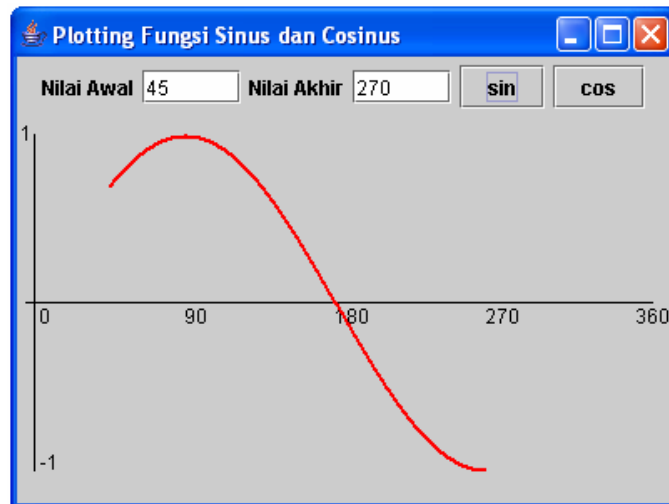
### 5.3. Latihan

- (1) Kembangkan program Animasi.java di depan dengan menambahkan batas awal dan batas akhir yang dapat dimasukkan secara interaktif melalui keyboard. Contoh tampilannya adalah sebagai berikut :



Gambar 5.8. Contoh tampilan latihan 1

- (2) Ubahlah program di atas dengan menambahkan tombol sinus dan cosinus dimana bila diklik tombol sinus maka grafik sinus yang akan ditampilkan dan bila diklik tombol cosinus maka grafik cosinus yang akan ditampilkan. Berikut adalah ilustrasi output yang diharapkan.



Gambar 5.9. Contoh tampilan latihan 2

## BAB 6

### KONEKSI DATABASE

---

**Tujuan:**

1. *Mahasiswa dapat menjelaskan konsep dasar pemrograman database pada bahasa Java.*
  2. *Mahasiswa dapat melakukan koneksi database*
  3. *Mahasiswa dapat membuat program untuk mengelola data pada suatu database.*
- 

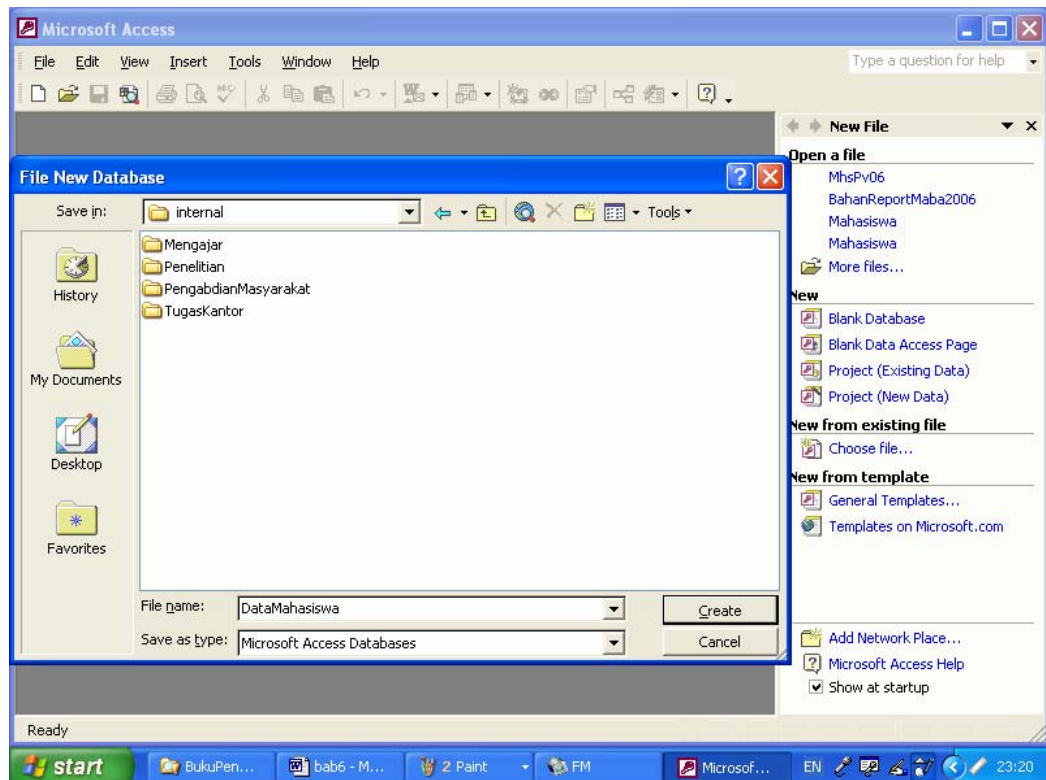
Pemrograman database merupakan aplikasi dasar dalam sistem informasi. Bagian ini merupakan dasar bagi pemrograman database lebih lanjut, terdiri dari bagaimana membuat database, menghubungkan program dengan database dan membuat aplikasi database sederhana.

Ada beberapa proses dasar di dalam pemrograman database, di antaranya adalah:

- koneksi database yang suatu teknik untuk menghubungkan aplikasi dengan sebuah DBMS (DataBase Management System).
- Menampilkan data dari database menggunakan query
- Mengelola data, seperti menambah data, menghapus data dan memperbaiki data.

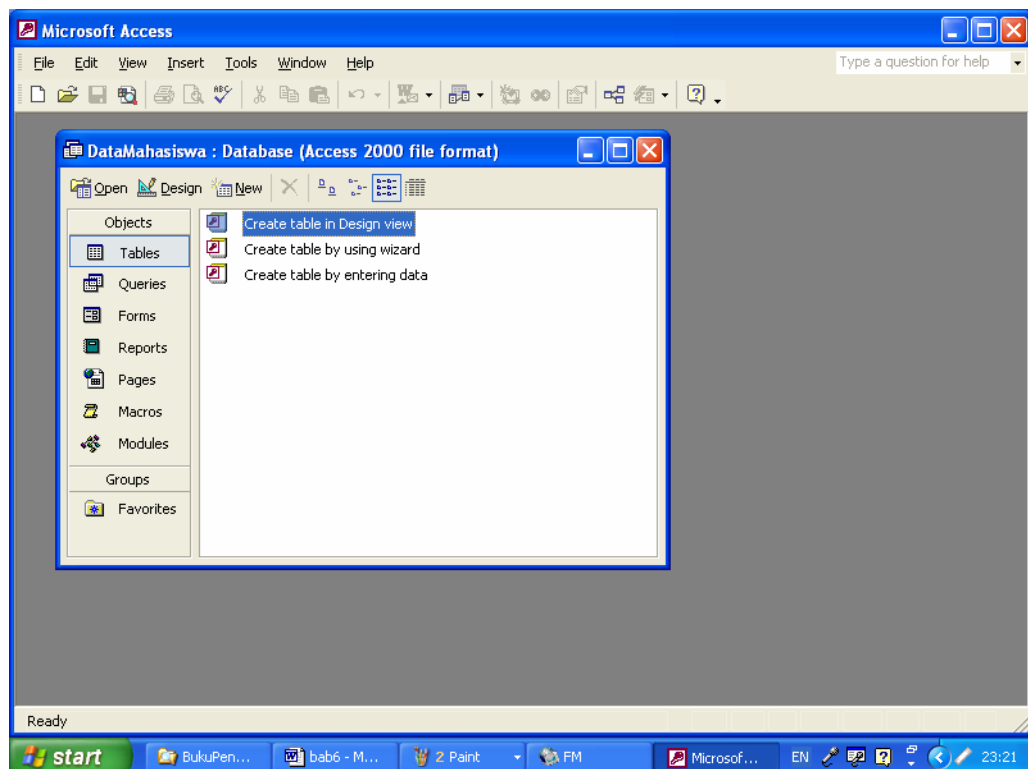
#### **6.1. Membangun Database**

Database adalah tempat semua data diletakkan dalam sistem komputer. Banyak DBMS yang dapat digunakan seperti MS Access atau mysql. Dalam contoh ini database dibangun dengan MS Access. Langkah pertama pembuatan database adalah dengan membuka aplikasi MS Access, memilih blank database dan membuat file database baru seperti berikut:



Gambar 6.1. Membuat file database

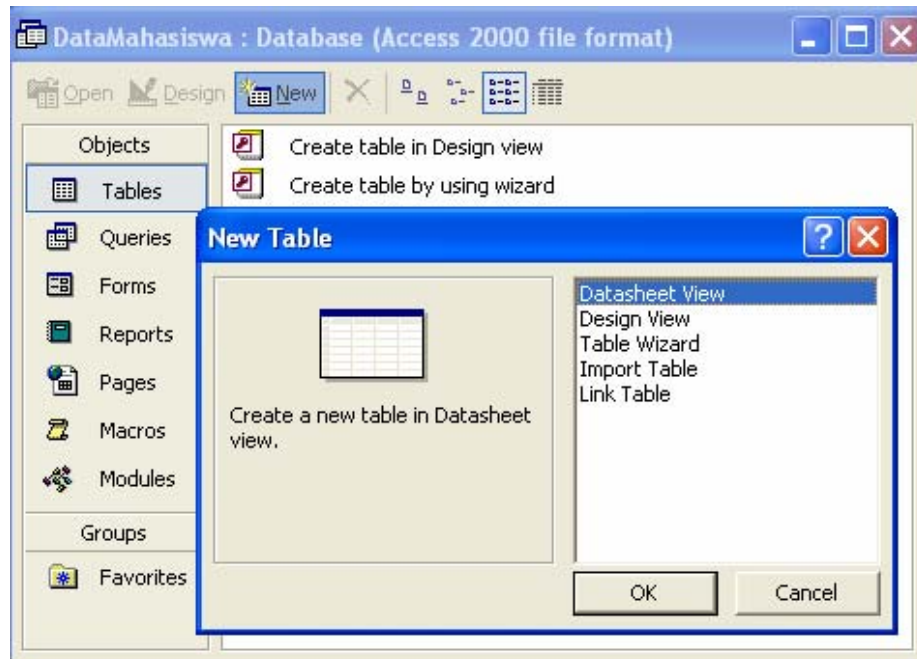
Klik create, akan muncul tampilan berikut:



Gambar 6.2. Tampilan awal database Ms. Access

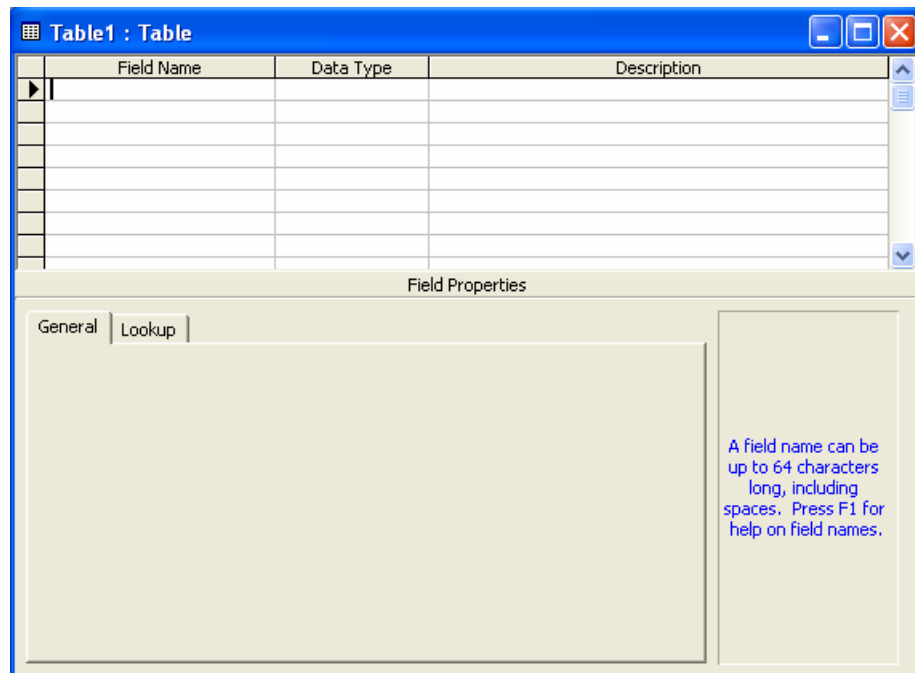


Klik New, pilih Design New, klik OK



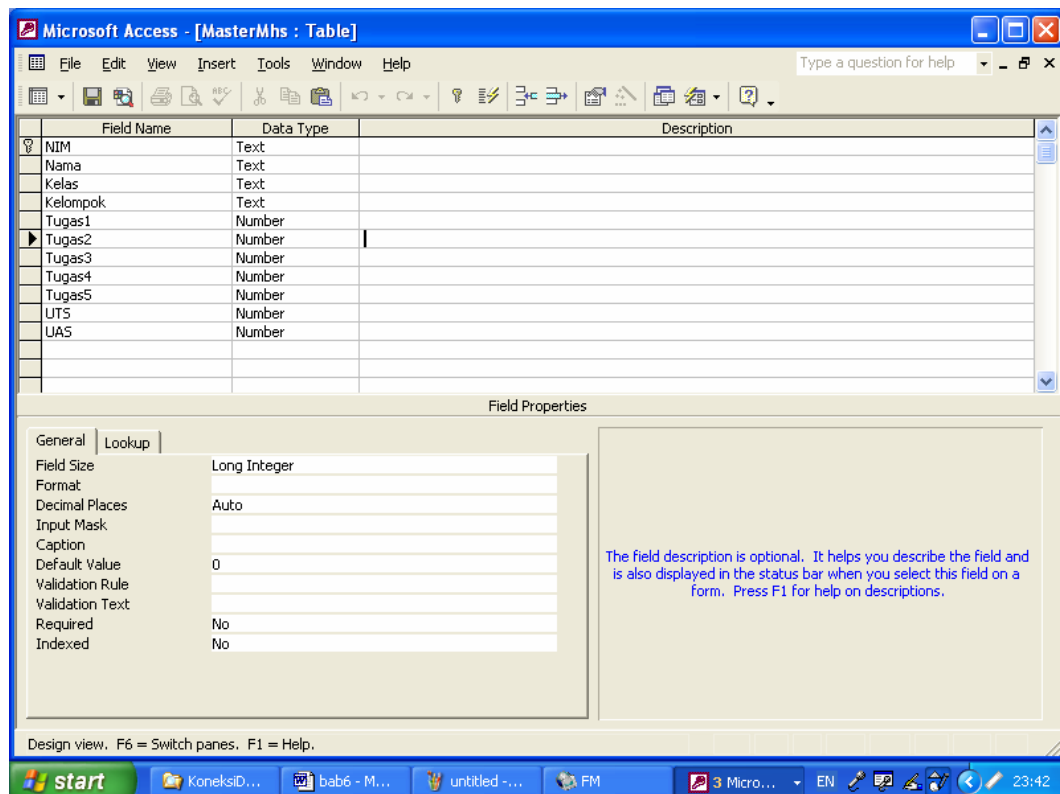
Gambar 6.3. Membuat table

Setelah itu akan muncul tampilan berikut :



Gambar 6.4. Pengisian attribut pada tabel

Pada tabel di atas kita isi dengan field dan tipe data yang akan kita gunakan sebagai berikut:



Gambar 6.5. Setelah pengisian atribut (nama kolom pada tabel)

Beri nama tabel tersebut dan simpan. Sampai disini database telah terbentuk dan siap diisi dengan data.

Pada langkah-langkah di atas terlihat bahwa tabel dibuat dengan atribut-atribut dan jenisnya adalah sebagai berikut:

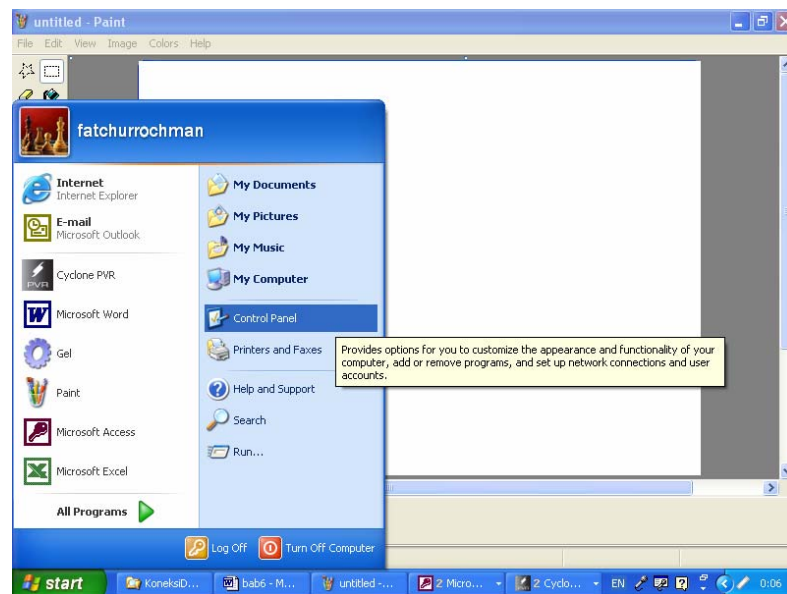
No.	Nama atribut	Format data
1.	NIM	Text
2.	Nama	Text
3.	Kelas	Text
4.	Kelompok	Text
5.	Tugas1	Number
6.	Tugas2	Number
7.	Tugas3	Number
8.	Tugas4	Number
9.	Tugas5	Number
10.	UTS	Number
11.	UAS	Number

## 6.2. Membuat ODBC

ODBC merupakan aturan digunakan untuk mengakses sebuah database. Dengan menggunakan ODBC ini maka akses sebuah database dapat dilakukan cukup dengan menggunakan nama yang kita kehendaki.

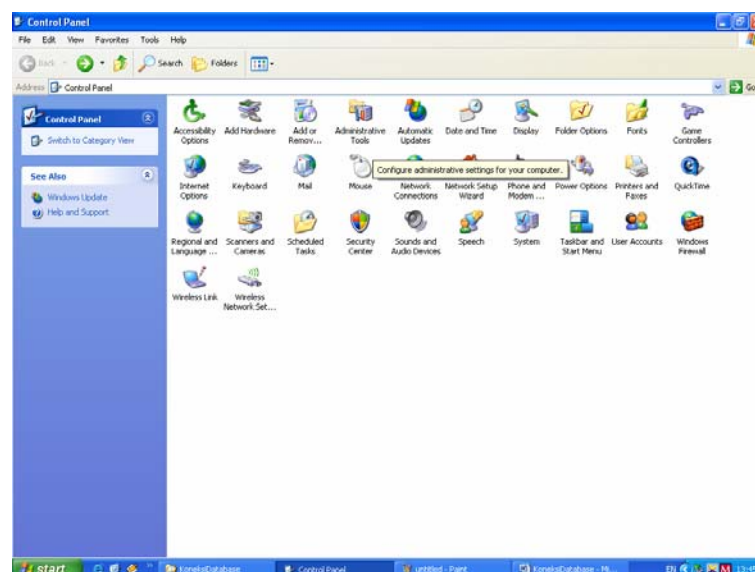
Cara membuat ODBC adalah dengan masuk ke control panel, masuk ke Administrative tool, memilih data source dan membuat nama ODBC berdasar driver database yang digunakan.

Pilih Control Panel



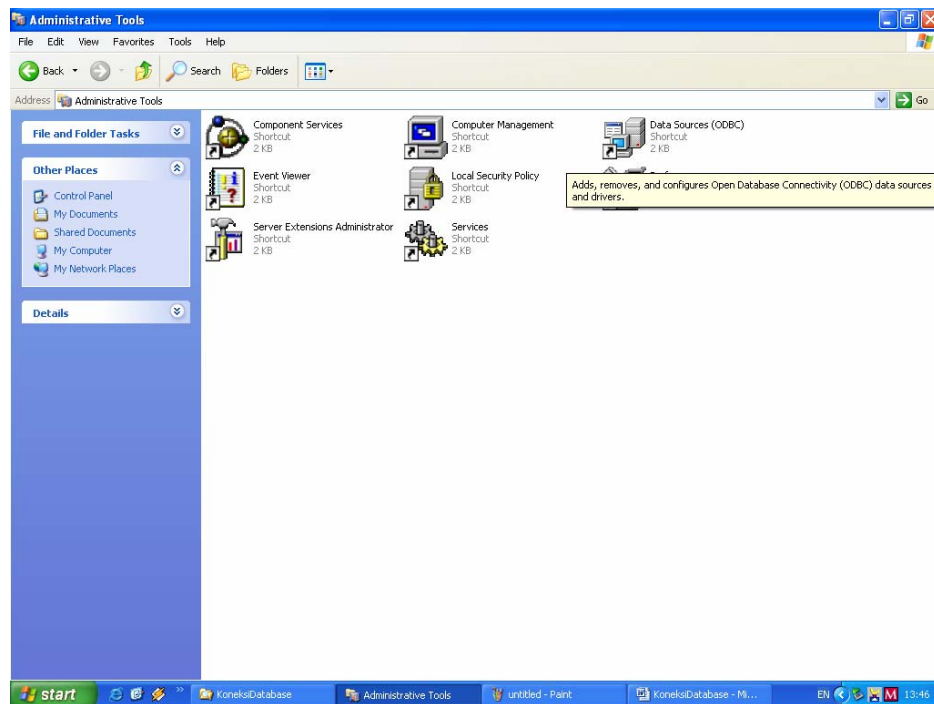
Gambar 6.6. Shortcut menu Control Panel pada Windows

Tampilan Control Panel sebagai berikut:



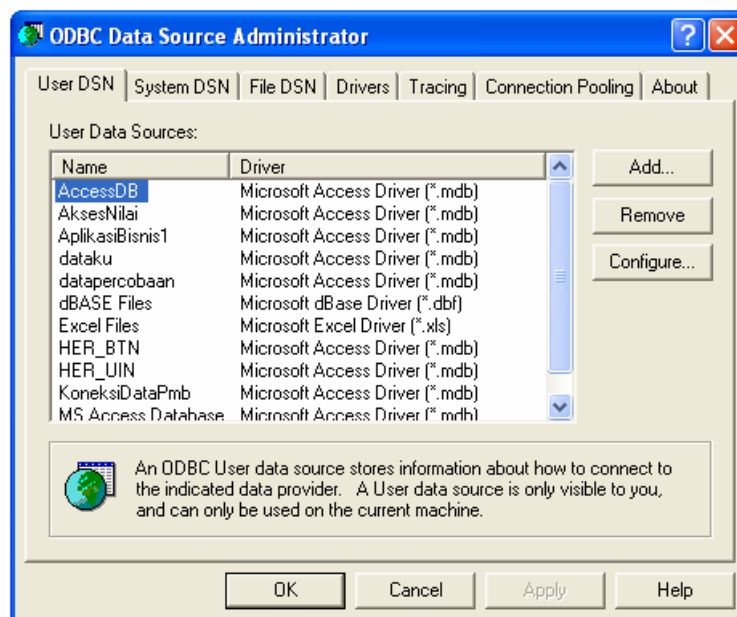
Gambar 6.7. Tampilan control panel

Pilih Administrative Tool pada control panel sehingga muncul tampilan berikut:



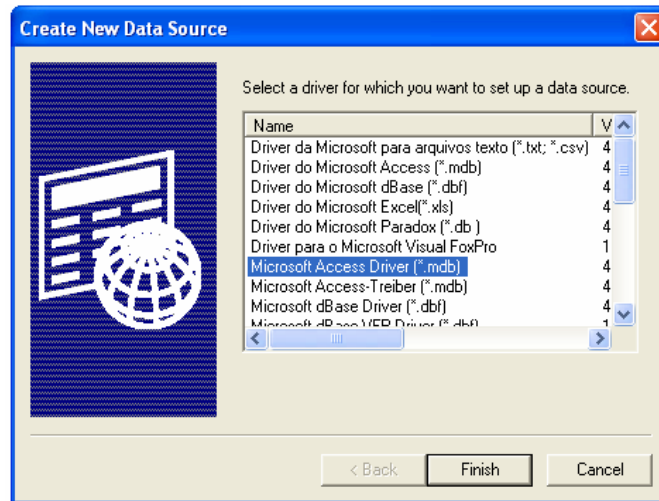
Gambar 6.8. Tampilan administrative tool

Pilih Data Source dan akan muncul tampilan Data Source lalu klik Add untuk menambahkan User Data Source



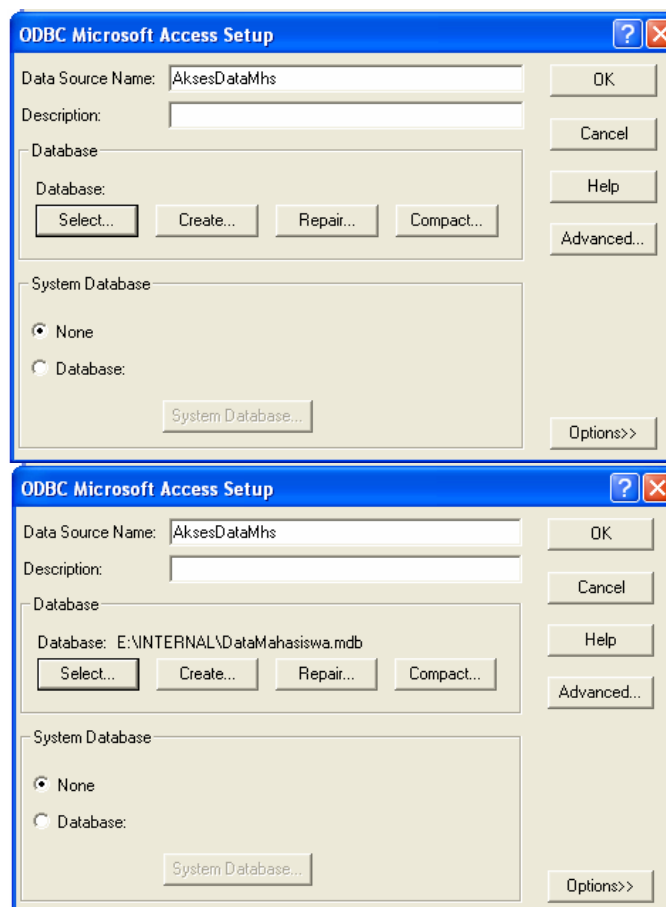
Gambar 6.9. Pemilihan user DSN

Menentukan driver database yang digunakan



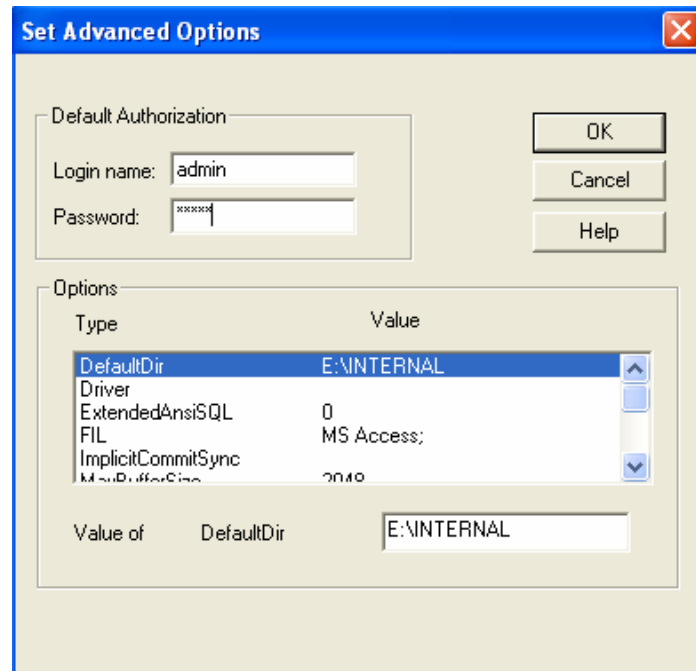
Gambar 6.10. Penentuan driver database

Memberi nama Data Source dan menentukan lokasi database yang digunakan dengan menekan tombol select.



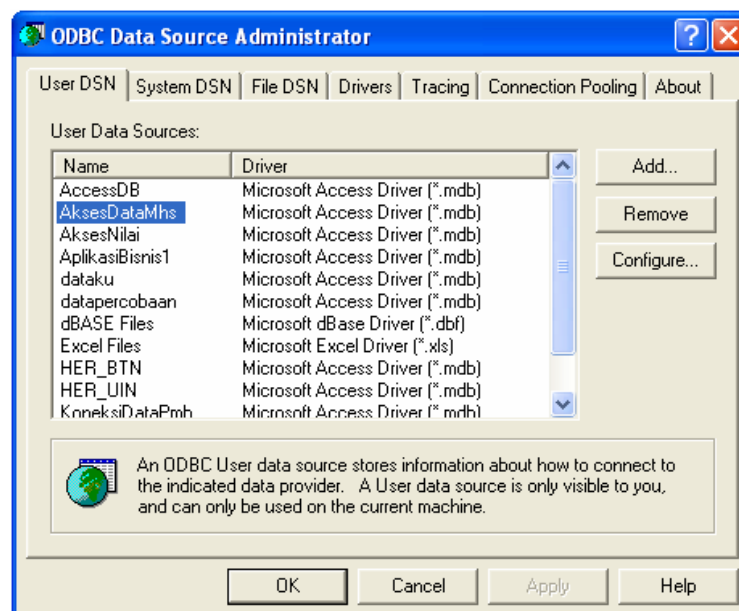
Gambar 6.11. Pembuatan DSN

Klik tombol advanced untuk mengisi login name dan password.



Gambar 6.12. Pengisian login dan password untuk DSN

Hasil akhir pembuatan ODBC adalah muncul nama dari koneksi yang telah kita buat pada user DSN



Gambar 6.13. Hasil DSN

### 6.3. Koneksi Database

ODBC (Open DataBase Connection) adalah suatu protocol koneksi database standard yang disediakan oleh sistem operasi. Hal ini perlu dilakukan karena saat ini DBMS dan aplikasi merupakan dua hal yang berbeda. Setelah ODBC dibuat, kita dapat menghubungkan sebuah program untuk mengakses database tersebut. Cara menghubungkan aplikasi dan database ini dinamakan **koneksi database**.

Contoh program berikut ini digunakan untuk melakukan koneksi database.

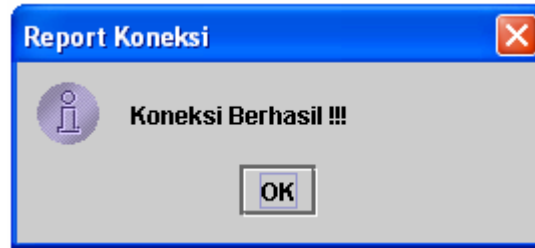
```
import javax.swing.*;
import java.awt.*;
import java.sql.*;

public class Koneksi
{
    void KoneksiDatabase()
    {
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection con = DriverManager.getConnection("jdbc:odbc:
sesDataMhs;uid='admin';pw='admin'");

            JOptionPane.showMessageDialog(null,"Koneksi Berhasil !!!","Report
oneksi",JOptionPane.INFORMATION_MESSAGE);
            con.close();
        }
        catch(Exception e)
        {
            System.err.println("Exception: " + e.getMessage());
        }
    }

    public static void main(String args[])
    {
        Koneksi konek=new Koneksi();
        konek.KoneksiDatabase();
        System.exit(0);
    }
}
```

Hasil dari program di atas adalah sebagai berikut:



Gambar 6.14. Tampilan konfirmasi koneksi database

## 6.4. Memasukkan Data ke Database

Langkah berikutnya adalah membuat aplikasi sederhana yang berhubungan dengan database. Berikut ini adalah beberapa aplikasi sederhana dalam berhubungan dengan database, diantaranya bagaimana memasukkan data ke dalam database, mencari data dan mengedit data.

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

class AplikasiPenilaian extends JFrame
{
    JLabel lblnama=new JLabel("Nama");
    JTextField txnama=new JTextField(20);
    JLabel lblnim=new JLabel("NIM ");
    JTextField txnim=new JTextField(20);
    JButton tblcari=new JButton("Cari");
    JLabel lblkelas=new JLabel("Kelas ");
    JRadioButton kelasA=new JRadioButton("A");
    JRadioButton kelasB=new JRadioButton("B");
    JRadioButton kelasC=new JRadioButton("C");
    ButtonGroup grupkelas=new ButtonGroup();
    JLabel lblkelompok=new JLabel("Kelompok");
    String[] jeniskelompok={"1","2","3","4","5","6","7"};
    JComboBox cbkelompok=new JComboBox(jeniskelompok);
    JLabel lblnilai=new JLabel("Nilai ");
    JLabel lbltugas1=new JLabel("Tugas1");
    JTextField txtugas1=new JTextField(10);
    JLabel lbltugas2=new JLabel("Tugas2");
    JTextField txtugas2=new JTextField(10);
    JLabel lbltugas3=new JLabel("Tugas3");
    JTextField txtugas3=new JTextField(10);
}
```



```

JLabel lbltugas4=new JLabel("Tugas4");
JTextField txtugas4=new JTextField(10);
JLabel lbltugas5=new JLabel("Tugas5");
JTextField txtugas5=new JTextField(10);
JLabel lbluts=new JLabel("UTS");
JTextField txuts=new JTextField(10);
JLabel lbluas=new JLabel("UAS");
JTextField txuas=new JTextField(10);
JButton tblsimpan=new JButton("Save");
JButton tblupdate=new JButton("Update");
JButton tblkeluar=new JButton("Exit");

String nim="";
String nama="";
String kelas="";
String kelompok="";
String tugas1="";
String tugas2="";
String tugas3="";
String tugas4="";
String tugas5="";
String uts="";
String uas="";
String cari="";

AplikasiPenilaian()
{
    txtugas1.setText("0");
    txtugas2.setText("0");
    txtugas3.setText("0");
    txtugas4.setText("0");
    txtugas5.setText("0");
    txuts.setText("0");
    txuas.setText("0");
    setTitle("Lembar Penilaian");
    setLocation(300,100);
    setSize(300,320);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

void komponenVisual()
{
    getContentPane().add(lblnim);
    lblnim.setBounds(10,10,70,20);

```

```

getContentPane().add(txnim);
txnim.setBounds(75,10,100,20);
getContentPane().add(tblcari);
tblcari.setBounds(180,10,95,20);
getContentPane().add(lblnama);
lblnama.setBounds(10,30,70,20);
getContentPane().add(txnama);
txnama.setBounds(75,30,200,20);
getContentPane().add(lblkelas);
lblkelas.setBounds(10,50,100,20);
getContentPane().add(kelasA);
kelasA.setBounds(75,50,50,20);
getContentPane().add(kelasB);
kelasB.setBounds(125,50,50,20);
getContentPane().add(kelasC);
kelasC.setBounds(175,50,50,20);
grupkelas.add(kelasA);
grupkelas.add(kelasB);
grupkelas.add(kelasC);
getContentPane().add(lblkelompok);
lblkelompok.setBounds(10,70,100,20);
getContentPane().add(cbkelompok);
cbkelompok.setBounds(75,70,100,20);
getContentPane().add(lblnilai);
lblnilai.setBounds(10,110,100,20);
getContentPane().add(lbltugas1);
lbltugas1.setBounds(10,130,100,20);
getContentPane().add(txttugas1);
txttugas1.setBounds(75,130,100,20);
getContentPane().add(lbltugas2);
lbltugas2.setBounds(10,150,100,20);
getContentPane().add(txttugas2);
txttugas2.setBounds(75,150,100,20);
getContentPane().add(lbltugas3);
lbltugas3.setBounds(10,170,100,20);
getContentPane().add(txttugas3);
txttugas3.setBounds(75,170,100,20);
getContentPane().add(lbltugas4);
lbltugas4.setBounds(10,190,100,20);
getContentPane().add(txttugas4);
txttugas4.setBounds(75,190,100,20);
getContentPane().add(lbltugas5);
lbltugas5.setBounds(10,210,100,20);
getContentPane().add(txttugas5);

```

```

txtugas5.setBounds(75,210,100,20);
getContentPane().add(tblsimpan);
tblsimpan.setBounds(180,210,100,20);
getContentPane().add(lbluts);
lbluts.setBounds(10,230,100,20);
getContentPane().add(txuts);
txuts.setBounds(75,230,100,20);
getContentPane().add(lbluas);
lbluas.setBounds(10,250,100,20);
getContentPane().add(txuas);
txuas.setBounds(75,250,100,20);
getContentPane().add(tblupdate);
tblupdate.setBounds(180,230,100,20);
getContentPane().add(tblkeluar);
tblkeluar.setBounds(180,250,100,20);
getContentPane().setLayout(null);
setVisible(true);
}

void AksiReaksi()
{
    tblsimpan.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent event)
        {
            if (event.getSource()==tblsimpan)
            {
                try
                {
                    nim=txnim.getText();
                    nama=txnama.getText();

                    if(kelasA.isSelected()==true)
                        kelas=kelasA.getText();
                    if(kelasB.isSelected()==true)
                        kelas=kelasB.getText();
                    if(kelasC.isSelected()==true)
                        kelas=kelasC.getText();
                    kelompok=(String) cbkelompok.getSelectedItem();
                    tugas1=txtugas1.getText();
                    tugas2=txtugas2.getText();
                    tugas3=txtugas3.getText();
                    tugas4=txtugas4.getText();
                    tugas5=txtugas5.getText();

```

```

        uts=txuts.getText();
        uas=txuas.getText();

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection connection = DriverManager.getConnection("jdbc:
        odbc:AksesDataMhs;uid='admin';pw='admin'");
        Statement statement = connection.createStatement();
        String sql="insert into MasterMhs values ('"+nim+"','"+nama+"",
        '"+kelas+"','"+kelompok+"','"+tugas1+"','"+tugas2+"
        '"+tugas3+"','"+tugas4+"','"+tugas5+"','"+uts+"",
        '"+uas+"');";
        statement.executeUpdate(sql);
        statement.close();
        connection.close();
        System.out.println("Data telah masuk");
        txnim.setText("");
        txnama.setText("");
        txnim.requestFocus();
    }
    catch(Exception e)
    {
        System.out.println("Error :"+e);
    }
}
});
}

public static void main(String args[])
{
    AplikasiPenilaian ap=new AplikasiPenilaian();
    ap.komponenVisual();
    ap.AksiReaksi();
}
}

```

Hasil dari program di atas adalah sebagai berikut:

Gambar 6.15. Tampilan memasukkan data

Setelah kita klik tombol Save maka data akan masuk ke dalam database DataMahasiswa. Untuk memastikan kebenarannya kita dapat membuka database tersebut dan melihat data pada tabel MasterMhs.

	NIM	Nama	Kelas	Kelompok	Tugas1
	05110011	Abdul Malik	B	3	90
▶	06510008	Fatimah	C	5	90
*					0

Gambar 6.16. Tampilan Data

## 6.5. Mencari Data dalam Database

Untuk mencari sebuah data dari database, tambahkan event berikut di dalam method AksiReaksi().

```
tblcari.addActionListener(new ActionListener()
{
```

```

public void actionPerformed(ActionEvent e)
{
    if (e.getSource()==tblcari)
    {
        try
        {
            cari=txnim.getText();

            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection connection = DriverManager.getConnection("jdbc:odbc:
                AksesDataMhs;uid='admin';pw='admin'");
            Statement statement = connection.createStatement();
            String sql="select * from MasterMhs where NIM like '"+cari+"'";
            ResultSet rs=statement.executeQuery(sql);

            if(rs.next())
            {
                txnim.setText(rs.getString(1));
                txnama.setText(rs.getString(2));

                kelas=rs.getString(3);
                if(kelas.equals("A"))
                {
                    kelasA.setSelected(true);
                }
                else if(kelas.equals("B"))
                {
                    kelasB.setSelected(true);
                }
                else
                {
                    kelasC.setSelected(true);
                }
                cbkelompok.setSelectedItem(rs.getString(4));
                txtugas1.setText(rs.getString(5));
                txtugas2.setText(rs.getString(6));
                txtugas3.setText(rs.getString(7));
                txtugas4.setText(rs.getString(8));
                txtugas5.setText(rs.getString(9));
                txuts.setText(rs.getString(10));
                txuas.setText(rs.getString(11));
            }
            else
            {

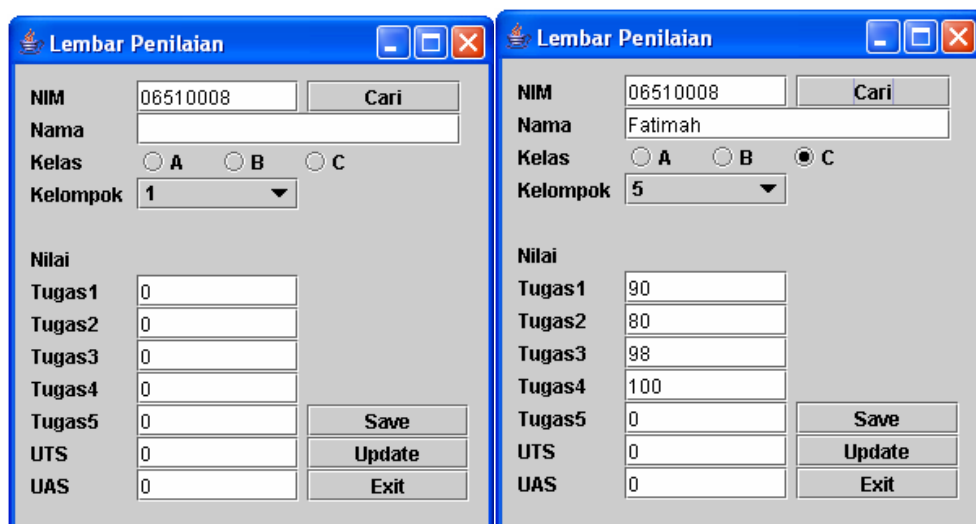
```

```

        System.out.println(nim+" tidak ada");
    }
    statement.close();
    connection.close();
}
catch(Exception ex)
{
    System.out.println("Error :"+ex);
}
}
});

```

Bila kode program di atas telah ditambahkan, compile dan jalankan program. Ketik NIM yang dicari dan klik tombol cari maka outputnya adalah sebagai berikut :



Gambar 6.17. Tampilan pencarian data sebelum dan sesudah

## 6.6. Program Update Data

Update data digunakan untuk melakukan perubahan terhadap data yang telah ada dalam database. Kita dapat menambahkan kode berikut ke dalam method AksiReaksi agar program sebelumnya dapat kita gunakan untuk update data.

```

tblupdate.addActionListener(new ActionListener()
{

```

```

public void actionPerformed(ActionEvent event)
{
    if (event.getSource()==tblupdate)
    {
        try
        {
            nim=txnim.getText();
            nama=txnama.getText();
            if(kelasA.isSelected()==true) kelas=kelasA.getText();
            if(kelasB.isSelected()==true) kelas=kelasB.getText();
            if(kelasC.isSelected()==true) kelas=kelasC.getText();
            kelompok=(String) cbkelompok.getSelectedItem();
            tugas1=txtugas1.getText();
            tugas2=txtugas2.getText();
            tugas3=txtugas3.getText();
            tugas4=txtugas4.getText();
            tugas5=txtugas5.getText();
            uts=txuts.getText();
            uas=txuas.getText();

            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection connection =
            DriverManager.getConnection("jdbc:odbc:
            AksesDataMhs;uid='admin';pw='admin'");
            Statement statement = connection.createStatement();
            String sql="update MasterMhs set nama='"+nama+"',kelas="
+kelas+"',kelompok='"+kelompok+"',tugas1='"+tugas1+"',
tugas2='"+tugas2+"',tugas3='"+tugas3+"',tugas4='"+tugas4+"',
            tugas5='"+tugas5+"',uts='"+uts+"',uas='"+uas+"
            where NIM='"+nim+"'";
            statement.executeUpdate(sql);
            statement.close();
            connection.close();
            System.out.println("Data teredit");
        }
        catch(Exception e)
        {
            System.out.println("Error :"+e);
        }
    }
}
});

```

Cara kerja program dimulai dengan memasukkan NIM mahasiswa yang akan dilakukan perubahan lalu klik tombol cari, seperti berikut:



Gambar 6.18. Update data

Kemudian ubah beberapa nilai attrbut seperti gambar 6.18 di atas. Setelah ditemukan ubah atribut yang ingin diubah, dan klik tombol update. Misalnya kita ingin mengubah kelompok menjadi kelompok 6. Untuk melihat perubahan, table Data Mahasiswa dapat kita buka menggunakan Ms. Access, dan hasilnya seperti pada gambar 6.19 berikut.

	NIM	Nama	Kelas	Kelompok	Tugas1
	05110011	Abdul Malik	C	3	90
▶	06510008	Fatimah	C	6	90
*					0

Gambar 6.19. Hasil update data

Pada hasil update data terlihat bahwa kelompok dari mahasiswa bernama Fatimah telah berubah dari 5 menjadi 6.

## 6.7. Menampilkan data Melalui Tabel

Agar tidak selalu membuka dan menutup database untuk melihat data mahasiswa, kita juga dapat membuat sebuah aplikasi yang secara spesifik digunakan untuk melihat data melalui sebuah tabel. Programnya adalah sebagai berikut :

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

class AplikasiViewData extends JFrame
{
    String[]
headers={"NIM","Nama","Kelas","Kelompok","Tugas1","Tugas2","Tugas3",
"Tugas4","Tugas5","UTS","UAS"};
    Object[][] data=new Object[0][0];
    JTable tableView;
    int n;
    AplikasiViewData()
    {
        super("View Data");
        setLocation(200,100);
        setSize(300,100);
        setDefaultCloseOperation(JFrame.HIDE_ON_CLOSE);
    }

    void KomponenVisual()
    {
        tableView=new JTable(data,headers);
        JScrollPane scrollpane=new JScrollPane(tableView);
        scrollpane.setPreferredSize(new Dimension(500,80));
        getContentPane().setLayout(new BorderLayout());
        getContentPane().add(BorderLayout.CENTER,scrollpane);
        pack();
        setVisible(true);
    }

    void KoneksiDatabase()
    {
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

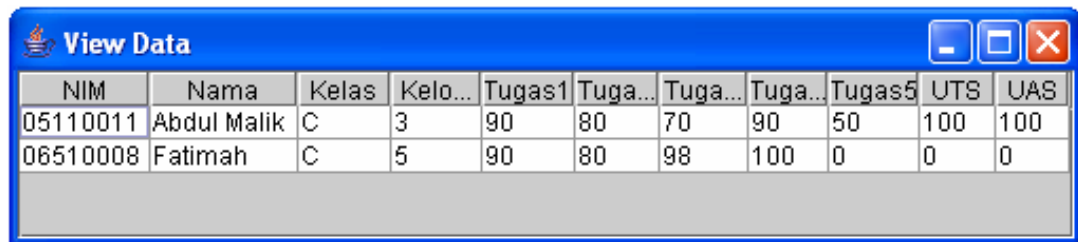
```

        Connection
connection=DriverManager.getConnection("jdbc:odbc:AksesDataMhs;uid
='admin';pw='admin'");
        Statement
state=connection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIV
E,ResultSet.CONCUR_READ_ONLY);
        String sql="select * from MasterMhs";
        ResultSet rs=state.executeQuery(sql);
        rs.last();
        n=rs.getRow();
        data=new Object[n][11];
        int p=0;
        rs.beforeFirst();
        while(rs.next())
        {
            data[p][0]=rs.getString(1);
            data[p][1]=rs.getString(2);
            data[p][2]=rs.getString(3);
            data[p][3]=rs.getString(4);
            data[p][4]=rs.getString(5);
            data[p][5]=rs.getString(6);
            data[p][6]=rs.getString(7);
            data[p][7]=rs.getString(8);
            data[p][8]=rs.getString(9);
            data[p][9]=rs.getString(10);
            data[p][10]=rs.getString(11);
            p++;
        }
        state.close();
        connection.close();
    }
    catch(Exception DBException)
    {
        System.err.println("Error : "+DBException);
    }
}

public static void main(String args[])
{
    AplikasiViewData td=new AplikasiViewData();
    td.KoneksiDatabase();
    td.KomponenVisual();
}
}

```

Hasil program di atas adalah sebagai berikut:



The screenshot shows a Java Swing window titled "View Data" with a standard Mac OS X-style title bar (red, yellow, and green buttons). Inside the window is a table with 11 columns and 2 data rows. The columns are labeled: NIM, Nama, Kelas, Kelo..., Tugas1, Tuga..., Tuga..., Tuga..., Tugas5, UTS, and UAS. The first row contains the data: 05110011, Abdul Malik, C, 3, 90, 80, 70, 90, 50, 100, 100. The second row contains: 06510008, Fatimah, C, 5, 90, 80, 98, 100, 0, 0, 0.

NIM	Nama	Kelas	Kelo...	Tugas1	Tuga...	Tuga...	Tuga...	Tugas5	UTS	UAS
05110011	Abdul Malik	C	3	90	80	70	90	50	100	100
06510008	Fatimah	C	5	90	80	98	100	0	0	0

Gambar 6.20. Contoh tampilan data dalam bentuk tabel

Agar aplikasi View Data tersebut dapat kita integrasikan dengan program sebelumnya, kita perlu menambahkan tombol baru yang akan kita gunakan untuk menampilkan semua data ketika di klik. Pada tombol tersebut kita berikan event berikut yang ditelakkan dalam method AksiReaksi().

```
tblview.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        AplikasiViewData vd=new AplikasiViewData();

        vd.KoneksiDatabase();
        vd.KomponenVisual();
    }
});
```