MODUL PRAKTIKUM

PEMROGRAMAN BERBASIS GUI



UNIVERSITAS TEKNOLOGI MATARAM

2019

Jl. Kampus Universitas Teknologi Mataram, Kekalik, Mataram

DAFTAR ISI

BAB 1 1
DASAR PEMROGRAMAN JAVA 1
BAB 2 11
INPUT DARI KEYBOARD 11
BAB 3 15
STRUKTUR KONTROL 15
BAB 4 26
ARRAY 26
BAB 5 30
KELAS DAN OBJEK 30
BAB 6 37
PEWARISAN, POLIMORFISME, INTERFACE 37
BAB 7 49
EXCEPTION HANDLING 49
BAB 8 57
ABSTRACT WINDOWING TOOLKIT (AWT) DAN SWING 57
BAB 9 76
GUI EVENT HANDLING 76
BAB 10 91
JDBC 91

BAB 1

DASAR PEMROGRAMAN JAVA

1.1 Tujuan

Mengidentifikasi bagian dasar dari program Java, membedakan mana yang termasuk ke dalam Java literals, tipe data dasar, tipe variabel, pengidentifikasian dan operator

Mengembangkan program Java sederhana menggunakan konsep pembelajaran pada bab ini

Menganalisa program Java pertama

1.2 Latar Belakang

Pada bab ini, kita akan mendiskusikan mengenai dasar-dasar pemrograman Java. Diantaranya penggunaan variabel, tipe data, dan operator pada Java dan bagaimana mengantisipasi error pada penulisan kode Java.

1.3 Percobaan

1.3..1 Percobaan 1 (hello.Java)

```
class hello {
public static void main (String[] args) {
System.out.println( “Halo Indonesia” ); }
}
```

Hasil :

```
Halo Indonesia
```

Pembahasan :

Setiap program Java, nama file yang berekstensi .java selalu sama dengan nama kelas pada kode program. Setiap blok statemen dibawah kelas selalu dimulai dengan kurung kurawal buka ({) dan pada akhir statemen ditutup dengan kurung kurawal tutup (}), tanpa menggunakan tanda tersebut program akan dinyatakan memiliki kesalahan pada penulisan kode (*syntax error*).

Statemen **public static void main (String[] args)** adalah method utama atau *main method* yang merupakan titik awal dari suatu program java. Sebuah *main method* juga selalu dibuka dengan tanda ({) dan diakhiri dengan (}), sama halnya dengan memulai suatu kelas tanpa tanda tersebut program akan dinyatakan *syntax error*.

Statemen `System.out.println("Halo Indonesia")` merupakan method untuk mencetak string ke layar. Teks didalam method `System.out.println()` selalu diapit oleh tanda kutip dua (" "), dan diakhir statemen selalu diakhiri dengan tanda titik koma (;). Tanpa menggunakan tanda tersebut program akan dinyatakan *syntax error*.

Kata-kata seperti **hello**, **main**, **System**, **out** merupakan Java identifier yaitu suatu tanda yang mewakili nama-nama variabel, method, class, dan sebagainya. Identifier bersifat *case sensitive* yang artinya membedakan huruf kecil dan besar, Identifier harus dimulai dengan salah satu huruf, underscore (_), atau tanda dolar ($) setelah itu dapat diikuti dengan menggunakan angka 0 sampai dengan 9.

Identifier tidak dapat menggunakan kata kunci yang sudah didefinisikan terlebih dahulu oleh Java yang digunakan untuk tujuan tertentu. Berikut adalah daftar kata kunci (*keyword*) dalam Java :

G

abstract	continue	for	new	switch
assert***	default	goto*	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum****	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp**	volatile
const*	float	native	super	while

* not used
** added in 1.2
*** added in 1.4
**** added in 5.0

J

ava keyword (Sumber : Materi JENI 1)

1.3..2 Percobaan 2 (variabel-tipedata.Java)

```
public class variabel {
public static void main (String[] args) {
char var1='Z';
int var2=10;
double var3=0.8;
boolean var4=true;
System.out.println( "Isi variabel var1 :  " +var1);
System.out.println( "Isi variabel var2 :  " +var2);
System.out.println( "Isi variabel var3 :  " +var3);
System.out.println( "Isi variabel var4 :  " +var4);}
}
```

Hasil :

```
Isi variabel var1 : Z
Isi variabel var2 : 10
Isi variabel var3 : 0.8
Isi variabel var4 : true
```

Pembahasan :

Variabel adalah item yang digunakan data untuk menyimpan pernyataan objek. Variabel selalu memiliki **nama** dan **tipe data**. Deklarasi variabel dapat dituliskan :

`<tipe-data> nama-variabel [=initial value]`

Pada percobaan 2 diatas, statemen `int var2=10;`, `int` adalah tipe data, `var2` adalah nama variabel, dan `10` adalah initial value.

Variabel juga dapat dibedakan menjadi dua, yaitu **variabel reference** dan **variabel primitif**. **Variabel primitif** adalah variabel dengan tipe data primitif. Mereka menyimpan data dalam lokasi memori yang sebenarnya dimana variabel tersebut berada. Variabel reference adalah variabel yang menyimpan alamat dalam lokasi memori yang menunjuk ke lokasi memori dimana data sebenarnya

berada. Ketika anda mendeklarasikan variabel pada class tertentu, anda sebenarnya mendeklarasikan variabel reference dalam bentuk objek dalam class tersebut.

Bahasa pemrograman Java mendefinisikan delapan tipe data primitif yaitu char (untuk bentuk tekstual), byte, short, int, long (integral / bilangan bulat), double dan float (floating point / bilangan pecahan). String bukan merupakan tipe data primitif melainkan sebuah kelas.

Tipe data character (char), diwakili oleh karakter single unicode. Tipe data ini harus memiliki ciri berada dalam tanda *single quotes* (' '), contoh : 'x'. Untuk menampilkan karakter khusus seperti ' (single quotes) atau “ (double quotes) menggunakan karakter escape (\), contoh : '\'' atau '\”'.

Tipe data integral dalam Java menggunakan tiga bentuk yaitu desimal , oktal atau heksadesimal. Range tipe data integral :

Tabel 1.1. Tipe-tipe integral dan range-nya

Integer Length	Name or Type	Range
8 bits	byte	-2^7 to 2^7-1
16 bits	short	-2^{15} to $2^{15}-1$
32 bits	int	-2^{31} to $2^{31}-1$
64 bits	long	-2^{63} to $2^{63}-1$

Tipe floating point memiliki double sebagai default tipe datanya. Tipe data floating point memiliki range sebagai berikut :

Tabel 1.2. Tipe-tipe Float dan range-nya

Float Length	Name or Type	Range
32 bits	Float	-2^{31} to $2^{31}-1$
64 bits	Double	-2^{63} to $2^{63}-1$

Tipe boolean diwakili oleh dua pernyataan : **true** dan **false**. Contoh : boolean var1=true. Isi dari variabel var1 adalah pernyataan true.

1.3..3 Percobaan 3 (operatoraritmatik.java)

```
public class operatoraritmatik {
```

```
public static void main (String[] args) {
int a = 10;
int b = 20;
int c;
System.out.println( “Operator Aritmatika :” );
System.out.println( “Penambahan : ” );
c = a+b;
System.out.println( “Isi  c =”  +c);
System.out.println( “Pengurangan : ” );
c = b-a;
System.out.println( “Isi  c  =”  +c);
System.out.println( “Perkalian :  ” );
c = a*b;
System.out.println( “Isi variabel c =”  +c);
System.out.println( “Pembagian : ” );
c = b/a;
System.out.println( “Isi variabel c =”  +c);
System.out.println( “Sisa Hasil Bagi : ” );
c = b%a;
Sytem.out.println( “Isi variabel c =”  +c);
}
}
```

Hasil :

```
Operator Aritmatika :
Penambahan :
Isi variabel c = 30
Pengurangan :
Isi variabel c = 10
Perkalian :
Isi variabel c = 200
Pembagian :
Isi variabel c : 2
Sisa Hasil Bagi :
```

```
Isi variabel c : 0
```

Pembahasan :

Operator ini mengikuti bermacam-macam prioritas yang pasti sehingga compilernya akan tahu yang mana operator untuk dijalankan lebih dulu dalam kasus beberapa operator yang dipakai bersama-sama dalam satu pernyataan.

Operator aritmatik yang dapat digunakan untuk membuat suatu program Java :

Tabel 1.3. Operator aritmatika dan fungsi-fungsinya

Operator	Penggunaan	Keterangan
+	op1+op2	Menambahkan op1 dengan op2
*	op1*op2	Mengalikan op1 dengan op2
/	op1/op2	Membagi op1 dengan op2
%	op1%op2	Menghitung sisa dari pembagian op 1 dengan op2
-	op1-op2	Mengurangkan op2 dari op1

1.3..4 Percobaan 4 (operatorincdec.java)

```
public class operatorincdec {
public static void main (String[] args) {
int a=10;
System.out.println(“Penambahan satu : “);
a++;
System.out.println(“Isi a:” +a);
System.out.println(“Pengurangan satu : “);
a--;
System.out.println(“Isi a:” +a);
}
}
```

Hasil :

```
Penambahan satu :
Isi a : 11
```

```
Pengurangan satu :
Isi a : 9
```

Pembahasan :

Operator increment dan decrement menambah dan mengurangi nilai yang tersimpan dalam bentuk variabel angka terhadap nilai 1.

Pada percobaan 4 diatas, statemen a++ dapat disamakan dengan statemen a=a+1 sedangkan a-- dapat disamakan dengan statemen a=a-1.

Penggunaan operator increment dan decrement sebagai berikut :

Tabel 1.4 Operator Increment dan Decrement

Operator	Penggunaan	Keterangan
++	op++	Menambahkan nilai 1 pada op; mengevaluasi nilai op sebelum diitambahkan
++	++op	Menabambahkan nilai 1 pada op; mengevaluasi nilai op setelah ditambahkan
--	op--	Mengurangkan nilai 1 pada op; mengevaluasi nilai op sebelum dikurangkan
--	--op	Mengurangkan nilai 1 pada op; mengevaluasi nilai op setelah dikurangkan

1.3..5 Percobaan 5 (operatorrelasi.java)

```
public class operatorrelasi {
public static void main (String[] args) {
int x=10;
int y=30;
System.out.println( “x > y adalah  “ + (x>y));
System.out.println( “x >= y adalah  “ + (x>=y));
System.out.println( “x < y adalah  “ + (x<y));
System.out.println( “x <= y adalah  “ + (x<=y));
System.out.println( “x = y adalah  “ + (x==y));
```

```
System.out.println( “x != y adalah  “ + (x!=y));
```

Hasil :

```
x > y adalah false
x >= y adalah false
x < y adalah true
x <= y adalah true
x = y adalah false
x != y adalah true
```

Pembahasan :

Operator relasi membandingkan dua nilai dan menentukan keterhubungan diantara nilai-nilai tersebut. Hasil keluarannya berupa nilai boolean yaitu true atau false.

Tabel 1.5 Operator Relasi

Operator	Penggunaan	Keterangan
>	Op1 > op2	Op1 lebih besar dari op2
>=	Op1 >= op2	Op1 lebih besar dari atau sama dengan op2
<	Op1 < op2	Op1 kurang dari op2
<=	Op1 <= op2	Op1 kurang dari atau sama dengan op2
==	Op1 == op2	Op1 sama dengan op2
!=	Op1 != op2	Op1 tidak sama dengan op2

1.3..6 Percobaan 6 (operatorlogika.java)

```
public class operatorlogika {
public static void main(String[] args){
int y=40;
int z=80;
tes1=(y>40) && (z<100);
System.out.println( “Hasil tes1 :  “ +tes1);
tes2=(y>40) || (z<100);
```

```
System.out.println( “Hasil tes1 :  “ +tes2);
}
}
```

Hasil :

```
Hasil tes1 : false
Hasil tes2 : true
```

Pembahasan :

Opearator logika memiliki satu atau lebih operand boolean yang menghasilkan nilai boolean. Operator logika AND (&&) dan boolean logika & memiliki tabel logika sebagai berikut :

Tabel 1.6 Tabel kebenaran untuk && dan &

X1	X2	Hasil
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

Contoh jika diberikan pernyataan exp1 && exp2, operator logika AND (&&) akan mengevaluasi pernyataan exp1, dan segera mengembalikan nilai false dan menyatakan bahwa exp1 bernilai false. Jika exp1 bernilai false, operator tidak akan pernah mengevaluasi exp2 karena hasil operasi operator akan menjadi false tanpa memperhatikan nilai dari exp2. Sebaliknya, operator logika AND (&) selalu mengevaluasi kedua nilai dari exp1 dan exp2 sebelum mengembalikan suatu nilai jawaban.

Operator logika OR (||) dan boolean logika OR (|), tabel kebenarannya adalah sebagai berikut :

Tabel 1.7 Tabel kebenaran untuk || dan |

X1	X2	Hasil
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

1.4 Latihan

a) Buatlah program dengan output sebagai berikut :

Halo <nama anda> Selamat Datang di Dunia Java !

b) Buatlah program dengan output sebagai berikut :

Harga barang sebelum diskon : Rp.10.000,-

Diskon : 10%

Harga barang setelah diskon : Rp. 9.000,-

BAB 2

INPUT DARI KEYBOARD

2.1 Tujuan

Membuat program java yang interaktif yang bisa membaca input dari keyboard

Menggunakan class BufferedReader untuk mendapatkan input dari keyboard melalui layar console

Menggunakan class JoptionPane untuk mendapatkan input dari keyboard menggunakan GUI

2.2 Latar Belakang

Pada bab ini kita akan mendiskusikan bagaimana Java menangani masukan (*input*) dari pengguna melalui keyboard sehingga program menjadi lebih interaktif.

2.3 Percobaan

2.3..1 percobaan 1 (input.java)

```
import
java.io.BufferedReader; import
java.io.InputStreamReader; import
java.io.IOException;


public class input {
public static void main (String[] args) {
BufferedReader InputData = new BufferedReader (new InputStreamReader
(System.in));
String nama= “ “;
System.out.print (“Masukkan nama anda : “ );
try {
nama = InputData.readLine();
}
catch (IOException e ){
System.out.println( “Error” );
}
System.out.println( “Halo nama saya “ +nama);
}
}
```

Hasil :

```
Masukkan nama anda : Bob
Halo nama saya Bob
```

Pembahasan :

Bahasa pemrograman Java tidak memiliki kelas default yang bisa digunakan dengan cara yang langsung untuk menerima input ari pengguna, tidak seperti bahasa-bahasa pemrograman lainnya. Meskipun demikian, dengan cara-cara tertentu, kita tentu saja dapat membuat program Java yang dapat menerima input dari penguna.

Statemen :

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;
```

Menjelaskan bahwa kita akan menggunakan class `BufferedReader`, `InputStreamReader` dan `IOException` yang berada di `java.io package`. Java Application Programming Interface (API) berisi ratusan class yang sudah didefinisikan sebelumnya yang dapat digunakan untuk program anda. Class-class tersebut dikumpulkan di dalam packages.

Packages berisi class yang mempunyai fungsi yang saling berhubungan. Seperti pada contoh diatas, `java.io` package mengandung class-class yang memungkinkan program untuk melakukan input dan output data. Pernyataan diatas juga dapat ditulis sebagai berikut,

```
import java.io.*;
```

yang akan mengeluarkan semua class yang berada dalam package, dan selanjutnya kita bisa menggunakan class-class tersebut dalam program kita.

Dalam statemen,

```
BufferedReader InputData = new BufferedReader(new InputStreamReader
(System.in));
```

kita mendeklarasikan sebuah variabel bernama InputData dengan tipe class `BufferedReader`.

Sekarang, kita akan mendeklarasikan variabel String dengan identifier nama,

```
String nama = “ “;
```

Pernyataan diatas merupakan tempat untuk menyimpan input dari user. Nama variabel diinisialisasi sebagai String kosong (“ “). Sebaiknya kita selalu menginisialisasi sebuah variabel setelah kita mendeklarasikannya.

Selanjutnya, blok dibawah ini merupakan try-catch block,

```
try {
nama=dataIn.readline();     }
catch (IOException e) {
System.out.println(“Error!”);}
```

Pada baris ini menjelaskan bahwa kemungkinan terjadi error pada pernyataan,

```
nama = dataIn.readline();
```

akan ditangkap. Kita perlu menambahkan kode ini untuk menggunakan method `readLine()` dari `BufferedReader` untuk mendapatkan input dari user dan memberikan sebuah nilai String. Nilai akan disimpan ke dalam variabel nama, yang akan kita gunakan pada statemen akhir untuk menyatakan nama kita.

```
System.out.println(“Halo nama saya “ +nama);
```

2.3..2 percobaan 2 (inputgui.java)

```
import javax.swing.JOptionPane;


public class inputgui
{
public static void main (String[] args) {
String nama= “ “;
nama=JOptionPane.showInputDialog(“Silahkan masukkan nama anda”);
String psn= “Halo saya “ +nama;
JOptionPane.showMessageDialog(null,psn);
}
}
```

Hasil :



Pembahasan :

Statemen pertama,

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

Menjelaskan bahwa kita mengimpor class JOptionPane dari package `javax.swing`.

Bisa juga ditulis seperti,

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

Pernyataan,

```
nama=JOptionPane.showInputDialog("Silahkan masukkan nama anda");
```

membuat sebuah input dialog `JoptionPane`,

Hasil dari dialog tersebut adalah String dan disimpan ke dalam variabel nama.

Sekarang kita membuat pesan, yang akan disimpan dalam variabel `psn`,

```
String psn= "Halo saya " +nama;
```

Pernyataan,

```
JoptionPane.showMessageDialog(null,psn);
```

adalah menampilkan sebuah dialog yang berisi sebuah pesan dan tombol OK.

2.4 Latihan

a) Buatlah Program dengan menggunakan class JoptionPane dari package javax.swing dengan output sebagai berikut :

Masukkan Nama : <Nama>

Masukkan NIM : <NIM>

Masukkan Jurusan : <Jurusan>

Halo <Nama>, NIM anda <NIM>, anda adalah Mahasiswa Jurusan <Jurusan>

BAB 3

STRUKTUR KONTROL

3.1 Tujuan

Menggunakan struktur kontrol pemilihan (if, else, switch) yang digunakan untuk memilih blok kode yang akan dieksekusi

Menggunakan struktur kontrol pengulangan (while, do-while, for) untuk mengeksekusi blok tertentu pada program beberapa kali

Menggunakan pernyataan-pernyataan percabangan (break, continue, return) yang digunakan untuk mengatur arah dari aliran programming

3.2 Latar Belakang

Pada bab ini kita akan mendiskusikan bagaimana membuat program yang menggunakan struktur kontrol pemilihan dan pengulangan serta menggunakan pernyataan-pernyataan percabangan.

3.3 Percobaan

3.3.1 Percobaan 1 (seleksi_if.java)

```
public class seleksi_if {
public static void main (String[] args) {
int jmlhmhs = 105;
int dayatampungkelas=100;
if (jmlhmhs>100) {
System.out.println ( “Kelas melebihi kapasitas” );
}
}
```

Hasil :

```
Kelas melebihi kapasitas
```

Pembahasan :

Struktur kontrol pemilihan menggunakan `if` adalah pernyataan kondisi yang digunakan untuk pengambilan keputusan terhadap dua buah kemungkinan. Statemen `if` bisa berdiri sendiri atau dengan menggunakan `else`.

Bentuk pernyataan `if` :

```
if (kondisi) {
```

```
// blok pernyataan yang dijalankan, bila kondisi benar }
```

Perbedaan kondisional `if` pada Java dibandingkan dengan bahasa pemrograman lain yaitu kondisional pada Java hanya menghasilkan nilai boolean (`true` atau `false`). Pada C dan C++, nilai balik bisa berupa integer.

3.3.2 Percobaan 2 (seleksi_if_else.java)

```
public class seleksi_if_else {
public static void main (String[] args) {
int totalbelanja = 100000;
int jmlh_brg_dibeli = 2;
double diskon = 0.25;
double hrg_brg_stlh_diskon;
if (totalbelanja>50000 && jmlh_brg_dibeli>1)
{hrg_brg_stlh_diskon = totalbelanja-(totalbelanja*diskon);
System.out.println ("Harga barang setelah diskon :"+hrg_brg_stlh_diskon);
}
else { System.out.println( "Maaf anda belum dapat diskon"); }
}
}
```

Hasil :

```
Harga barang setelah diskon :75000.0
```

Pembahasan :

Pernyataan `if-else` mengatur pernyataan yang dijalankan sewaktu kondisi bernilai `true` atau `false`.

Bentuk pernyataan :

```
if (kondisi) {
// blok pernyataan yang dijalankan, bila kondisi benar }
else {
//blok pernyataan yang dijalankan, bila kondisi salah}
```

3.3.3 Percobaan 3 (seleksi_if_bersarang.java)

```
public class seleksi_if_bersarang {
```

```
public static void main (String[] args) {
int nilai_angka = 75;
if (nilai_angka>=85 && nilai_angka <=100) {
System.out.println ("Nilai Huruf A"); }
else if (nilai_angka >=70 && nilai_angka <85) {
System.out.println ("Nilai Huruf B"); }
else if (nilai_angka >=55 && nilai_angka <70) {
System.out.println ("Nilai Huruf C"); }
else if (nilai_angka ==50 && nilai_angka <55) {
System.out.println ("Nilai Huruf D"); }
else {
System.out.println("Nilai Huruf E"); }
}
}
```

Hasil :

```
Nilai Huruf B
```

Pembahasan :

Pernyataan if-else if mengatur pernyataan yang dijalankan sewaktu kondisi berupa pilihan

Bentuk pernyataannya :

```
if (kondisi x) {
// pernyataan yang dijalankan, apabila kondisi x benar }
else if (kondisi y) {
// pernyataan yang dijalankan, apabila kondisi y benar }
else if (kondisi z) {
// pernyataan yang dijalankan, apabila kondisi z benar }
else {
// pernyataan yang dijalankan untuk kondisi selain itu }
```

3.3.4 Percobaan 4 (seleksi_switch.java)

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
```

```
import java.io.IOException;
public class seleksi_switch {
public static void main (String[] args){
BufferedReader InputData = new BufferedReader (new InputStreamReader
(System.in));
String angkaInput= null;
System.out.print ("Masukkan kode hari [1...7] : " );
try {
angkaInput = InputData.readLine();
}
catch (IOException e ){
System.out.println("Error");
}
int Hari=Integer.valueOf(angkaInput).intValue();
switch (Hari){
case 1:System.out.println("Minggu"); break;
case 2:System.out.println("Senin"); break;
case 3:System.out.println("Selasa"); break;
case 4:System.out.println("Rabu"); break;
case 5:System.out.println("Kamis"); break;
case 6:System.out.println("Jum'at"); break;
case 7:System.out.println("Sabtu"); break;
default :System.out.println("Kode hari yang anda masukkan salah"); }
}
}
```

Hasil :

```
Masukkan kode hari [1...7] : 4
Rabu
```

Pembahasan :

Pernyataan switch mengijinkan beberapa pilihan jalur eksekusi. Pernyataan Switch bekerja pada tipe data byte, short, char dan int yang merupakan tipe data primitif.

Bentuk pernyataan switch :

```
switch(variabel) {
case [data] : <pernyataan> // dieksekusi apabila seleksi pada variabel sesuai dengan data (bernilai true)
default : <pernyataan> // dieksekusi apabila seleksi pada variabel sesuai dengan data (bernilai false)
}
```

Tidak seperti pada pernyataan if, beberapa pernyataan pada struktur pernyataan switch akan dieksekusi tanpa memerlukan tanda kurung kurawal ({})

Pada program diatas pernyataan switch akan menganalisa hasil inputan dari keyboard yang ditampung dalam variabel angkaInput yang awalnya adalah sebuah String. Pada pernyataan :

```
int Hari=Integer.valueOf(angkaInput).intValue();
```

variabel angkaInput yang tadinya berupa String dirubah kedalam Integer yang ditampung dalam variabel Hari. Pernyataan switch nantinya akan menyeleksi isi dari variabel Hari tersebut.

3.3.5 Percobaan 5 (loop_for.java)

```
public class loop_for {
public static void main (String[] args){
int i;
for(i=1;i<5;i++)
{
  if(i%2==0)
        {System.out.println(i+" adalah Genap");}
     else
        {System.out.println(i+" adalah Ganjil");}
}
}
}
```

Hasil :

```
1 adalah Ganjil
2 adalah Genap
```

```
3 adalah Ganjil
4 adalah Genap
```

Pembahasan :

For sering disebut dengan for loop, karena digunakan untuk proses looping atau pengulangan.

Bentuk pernyataan :

```
for (inisialisasi;kondisi;step_ekspresi) {
pernyataan; }
```

dimana,

`inisialisasi` yaitu inisialisasi dari variabel loop

`kondisi` yaitu membandingkan variabel loop pada nilai batas tertentu

`step_ekspresi` yaitu melakukan update terhadap variabel loop, biasanya berupa penambahan atau pengurangan

Sebagai contoh pada pernyataan :

```
for(i=1;i<5;i++)
```

`i=1` merupakan inisialisasi dimana variabel `i` merupakan variabel loop, sedangkan `i<5` merupakan kondisi dimana jika nilai variabel `i` mencapai batas `(i<5)` maka loop akan berhenti. Dan pernyataan `i++` merupakan step_ekspresi dimana variabel `i` mengalami penambahan 1 atau dapat juga dinyatakan dengan `i=i+1`

3.3.6 Percobaan 6 (loop_while.java)

```
public class loop_while {
public static void main(String[] args){
int x=10; while (x>0) {
System.out.println("Nilai x :" +x);
x--;    }
}
}
```

Hasil :

```
Nilai x :10
```

```
Nilai x :9
Nilai x :8
Nilai x :7
Nilai x :6
Nilai x :5
Nilai x :4
Nilai x :3
Nilai x :2
Nilai x :1
```

Pembahasan :

Pernyataan while digunakan untuk melakukan proses pengulangan suatu blok pernyataan selama kondisinya bernilai true. Ketika kondisi salah, pernyataan dalam blok bisa saja tidak dijalankan sama sekali.

Bentuk pernyataan:

```
while (kondisi) {
pernyataan; }
```

3.3.7 Percobaan 7 (loop_do_while.java)

```
public class loop_do_while {
public static void main(String[] args){
    int i=1;
    do {
       System.out.println("Nilai i :" +i);
       i++;
    }
    while (i<10);
}
}
```

Hasil :

```
Nilai i :1
```

```
Nilai i :2
Nilai i :3
Nilai i :4
Nilai i :5
Nilai i :6
Nilai i :7
Nilai i :8
Nilai i :9
```

Pembahasan :

Do-while loop mirip dengan while-loop. Pernyataan di dalam do-while loop akan diekseskusi beberapa kali selama kondisi bernilai benar (true).

Perbedaan antara while dan do-while loop adalah dimana pernyataan di dalam do-while loop akan dieksekusi sedikitnya satu kali.

Bentuk pernyataan do-while,

```
do {
statemen1;
statemen2;
}
while (boolean-expression);
```

3.3.8 Percobaan 8 (pernyataanBreak.java)

```
public class pernyataanBreak {
public static void main (String[] args){
    for(int i=1;i<5;i++){
        if(i%2==0) break;
            System.out.println("Nilai i: " +i);
    }
}
}
```

Hasil:

```
Nilai i: 1
```

Pembahasan :

Pernyataan break digunakan untuk menghentikan loop, pada pernyataan `for(int i=1;i<5;i++)` , sejatinya loop akan berhenti apabila `i` bernilai 5 karena menghasilkan nilai `boolean (false)`, namun karena pada pernyataan `if(i%2==0) break;` menghendaki jika `i` bernilai genap maka loop berhenti.

3.3.9 Percobaan 9 (pernyataanContinue.java)

```
public class pernyataanContinue {
public static void main(String[] args){
    int i=1;
    while (i<11){
        if(i==7);{
        i++;
        continue;       }
    }
    System.out.println("Nilai i "+ i);
    i++;  }
}
```

Hasil :

```
Nilai i : 11
```

Pembahasan :

pernyataan continue digunakan untuk melanjutkan eksekusi ke suatu pengulangan (loop). Sama halnya dengan pernyataan break, penggunaan continue bisa berbentuk tanpa label atau berlabel.

3.3.10 Percobaan 10 (pernyataanReturn.java)

```
public class pernyataanReturn {
   void perkalian(int i,int j){
       int hasilkali=i*j;
       System.out.println("Hasil perkalian :" +hasilkali);
       return;
    }
```

```
    public static void main(String[] args){
    pernyataanReturn a = new pernyataanReturn();
    a.perkalian(5, 6);
    }
}
```

Hasil :

```
Hasil perkalian :30
```

Pembahasan :

Pernyataan `return` digunakan untuk keluar dari sebuah method. Pernyataan `return` memiliki dua bentuk : memberikan sebuah nilai, dan tidak memberikan sebuah nilai.

Untuk memberikan nilai, cukup berikan nilai (atau ekspresi yang menghasilkan sebuah nilai) sesudah kata `return`. Contohnya,

```
return ++count;
```

atau

```
return “Hello”;
```

Tipe data dari nilai yang diberikan harus sama dengan tipe dari method yang dibuat. Ketika sebuah method void dideklarasikan, gunakan bentuk `return` yang tidak memberikan nilai. Contohnya ,

```
return;
```

3.4 Latihan

a) Buatlah program dengan menggunakan pernyataan seleksi yang menentukan hari kerja atau hari libur, dimana hari kerja yaitu hari senin sampai dengan jum'at. Sedangkan hari libur adalah sabtu dan minggu.

b) Buatlah program dengan output sebagai berikut :

*

**

***

c) Buatlah program menggunakan seleksi dipadukan dengan input keyboard dimana outputnya adalah sebagai berikut :

Masukkan kode buku : T-INF-001-WAH

Masukkan Tanggal-Pinjam : 10 September 2010

Masukkan Tanggal-Kembali : 19 Oktober 2010

Denda : Rp. 3.000,-

Ketentuan peminjaman buku : Maksimal 3 Hari, denda per hari Rp.1.000,-

BAB 4

ARRAY

4.1 Tujuan

Mendeklarasikan dan membuat array

Mengakses elemen-elemen didalam array

Menentukan jumlah elemen didalam sebuah array

Mendeklarasikan dan membuat array multidimensi

4.2 Latar Belakang

Pada bab ini kita akan mendiskusikan mengenai array dalam Java. Pertama, kita akan mendefinisikan apa yang dimaksud dengan array, kemudian kita juga akan mendiskusikan bagaimana mendeklarasikannya dan menggunakannya dalam Java.

4.3 Percobaan

4.3.1 Percobaan1

```
public class array1 {
    public static void main(String[] args){
        int myArray[]={10,20,30,40,50};
System.out.println("Isi elemen MyArray pertama : " +myArray[0]);
System.out.println("Isi elemen MyArray kedua   : " +myArray[1]);
System.out.println("Isi elemen MyArray pertama : " +myArray[2]);
System.out.println("Isi elemen MyArray pertama : " +myArray[3]);
System.out.println("Isi elemen MyArray pertama : " +myArray[4]);
    }
}
```

Hasil :

```
Isi elemen MyArray pertama : 10
Isi elemen MyArray kedua   : 20
Isi elemen MyArray pertama : 30
Isi elemen MyArray pertama : 40
Isi elemen MyArray pertama : 50
```

Pembahasan :

Sebuah array akan menyimpan beberapa item data yang memiliki tipe data sama dalam didalam sebuah blok memori yang berdekatan yang kemudian dibagi menjadi beberapa ruang. Array adalah sebuah variabel / sebuah lokasi tertentu yang memiliki satu nama sebagai identifier, namun identifier ini dapat menyimpan lebih dari sebuah nilai.

Langkah-langkah menciptakan array yaitu pertama, mendeklarasikan variabel array, kedua menciptakan objek array.

Bentuk deklarasi variabel array :

```
tipedataprimitif[] namaVariabel;
namakelas[] namaVariabel;
```

Atau,

```
tipedataprimitif namaVariabel[];
namakelas namaVariabel[];
```

Contoh :

```
String[] kata;
int[] nomor;
```

```
Atau,
String kata[];
int nomor[];
```

Untuk menciptakan objek array, bentuk deklarasinya adalah sebagai berikut :

```
namaVariabel = new tipedataprimitif[jumlahelemen];
namaVariabel = new namakelas[jumlahelemen];
```

Contoh :

```
nomor = new int[10];
kota = new String[5];
```

Atau kita dapat mempersingkat deklarasi variabel dan objek array sebagai berikut

:

```
String[] kota = new String[8];
int [] nomor = new int[7];
```

Selanjutnya, kita dapat mengakses elemen array dengan deklarasi sebagai berikut :

```
namaVariabelArray[nomorelemen];
```

Contoh :

```
negara[0] = “Indonesia” ;
```

Atau kita dapat secara langsung memberi nilai ketika objek array diciptakan. Seperti pada percobaan1 diatas :

```
int myArray[]={10,20,30,40,50};
```

4.3.2 Percobaan2

```
public class array2 {
public static void main(String[] args){
    int myArray[]={30,50,70,90,110};
    for (int i=0;i<myArray.length;i++)
        System.out.println("Elemen ke-"+i +":"+myArray[i] );
    }
}
```

Hasil :

```
Elemen ke-0:30
Elemen ke-1:50
Elemen ke-2:70
Elemen ke-3:90
Elemen ke-4:110
```

Pembahasan:

Kita dapat memodifikasi kode program untuk dapat mengakses elemen array dengan menggunakan loop, seperti pada contoh diatas kita mendefinisikan

variabel i untuk dapat mengakses setiap elemen secara lebih ringkas.

Statemen :

```
for (int i=0;i<myArray.length;i++)
```

Maksudnya apabila `i` nilainya lebih kecil dari panjang array `(myArray.length)`, maka tambahkan satu nilai `i (i=i+1)` sampai nilai `i` sama dengan panjang array sehingga keluar dari loop.

4.3.3 Percobaan3

```
public class array3 {
public static void main(String[] args){
String myArray[][] = {{"Negara ","Ibukota "},{"Indonesia","Jakarta"}};
System.out.println(myArray[0][0]+myArray[1][0]);
System.out.println(myArray[0][1]+myArray[1][1]);
}
}
```

Hasil :

```
Negara Indonesia
Ibukota Jakarta
```

Pembahasan :

Array multidimensi adalah array dari array, dengan pengaksesan [noBaris][noKolom]. Jadi jika dibuat tabel, gambarannya sebagai berikut :

Negara [Baris0][Kolom0]	Ibukota [Baris0][Kolom1]
Indonesia[Baris1][Kolom0]	Jakarta [Baris1][Kolom1]

4.4 Latihan

a) Buatlah program dengan output sebagai berikut menggunakan array multidimensi :

Budi adalah anak dari Bapak Yanto

Joko adalah anak dari Bapak Yudi

Ibu Rina adalah Istri dari Bapak Yanto

Ibu Lina adalah Istri dari Bapak Yudi

BAB 5

KELAS DAN OBJEK

5.1 Tujuan

Membuat *class* sendiri

Mendeklarasikan atribut dan method pada *class*

Membuat dan memanggil *overloading method*

Mengimport dan membuat *constructor*

Menggunakan *access modifier* untuk mengendalikan akses terhadap *class* member

5.2 Latar Belakang

Pada bab ini kita akan mempelajari bagaimana menuliskan sebuah class sendiri.

5.3 Percobaan

5.3.1 Percobaan1 (Manusia.java)

```
public class Manusia {
public void identitas(String nama, String tempat_lahir, String tgl_lahir,int
berat_badan, int tinggi_badan){
System.out.println("Nama saya : " +nama);
System.out.println("Tempat lahir : " +tempat_lahir);
System.out.println("Tanggal lahir : " +tgl_lahir);
System.out.println("Berat Badan : " +berat_badan +" kg");
System.out.println("Tanggal lahir : " +tinggi_badan +" cm"); }
public static void main(String[] args){
Manusia Orang_Indonesia = new Manusia();
Orang_Indonesia.identitas("Budi", "Yogyakarta", "01-01-1991", 65, 175);
}}
```

Hasil :

```
Nama saya : Budi
Tempat lahir : Yogyakarta
Tanggal lahir : 01-01-1991
Berat Badan : 65 kg
Tanggal lahir : 175 cm
```

Pembahasan :

Untuk membuat sebuah objek atau sebuah instance pada sebuah class. Kita menggunakan operator **new.** Sebagai contoh, jika anda ingin membuat instance dari class String, kita menggunakan kode berikut :

```
String str2 = new String ( “Hello World” );
```

ini juga sama dengan,

```
String str2 =  “Hello” ;
```

Pada contoh diatas, sebuah Class Manusia dibuat kemudian dibuatlah metode/fungsi `identitas` yang berisi parameter `nama, tempat_lahir, tgl_lahir, berat_badan, tinggi_badan.` metode/fungsi adalah bagian-bagian kode yang dapat dipanggil oleh program utama atau dari metode/fungsi lainnya untuk menjalankan fungsi yang spesifik.

5.3.2 Percobaan2 (Mobil.java)

```
public class Mobil {
String warna;
int jumlah_pintu;
float isi_tangki;

public void Maju(int maju){
int majukedepan=0;
majukedepan=majukedepan+maju;
System.out.println("Mobil maju sejauh " +maju +" meter");
    }
public void Mundur(int mundur){ int
mundurbelakang=0;
mundurbelakang=mundurbelakang+mundur;
System.out.println("Mobil mundur sejauh " +mundur +" meter");
}
public static void main(String[] args){
Mobil BMW = new Mobil();
BMW.Maju(75);
BMW.Mundur(5);
}}
```

Hasil :

```
Mobil maju sejauh 75 meter
Mobil mundur sejauh 5 meter
```

Pembahasan :

Perhatikan *syntax* program diatas bahwa baik metode/fungsi `Maju()` serta `Mundur()` kita deklarasikan sebagai public. Apa artinya ? Artinya :

`public`. Berarti kelak kita dapat memanfaatkan metode Maju() dengan memanggilnya dari kelas yang lain

`Maju(int maju)`. Berarti metode/fungsi `Maju()` dapat menerima nilai-nilai tertentu dari metode/fungsi lain yang memanggilnya. (Misalnya saat metode/fungsi `Maju()` ini dipanggil dari metode/fungsi `main()`, metode/fungsi `Maju()` akan menerima bilangan bulat [integer] dari pengguna program).

Selanjutnya, kita akan membahas logika pemrograman untuk perintah-perintah yang ada di dalam metode/fungsi `Maju()`. Sebelumnya kita akan menulis ulang perintah-perintah yang ada dalam metode/fungsi `Maju()` sebagai berikut :

```
int majukedepan = 0;
majukedepan=majukedepan+maju;
```

Peubah (variabel) majukedepan adalah peubah yang ada dalam metode/fungsi `Maju()` yang kelak digunakan untuk menampung nilai peubah maju yang berasal dari metode/fungsi pemanggilnya serta digunakan untuk mengembalikan nilai ke metode pemanggilnya. Kita memberi nilai awal (melakukan inisialisasi) peubah `majukedepan` ini sebagai nol(0) sehingga kelak akan mencerminkan nilai peubah `maju` yang berasal dari metode pemanggilnya. Kemudian, persamaan sederhana :

```
majukedepan = majukedepan+maju;
```

digunakan untuk memasukkan nilai peubah `maju` ke peubah `majukedepan`

Pada metode/fungsi utama (*Main program*), dibuatlah sebuah objek `BMW` dari kelas `Mobil` menggunakan kata kunci `new` yang nantinya akan memanggil metode/fungsi `Maju()` maupun `Mundur()` pada baris :

```
BMW.Maju(75);
BMW.Mundur(5);
```

Syntax untuk pemanggilan metode/fungsi itu secara umum adalah :
`Nama_Kelas/Nama_Objek.Nama_metode(parameter)`

parameter yang kita berikan saat pemanggilan metode `Maju()` maupun `Mundur()` adalah bilangan bulat (integer).

5.3.3 Percobaan3 (penduduk.java)

```
public class penduduk {
    private String nama,ttl,alamat,nama_ortu,pekerjaan;
    private int penghasilan;
public void setNama(String nama){
}
public void setTTL(String ttl){
}
public void setAlamat(String alamat){
}
public void setOrtu(String nama_ortu){
}
public void setPekerjaan(String pekerjaan){
}
public void setPenghasilan(int penghasilan){
}
public  void  printIdentitas(String  nama,String  ttl,  String  alamat,
String nama_ortu, String pekerjaan){
    System.out.println("Nama Penduduk : " +nama);
    System.out.println("Tempat/Tanggal Lahir : "+ttl);
    System.out.println("Alamat Rumah : "+alamat);
    System.out.println("Nama Orang Tua : " +nama_ortu);
    System.out.println("Pekerjaan : " +pekerjaan);
}
public void printIdentitas(int penghasilan){
```

```
    System.out.println("Penghasilan bersih perbulan Rp."+penghasilan);
}



public static void main (String[] args){
    penduduk pk_kota=new penduduk();
    pk_kota.printIdentitas("Ruby","Gorontalo/01                    Januari
1990","Jl.Cempaka No.5 Gorontalo","Wiryana","Wiraswasta");
    pk_kota.printIdentitas(5000000);
}
}
```

Hasil :

```
Nama Penduduk : Ruby
Tempat/Tanggal Lahir : Gorontalo/01 Januari 1990
Alamat Rumah : Jl.Cempaka No.5 Gorontalo
Nama Orang Tua : Wiryana
Pekerjaan : Wiraswasta
Penghasilan bersih perbulan Rp.5000000
```

Pembahasan :

Dalam class yang kita buat, kadangkala kita menginginkan untuk membuat *method* dengan nama yang sama namun mempunyai fungsi yang berbeda menurut parameter yang digunakan. Kemampuan ini dimungkinkan dalam pemrograman java, dan dikenal sebagai *overloading method.*

Overloading method mengijinkan sebuah metode/fungsi dengan nama yang sama namun memiliki parameter yang berbeda sehingga mempunyai implementasi dan *return value* yang berbeda pula. Daripada memberikan nama yang berbeda pada setiap pembuatan metode/fungsi, *overloading method* dapat digunakan pada operasi yang sama namun berbeda dalam implementasinya.

Overloading method memiliki *properties* sebagai berikut :

Nama yang sama

Parameter yang berbeda

Nilai kembalian (*return*) bisa sama ataupun berbeda

5.3.4 Percobaan4 (konstruktor.java)

```
public class Aktor {
String nama;
int umur;
Aktor(String n,int u){
    nama=n;
    umur=u;

}
void tampilAktor(){ System.out.println("Namaku :
    "+nama); System.out.println("Umurku : "+umur+"
    tahun");
}

public static void main(String args[]){
    Aktor a;
    System.out.println("");
    a=new Aktor("Dedy Mizwar",50);
    a.tampilAktor();
    System.out.println("============");
    System.out.println("");
    a=new Aktor("Rano Karno",45);
    a.tampilAktor();
    System.out.println("============");
}
}
```

Hasil :

```
Namaku : Dedy Mizwar
Umurku : 50 tahun
============
Namaku : Rano Karno
Umurku : 45 tahun
============
```

Pembahasan :

Constructor sangatlah penting pada pembentukan sebuah objek. Constructor adalah metode dimana seluruh inisialisasi objek ditempatkan.

Berikut ini adalah properties dari constructor :

Constructor memiliki nama yang sama dengan class

sebuah constructor mirip dengan metode pada umumnya, namun informasi-informasi berikut yang dapat ditempatkan pada *header* sebuah constructor, scope atau identifikasi pengaksesan (misal : public), nama dari konstruktor dan parameter

constructor tidak memiliki nilai balik (*return value*)

constructor tidak dapat dipanggil secara langsung, namun harus dipanggil dengan menggunakan operator new pada pembentukan sebuah class.

5.4 Latihan

5.4.1 Modifikasi program Mobil.java dengan menambahkan beberapa metode/fungsi pada kelas Mobil dan buatlah dua objek lagi dari kelas Mobil yang memanggil metode/fungsi baru yang sudah anda buat !

5.4.2 Buatlah program yang menerapkan konsep kelas dan objek dengan *output* sebagai berikut :

```
Nama Mahasiswa          : Joni
NIM                     : 5314110007
Semester                : V
IP Semester ini         : 3,50
Anda berhak mengontrak 24 SKS pada Semester VI
```

5.4.3 Buatlah program yang menerapkan konsep *overloading method* dan *constructor* pada sistem perpustakaan. Kembangkan sesuai kemampuan anda !

BAB 6

PEWARISAN, POLIMORFISME, INTERFACE

6.1 Tujuan

Mendefinisikan *superclass* dan *subclasses*

Override method dari *superclasses*

Membuat *method final* dan *class final*

6.2 Latar Belakang

Pada bab ini kita akan membicarakan bagaimana suatu *class* dapat mewariskan sifat dari *class* yang sudah ada. Kita juga akan membicarakan sifat khusus dari Java dimana kita dapat secara otomatis memakai *method* yang tepat untuk setiap objek tanpa memperhatikan asal dari *subclass object*. Pada akhirnya, kita akan mendiskusikan tentang *interface*

6.3 Percobaan

6.3..1 Percobaan 1

```
public class Person {

protected String name;

protected String address;

/**

* Default constructor

*/

public Person(){

System.out.println("Inside Person:Constructor");

name = "";

address = "";

}

/**
```

```
* Constructor dengan dua parameter
*/
public Person( String name, String address) {
this.name = name;
this.address = address;
}
/**
* Method accessor
*/
public String getName() {
return name;
}
public String getAddress() {
return address;
}
public void setName(String name) {
this.name = name;
}
public void setAddress(String add) {
this.address = add;
}
```

```
}

public class Student extends Person{

public Student()

{

//super( "SomeName", "SomeAddress");

//super();

//super.name = "name";

System.out.println("Inside Student:Constructor");

}

public static void main( String[] args) {

Student anna = new Student();

}

}
```

Hasil :

```
Inside Person:Constructor
Inside Student:Constructor
```

Pembahasan :

Perhatikan bahwa atribut name dan address dideklarasikan sebagai protected. Alasannya kita melakukan ini yaitu, kita inginkan atribut-atribut ini untuk bisa diakses oleh subclasses dari superclassess. Jika kita mendeklarasikannya sebagai private, subclasses tidak dapat menggunakannya. Catatan bahwa semua properti dari superclass yang dideklarasikan sebagai public, protected dan default dapat diakses oleh subclasses-nya.

Sekarang, kita ingin membuat class lain bernama Student. Karena Student juga sebagai Person, kita putuskan hanya meng-extend class Person, sehingga kita dapat mewariskan semua properti dan method dari setiap class Person yang ada. Untuk melakukan ini, kita tulis,

```
public class Student extends Person

{

public Student(){

System.out.println( “Inside Student:Constructor” );

//beberapa kode di sini

}

// beberapa kode di sini

}
```

Ketika object Student di-instantiate, default constructor dari superclass secara mutlak meminta untuk melakukan inisialisasi yang seharusnya. Setelah itu, pernyataan di dalam subclass dieksekusi. Untuk mengilustrasikannya, perhatikan kode berikut,

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

Student anna = new Student();

}
```

Dalam kode ini, kita membuat sebuah object dari class Student.

6.3..2 Percobaan 2

```
public class Person {

protected String name;

protected String address;
```

```
/**
* Default constructor
*/
public Person(){
System.out.println("Inside Person:Constructor");
name = "";
address = "";
}
/**
* Constructor dengan dua parameter
*/
public Person( String name, String address) {
this.name = name;
this.address = address;
}
/**
* Method accessor
*/
public String getName() {
System.out.println("Person Name : " +name);
return name;
```

```
}
public String getAddress() {
return address;
}
public void setName(String name) {
this.name = name;
}
public void setAddress(String add) {
this.address = add;
}
}
public class Student extends Person{
public Student()
{
//super( "SomeName", "SomeAddress");
//super();
//super.name = "name";
System.out.println("Inside Student:Constructor");
}
public String getName() {
System.out.println("Student Name : " +name);
return name;
```

```
}
public static void main( String[] args) {
Student anna = new Student();
}
}
public class Employee extends Person{
public String getName() {
System.out.println("Employee Name:" +name);
return name;
}
public static void main(String[] args)
{
Person ref;
Student studentObject = new Student();
Employee employeeObject = new Employee();
ref = studentObject; //Person menunjuk kepada object Student
String temp = ref.getName(); //getName dari Student class dipanggil
System.out.println(temp);
ref = employeeObject; //Person menunjuk kepada object Employee
temp = ref.getName(); //getName dari Employee class dipanggil
System.out.println(temp);
```

```
}

}
```

Hasil :

```
Inside Person:Constructor
Inside Student:Constructor

Inside Person:Constructor
Inside Student:Constructor
Inside Person:Constructor
Student Name :

Employee Name:
```

Pembahasan :

Dalam Java, kita dapat membuat referensi yang merupakan tipe dari superclass ke sebuahobject dari subclass tersebut. Sebagai contohnya,

```
public static main( String[] args )
{ Person
ref;
Student
Employee

studentObject = new Student();
employeeObject = new Employee();
```

```
ref = studentObject; //Person menunjuk kepada

// object Student

//beberapa kode di sini

}
```

Sekarang dimisalkan kita punya method getName dalam superclass Person kita, dan kita override method ini dalam kedua subclasses Student dan Employee,

```
public class Person

{

public String getName(){

System.out.println( “Person Name:” + name);

return name;


}

}

public class Student extends Person

{

public String getName(){

System.out.println( “Student Name:” + name);

return name;


}

}

public class Employee extends Person

{
```

```
public String getName(){

System.out.println( “Employee Name:”  + name);

return name;

}
```

Kembali ke method utama kita, ketika kita mencoba memanggil method getName dari reference Person ref, method getName dari object Student akan dipanggil. Sekarang, jika kita berikan ref ke object Employee, method getName dari Employee akan dipanggil.

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

{ Person

ref;

Student

Employee


studentObject = new Student();

employeeObject = new Employee();

ref = studentObject; //Person menunjuk kepada

// object Student

String temp = ref.getName(); //getName dari Student

//class dipanggil

System.out.println( temp );

ref = employeeObject; //Person menunjuk kepada

// object Employee
```

```
String temp = ref.getName(); //getName dari Employee

//class dipanggil

System.out.println( temp );

}
```

Kemampuan dari reference untuk mengubah sifat menurut object apa yang dijadikan acuan dinamakan polimorfisme. Polimorfisme menyediakan multiobject dari subclasses yang berbeda untuk diperlakukan sebagai object dari superclass tunggal, secara otomatis menunjuk method yang tepat untuk menggunakannya ke particular object berdasar subclass yang termasuk di dalamnya.

Contoh lain yang menunjukkan properti polimorfisme adalah ketika kita mencoba melalui reference ke method. Misalkan kita punya method static printInformation yang mengakibatkan object Person sebagai reference, kita dapat me-reference dari tipe Employee dan tipe Student ke method ini selama itu masih subclass dari class Person.

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

{

Student


studentObject = new Student();

Employee


employeeObject = new Employee();

printInformation( studentObject );

printInformation( employeeObject );


}

public static printInformation( Person p ){

. . . .

}
```

6.4 Latihan

6.4..1 Kembangkan class **matematika** dan **matematikaBeraksi**, lakukan overloading pada method yang ada (pertambahan, pengurangan, perkalian, pembagian). Method baru adalah bertipe data double (pecahan) dan memiliki 3 parameter

Uji di kelsa matematikaBeraksi dengan parameter pecahan : 12,5 28,7, 14,2

Misalnya : pertambahan(12,5, 28,7, 14,2)

BAB 7

EXCEPTION HANDLING

Tujuan

Menangani exception dengan menggunakan try, catch dan finally

Membedakan penggunaan antara throw dengan throws

Menggunakan exception class yang berbeda-beda

Membedakan antara checked exceptions dan unchecked exceptions

Membuat exceptions class sendiri

Latar Belakang

Pada bab ini akan dipelajari bagaimana penanganan akan kesalahan pada program yang dikembangkan. Bahasa pemrograman Java dapat menangani kesalahan-kesalahan yang secara potensial dapat dipulihkan melalui suatu objek yang bernama exception, yaitu suatu objek khusus yang dapat secara virtual menangani semua kesalahan dalam program yang ditulis menggunakan Java.

Percobaan

Percobaan1

```
public class TesException {
    public static void main(String[] args){
        int array1[]=new int[10];
        try {
    array1[1]=10;
    System.out.println("Pernyataan yang benar");
    array1[100]=20;
    System.out.println("Pernyataan yang tidak benar");
        }
        catch (ArrayIndexOutOfBoundsException e) {
   System.out.println("Kesalahan dalam indeks array");
        }
    } }
```

Hasil :

```
Pernyataan yang benar
Kesalahan dalam indeks array
```

Pembahasan :

Pada contoh program diatas, larik `array1` dibuat dengan panjang 10. Kemudian, pernyataan deklarasi larik diikuti oleh pernyataan `try` yang memuat beberapa pernyataan yang akan dilakukan oleh bahasa Java saat tidak terjadi masalah. Pernyataan `array1[1]=10;` dan `System.out.println("Pernyataan yang benar");` akan dieksekusi secara normal. Namun pada pernyataan `array1[100]=20;` akan menghasilkan `exception` karena ukuran `array1` yang hanya 10 dipaksakan diisi oleh elemen ke-100 sehingga `try` akan langsung melempar pada `catch` sehingga pernyataan `System.out.println("Pernyataan yang tidak benar");` tidak akan pernah dieksekusi melainkan pernyataan yang ada didalam blok `catch` yaitu `System.out.println("Kesalahan dalam indeks array");`

Percobaan2

```
public class TesFinally {
public static void main(String[] args){ int
    bilangan=100;
    System.out.println("Sebelum pembagian");
    for(int i=5;i>=0;i--){
        try {
          System.out.print(bilangan +" / " +i +" = ");
            System.out.println(bilangan/i);
        }
        catch(RuntimeException r){
```

```
   System.out.println("Error karena pembagian nol !");
        }
        finally {
      System.out.println("Bagian finally dikerjakan");
        }
    }
    System.out.println("Selesai"); }
}
```

Hasil :

```
Sebelum pembagian
100 / 5 = 20
Bagian finally dikerjakan
100 / 4 = 25
Bagian finally dikerjakan
100 / 3 = 33
Bagian finally dikerjakan
100 / 2 = 50
Bagian finally dikerjakan
100 / 1 = 100
Bagian finally dikerjakan
100 / 0 = Error karena pembagian nol !
Bagian finally dikerjakan
Selesai
```

Pembahasan :

Pada program diatas terdapat perulangan dimana bilangan yang nilainya adalah 100 akan dibagi dengan nilai i dan pengurangannya satu persatu sehingga dapat dilihat bahwa blok pernyataan dibawah try yaitu `System.out.print(bilangan`

+" / " +i +" = "); dan System.out.println(bilangan/i); akan selalu dieksekusi hingga nilai i bernilai 0 maka akan menghasilkan RuntimeException yang akan tampil dimonitor , apabila tidak menangkap kesalahan tersebut dengan pernyataan catch(RuntimeException r). Sehingga ketika bilangan 100/0 yang menghasilkan RuntimeException ditangkap maka pernyataan System.out.println("Error karena pembagian nol !"); akan dieksekusi. Untuk kata kunci finally ini akan selalu dieksekusi walaupun kesalahan sudah ditangkap pada catch atau walaupun program tidak memiliki kesalahan. Fungsi finally yaitu untuk kasus yang tidak sesuai dengan pernyataan-pernyataan catch. Ini mirip dengan kata kunci default pada pernyataan switch..case.

Percobaan3

```
public class TesThrow {
public static void main(String[] args){
    int[] lariksaya = new int[10];
    try{
        lariksaya[100] = 1;
    }
    catch (ArrayIndexOutOfBoundsException e)
    {
   e = new ArrayIndexOutOfBoundsException("Masukkan elemen yang
diperkenankan!");
        throw(e);
    }
} }
```

Hasil :

Exception in thread "main"

```
java.lang.ArrayIndexOutOfBoundsException: Masukkan elemen yang diperkenankan! at pertemuan7.TesThrow.main(TesThrow.java:20)
Java Result: 1
```

Pembahasan :

Kata kunci throw memulai konstruksi throw, diikuti dengan daftar parameter dimana parameter ini merupakan objek eksepsi dan objek dideklarasikan sebagai jenis eksepsi (exception) yang dikehendaki. Dalam Java (yaitu dalam java.lang.Exception) atau exception-exception (exception) yang kita tentukan sendiri.

Saat exception dilemparkan menggunakan pernyataan throw, kode yang sedang berjalan akan dihentikan. Eksepsi (exception) akan dilewatkan ke rutin pemanggil dan tidak ada kode yang dijalankan hingga eksepsi ditangani di suatu tempat di bagian lain program.

Pada contoh diatas pada pernyataan `lariksaya[100] = 1;` akan menghasilkan exception yang akan ditangkap oleh pernyataan `catch (ArrayIndexOutOfBoundsException e)` sehingga, pernyataan `e = new ArrayIndexOutOfBoundsException("Masukkan elemen yang diperkenankan!");` akan menyampaikan pesan sesuai dengan apa yang kita definisikan, kemudian kita melempat exception tersebut dengan pernyataan `throw(e);` .

Percobaan4

```
public class ThrowingClass {
static void myMethod() throws ClassNotFoundException {
throw new ClassNotFoundException ("just a demo");
} }

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

```
ThrowingClass.myMethod();
} catch (ClassNotFoundException e) {
System.out.println(e);
} } }
```

Hasil :

```
java.lang.ClassNotFoundException: just a demo
```

Pembahasan :

Jika sebuah method dapat menyebabkan sebuah exception namun tidak menangkapnya, maka digunakan keyword throws. Aturan ini hanya berlaku pada checked exception. Berikut ini penulisan syntax menggunakan keyword throws :

```
<type> <methodName> (<parameterList>) throws <exceptionList> {
 <methodBody>
 }
```

Sebuah method perlu untuk menangkap ataupun mendaftar seluruh exceptions yang mungkin terjadi, namun hal itu dapat menghilangkan tipe Error, RuntimeException, ataupun subclass-nya.

Contoh diatas menunjukkan bahwa method `myMethod` tidak menangani `ClassNotFoundException` Checked exceptions adalah exception yang diperiksa oleh Java compiler. Compiler memeriksa keseluruhan program apakah menangkap atau mendaftar exception yang terjadi dalam sintax throws. Apabila checked exception tidak didaftar ataupun ditangkap, maka compiler error akan ditampilkan.

Tidak seperti checked exceptions, unchecked exceptions tidak berupa compile-time checking dalam penanganan exceptions. Pondasi dasar dari unchecked exception classes adalah Error, RuntimeException dan subclass-nya.

Percobaan5

```
public class HateStringException extends RuntimeException {
}


public class TestHateString {
public static void main(String args[]) {
String input = "invalid input";
try {
if (input.equals("invalid input")) {
throw new HateStringException(); }
System.out.println("String accepted."); }
catch (HateStringException e) {
System.out.println("I hate this string: " + input +
".."); }
}}
```

Hasil :

I hate this string: invalid input.

Pembahasan :

Meskipun beberapa exception classes terdapat pada package java.lang namun tidak mencukupi untuk menampung seluruh kemungkinan tipe exception yang mungkin terjadi. Sehingga sangat mungkin bahwa Anda perlu untuk membuat tipe exception tersendiri.

Dalam pembuatan tipe exception Anda sendiri, Anda hanya perlu untuk membuat sebuah extended class terhadap RuntimeException class, maupun Exception class lain. Selanjutnya tergantung pada Anda dalam memodifikasi class sesuai permasalahan yang akan diselesaikan. Members dan constructors dapat dimasukkan pada exception class milik Anda.

Latihan

Tentukan sebuah huruf bilangan heksadesimal (a-f) sebagai input. Konversi huruf tersebut menjadi bilangan desimal. Tentukan exception class anda sendiri dan lakukan

penanganan jika input dari user bukan merupakan bilangan heksadesimal

Buatlah suatu program yang memilih apakah Mahasiswa tersebut merupakan mahasiswa informatika atau bukan berdasarkan input kode jurusan. Misalkan kode jurusan : 01-Teknik Informatika, 02-Teknik Sipil, 03-Teknik Elektro, 04-Teknik Industri, 05-Teknik Arsitektur, 06-Teknik Kriya. Jika yang dimasukkan bukan kode (misalkan huruf) maka akan ditangani kesalahan tersebut menggunakan try..catch

BAB 8

ABSTRACT WINDOWING TOOLKIT (AWT) DAN SWING

8.1 Tujuan

Memahami persamaan dan perbedaan antara AWT dan Swing

Mengetahui perbedaan antara komponen dan kontainer

Mendesain aplikasi GUI menggunakan AWT

Mendesain aplikasi GUI menggunakan Swing

Menjelaskan tentang *flow layout, border layout, dan grid layout* dalam komponen GUI

8.2 Latar Belakang

Tanpa mempelajari tentang *Grapichal User Interface* (GUI) API. Anda masih tetap bisa membuat suatu program. Tetapi, program anda akan kelihatan tidak menarik dan tidak nyaman digunakan bagi para user. Memiliki GUI yang baik dapat memberi efek pada penggunaan aplikasi. Java menyediakan banyak *tool* seperti *Abstract Windowing Toolkit* (AWT) dan *Swing* untuk mengembangkan aplikasi GUI yang interaktif.

8.3 Percobaan1 (TesLabel.java)

```
import java.awt.*;
public class TesLabel extends java.applet.Applet{
    @Override
public void init(){
   setFont(new Font ("Arial", Font.BOLD, 14));
   setLayout(new GridLayout(3, 1));
   add(new Label("rata kiri", Label.LEFT));
   add(new Label("rata tengah", Label.CENTER));
   add(new Label("rata kiri", Label.LEFT));
}
}
```

Hasil :



Pembahasan :

Bentuk paling sederhana dari komponen UI adalah label, yakni string teks yang dapat kita pakai untuk memberi label pada komponen UI lainnya. Label tidak bisa diedit.

Keunggulan label atas string teks lainnya adalah :

Kita tidak perlu menggambar ulang label sebagaimana string teks. Label adalah elemen **awt** sehingga penggambarannya adalah tanggung jawab **awt.**

Label mengikuti tata letak panel dimana ia berada dan dapat ditata bersama dengan komponen UI lainnya.

Label adalah string teks yang tidak dapat diedit yang bekerja sebagai pendeskripsi komponen **awt** lain. Untuk membuat label gunakan konstruktor berikut ini :

Label() membuat label kosong, dengan teks tertera rata kiri

Label(String) membuat label dengan string teks tertentu, juga tertata rata kiri

Label(String,int) membuat label dengan string teks dan cara pengaturan tertentu. Angka yang menyatakan penataan teks disimpan dalam variabel

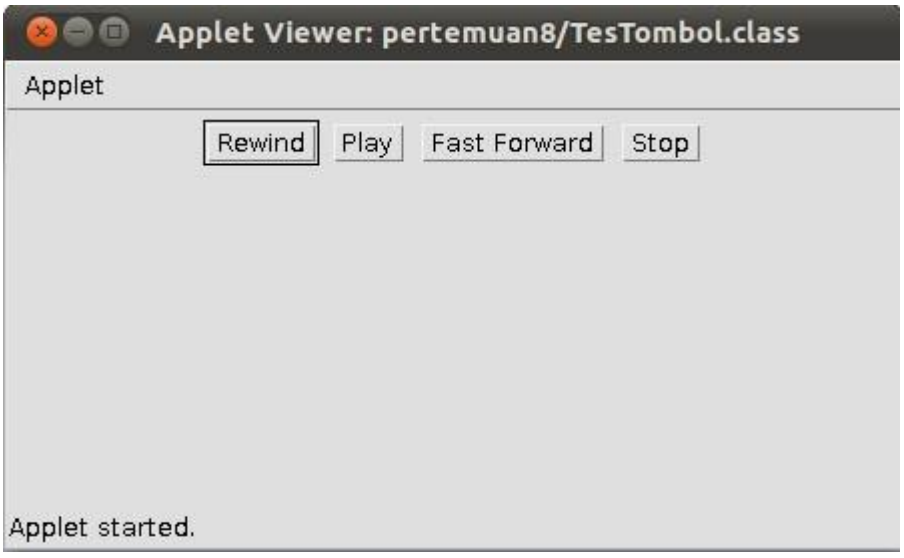
kelas dalam **Label** sehingga mudah diingat **Label.RIGHT, Label.LEFT,** dan **Label.CENTER**

Kita dapat mengganti font label dengan menggunakan metode **setFont()**, baik dipanggil pada label itu sendiri untuk mengubah masing-masing label, atau pada komponen untuk mengubah seluruh label.

8.4 Percobaan2 (TesTombol.java)

```
import java.awt.*;
public class TesTombol extends java.applet.Applet {
    @Override
public void init(){
    add(new Button("Rewind"));
    add(new Button("Play"));
    add(new Button("Fast Forward"));
    add(new Button("Stop"));
}
}
```

Hasil :



Pembahasan :

Button adalah komponen antarmuka UI yang memicu beberapa aksi dalam

antarmuka saat di-klik. Untuk membuat button atau tombol ini dapat digunakan konstruktor-konstruktor berikut ini :

Button() untuk membuat tombol kosong tanpa label

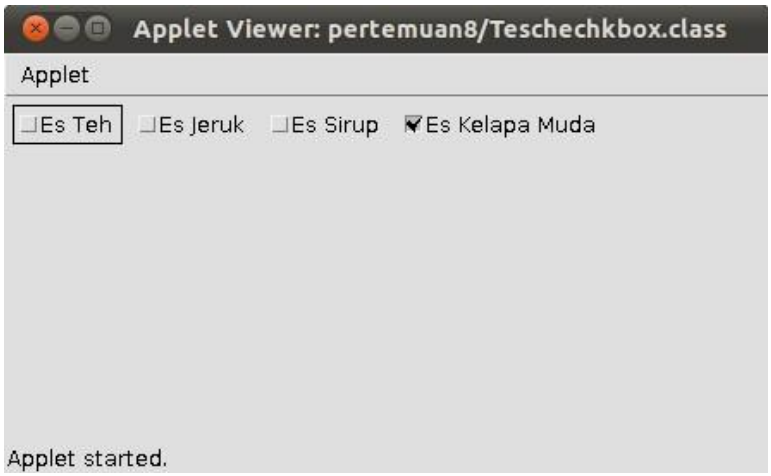
Button(String) untuk membuat tombol dengan string tertentu sebagai label

Setelah memiliki objek **Button,** kita dapat mengambil nilai labelnya dengan metode **getLabel()** dan mengatur labelnya dengan menggunakan metode **setLabel(String)**

8.5 Percobaan3 (TesCheckbox.java)

```
import java.awt.*;
    public class Teschechkbox extends java.applet.Applet{
        @Override
    public void init(){
        setLayout(new FlowLayout(FlowLayout.LEFT));
        add(new Checkbox("Es Teh"));
        add(new Checkbox("Es Jeruk"));
        add(new Checkbox("Es Sirup"));
        add(new Checkbox("Es Kelapa Muda",null,true));
        add(new Checkbox("Es Campur"));
    }
    }
```

Hasil :



Pembahasan :

Checkbox adalah komponen UI yang memiliki dua kondisi : on dan off (atau dipilih dan tidak dipilih, ditandai dan tidak ditandai, benar atau salah, dan sebagainya). Tidak seperti button, checkbox biasanya tidak memicu aksi langsung dalam UI, tetapi digunakan untuk menunjukkan fitur tambahan atau aksi lain.

Checkbox dapat digunakan dalam dua cara :

Non-eksklusif. Diberikan seperangkat check box, beberapa diantaranya dapat dipilih

Eksklusif. Diberikan seperangkat check box, hanya satu yang dapat dipilih. Check box eksklusif disebut radio button

Kita dapat membuat check box dengan menggunakan salah satu konstruktor berikut ini :

Checkbox() untuk membuat check box kosong, dalam keadaan tidak terpilih

Checkbox(String) untuk membuat check box dengan string tertentu sebagai label

Checkbox(String, null, boolean) untuk membuat checkbox yang dipilih atau tidak dipilih berdasarkan argumen boolean **true** atau **false**

8.6 Percobaan4 (CheckboxGroupTest.java)

```
import java.awt.*;
public class CheckboxGroupTest extends java.applet.Applet{
    @Override
public void init(){
    setLayout(new FlowLayout(FlowLayout.LEFT));
    CheckboxGroup cbg = new CheckboxGroup();
    add(new Checkbox("Red", cbg, false));
    add(new Checkbox("Blue", cbg, false));
    add(new Checkbox("Yellow", cbg, false));
```

```
    add(new Checkbox("Green", cbg, false));
    add(new Checkbox("Orange", cbg, false));
    add(new Checkbox("Purple", cbg, false));
}
}
```

Hasil :



Pembahasan :

Radio button memiliki penampilan yang sama dengan check box, tetapi hanya satu dari sekelompok yang dapat dipilih atau ditandai. Untuk membuat sekelompok radio button, mula-mula harus dibuat instance dari **CheckboxGroup**.

```
CheckboxGroup cbg = new CheckboxGroup();
```

Kemudian kita buat dan kita tambahkan satu persatu check box menggunakan konstruktor dengan 3 (tiga) argumen (yakni label, grup, dan kondisi checkbox). Harap diingat bahwa karena radio button, perdefinisi hanya memiliki satu checkbox yang terpilih pada satu saat, maka kondisi **true** terakhirlah yang akan dipilih secara *default.*

```
add(new Checkbox( “Tes” , cbg, true);
add(new Checkbox( “Tes” , cbg, true);
```

8.7 Percobaan5 (TesChoice.java)

```
import java.awt.*;
public class TesChoice extends java.applet.Applet {
    @Override
public void init(){
    Choice c = new Choice();
    c.addItem("Novel");
    c.addItem("Komik");
    c.addItem("Kumpulan Puisi");
    c.addItem("Majalah");
    c.addItem("Koran");
        Component add = add(c);
}
}
```

Hasil :



Pembahasan :

Menu pilihan (choice menu) adalah komponen UI yang lebih konpleks dari pada label, *button* ataupun *checkbox*. Menu pilihan adalah menu pop-up (atau pull-down) dimana dapat memilih satu dari beberapa item yang tersedia. Kemudian

menu akan menampilkan pilihan tersebut pada layar. Fungsi menu pilihan ada dasarnya sama untuk semua platform, tetapi tampilannya berbeda-beda tergantung platform dimana menu tersebut dijalankan.

Untuk membuat menu pilihan kita harus membuat instance dari kelas **Choice** dan kemudian menggunakan metode **addItem()** untuk menambahkan satu-persatu item dalam urutan tertentu.

8.8 Percobaan6 (TesTextField.java)

```
import java.awt.*;
public class TesTextField extends java.applet.Applet {
    @Override
public void init() {
    add(new Label("Ketikkan nama Anda")); add(new
    TextField("Nama anda disini",45)); add(new
    Label("Ketikkan nomor telepon anda")); add(new
    TextField(25));
    add(new Label("Ketikkan password yang anda inginkan"));
    TextField t = new TextField(20);
    t.setEchoChar('*');
    add(t);
}
}
```

Hasil :



Pembahasan :

Text field adalah komponen UI yang memungkinkan kita memasukkan dan mengedit teks. Biasanya text field hanya memiliki satu baris tampilan dan tidak mempunyai penggulung (scrollbar). Berbeda dengan label, text field dapat diedit, sehingga cocok untuk mengambil masukan teks dari pengguna, sedangkan label hanya cocok untuk menampilkan teks.

Untuk membuat text field, gunakan konstruktor berikut ini :

TextField() untuk membuat **TextField** kosong yang lebarnya 0 karakter

TextField(int) untuk membuat text field kosong. Argumen integer menunjukkan jumlah karakter minimum yang ditampilkan

TextField(String) untuk membuat text field yang diinisialisasi dengan string tertentu

TextField(String,int) untuk membuat text field dengan lebar sejumlah karakter tertentu (bergantung argumen integernya) berisi string tertentu. Jika string lebih lebar dari daripada lebar ketentuannya, kita dapat menggeser kotak teks ke kiri dan ke kanan.

Sebagai contoh, baris berikut ini membuat text field selebar 30 karakter dengan string **"Zinedine Zidane"** sebagai isi awalnya.

```
TextField tf = new TextField( "Zinedine Zidane" ,30);
add(tf);
```

Kita juga dapat membuat text field yang menutupi karakter yang sebenarnya diketik seperti misalnya pada *password*. Untuk membuatnya, mula-mula kita harus membuat text field itu sendiri, kemudian menggunakan metode **setEchoCharacter()** untuk mengatur karakter yang ditampilkan pada layer. Berikut ini adalah contohnya :

```
TextField tf = new TextField(30);
tf.setEchoCharacter('*');
```

8.9 Percobaan7 (TesFlowLayout.java)

```
import java.awt.*;
public class TesFlowLayout extends java.applet.Applet{
       @Override
public void init(){
           setLayout(new  FlowLayout());
           add(new Button("Tombol x"));
           add(new Button("Tombol y"));
           add(new Button("Tombol z"));
           add(new Button("Tombol 1"));
           add(new Button("Tombol 2"));
           add(new Button("Tombol 3"));
       }
}
```

Hasil :



Pembahasan :

Kelas **FlowLayout** adalah layout yang paling dasar. Pada flow layout ini komponen ditambahkan ke panel satu persatu, baris demi baris. Jika komponen tidak pas masuk satu baris, ia akan diteruskan ke baris lainnya. Flow layout juga memiliki tatanan perataan (alignment) yang menentukan penataan tiap baris.

Secara default setiap baris diatur rata tengah (centered)

Untuk membuat penataan yang bersifat mengalir (flow layout) dapat digunakan perintah berikut ini di dalam inisialisasi panel.

setLayout(new FlowLayout());

Untuk membuat penataan flow layout dengan perataan selain rata tengah, kita bisa menambahkan variabel kelas **FlowLayout.RIGHT** atau **FlowLayout.LEFT** sebagai argumen

setLayout(new FlowLayout(FlowLayout.LEFT);

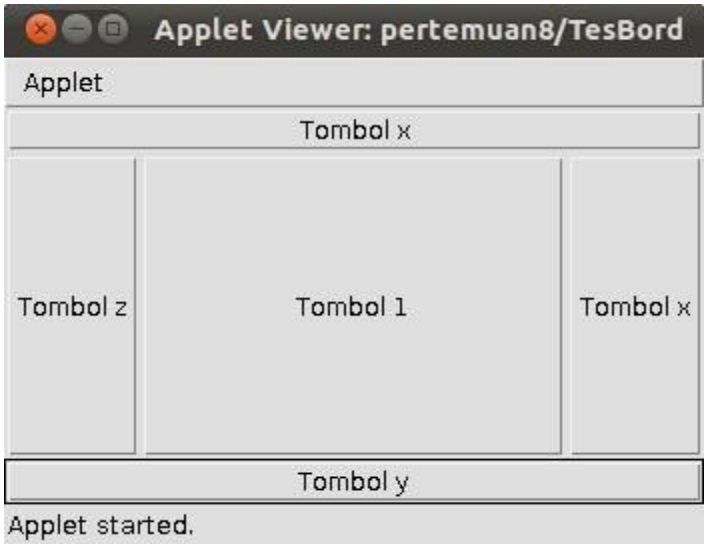
kita juga dapat mengatur gap horisontal dan vertikal dengan menggunakan penataan mengalir. Gap adalah jumlah piksel antarkomponen dalam panel. Secara default, jarak atau gap horisontal dan vertikal ini adalah 3 (tiga) piksel. Penambahan gap dapat dilakukan dengan menambahkan argumen integer ke konstruktor flow layout. Berikut ini menunjukkan hasil penambahan gap 30 piksel horisontal, dan 10 piksel vertikal.

setLayout(new FlowLayout(FlowLayout.LEFT, 30, 10));

8.10 Percobaan8 (tesGridLayout.java)

```
import java.awt.*;
public class TesGridLayout extends java.applet.Applet{
    @Override
public void init(){
    setLayout(new GridLayout(3,2));
    add(new Button("Tombol x"));
        add(new Button("Tombol y"));
            add(new Button("Tombol z"));
                add(new Button("Tombol 1"));
                    add(new Button("Tombol 2"));
                        add(new Button("Tombol 3"));
}
}
```

Hasil :



Pembahasan :

Grid Layout memberikan pengontrolan yang lebih baik dalam penempatan komponen di dalam panel. Dengan menggunakan grid layout kita membagi area tampilan panel ke dalam baris dan kolom. Setiap komponen yang ditambahkan pada panel akan ditempatkan dalam sel, dari baris paling atas menuju baris-baris berikutnya melalui kolom dari kiri ke kanan.

Pembuatan layout pola jala (grid layout) ini dilakukan dengan menentukan jumlah baris dan kolom jala ketika kita membuat instance baru dari kelas GridLayout.

Layout pola jala juga dapat diberi gap horisontal dan vertikal antarkomponen. Untuk membuat gap antarkomponen kita harus menambahkan baris perintah berikut ini :

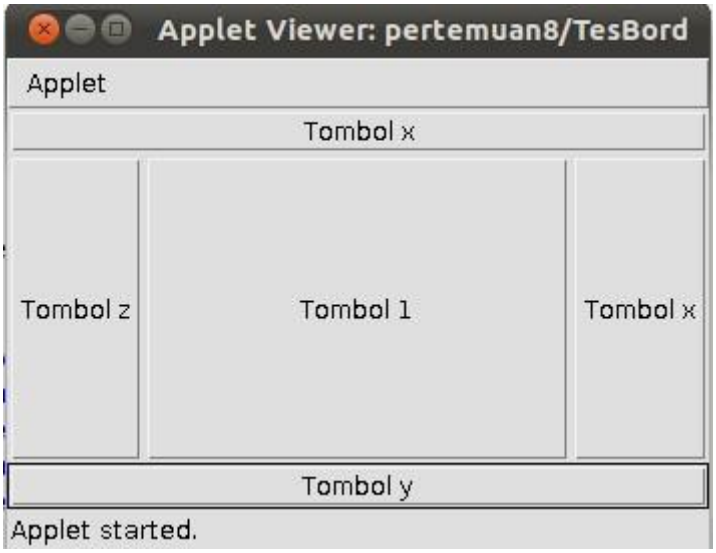
setLayout(new GridLayout(3, 3, 10, 30);

8.11 Percobaan9 (TesBorderLayout.java)

```
import java.awt.*;
public class TesBorderLayout extends java.applet.Applet{
    @Override
    public void init(){
        setLayout(new BorderLayout());
```

```
        add("North",new Button("Tombol x"));
        add("East",new Button("Tombol x"));
        add("South",new Button("Tombol y"));
        add("West",new Button("Tombol z"));
        add("Center",new Button("Tombol 1"));
    }
}
```

Hasil :



Pembahasan :

Pada border layout, penempatan komponen ditunjukkan dengan geografis , yakni utara, selatan, barat, timur, dan tengah. Sama seperti pada layout lainnya, untuk menggunakan border layout kita harus mendefinisikannya, kemudian menambahkan komponen satu persatu dengan metode khusus **add()** yang memiliki dua argumen. Argumen pertama adalah string yang menunjukkan posisi dari komponen dalam layout, dan kedua adalah komponen yang ditambahkan, misalnya :

```
add( "North" , new TextField( "Title" , 50);
```

Kita juga dapat menggunakan bentuk **add()** tersebut untuk pengelola layout lainnya, argumen string akan diabaikan jika tidak diperlukan.

Komponen-komponen dalam border layout bisa dibuat berjarak dari komponen lain. Untuk memberi gap antar komponen, nilai pikselnya disertakan dalam

konstruktor seperti pengelola layout lain.

```
setLayout(new BorderLayout(10,10));
```

8.12 Percobaan10 (TesButtonSwing.java)

```
import javax.swing.*;
public class TesButtonSwing extends JFrame{
    JButton load = new JButton("load");
    JButton save = new JButton("Save");

    public TesButtonSwing(){
        super("Button Beraksi");
        setSize(140,170);
       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel pane = new JPanel();
        pane.add(load);
        pane.add(save);
        add(pane);
        setVisible(true);
    }
    public static void main(String[] args){
    TesButtonSwing t = new TesButtonSwing();
}
}
```

Hasil :



Pembahasan :

Seperti pada package AWT, package dari Swing menyediakan banyak class untuk membuat aplikasi GUI. Package tersebut dapat ditemukan di javax.swing. Perbedaan utama antara keduanya adalah komponen Swing ditulis menyeluruh menggunakan Java. Kesimpulannya, program GUI ditulis menggunakan banyak class dari Package Swing yang mempunyai tampilan look and feel yang sama meski dijalankan pada platform yang berbeda. Lebih dari itu, Swing menyediakan komponen yang lebih menarik seperti color chooser dan option pane.

Nama dari komponen GUI milik Swing hampir sama persis dengan komponen GUI milik AWT. Perbedaan jelas terdapat pada penamaan komponen. Pada dasarnya, nama komponen Swing sama dengan nama komponen AWT tetapi dengan tambahan huruf J pada prefixnya. Sebagai contoh, satu komponen dalam AWT adalah button class. Sedangkan pada Swing, nama komponen tersebut menjadi JButton Class.

Pada kode berikut ini :

```
JButton load = new JButton("load");
JButton save = new Jbutton("Save");
```

Ini membuat objek `load` dan `save` dengan menggunakan kelas JButton pada Swing.

8.13 Percobaan11 (FrameSwing.java)

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

public class FrameSwing {
JFrame frame;
JPanel panel;
JTextField textfield;
JButton button;
Container contentpane;
```

```
void launchframe(){
//inisialisasi
frame = new JFrame("Frameku");
panel = new JPanel();
textfield = new JTextField("Teks");
button=new JButton("Klik saya");
contentpane = frame.getContentPane();
//menambahkan komponen-komponen ke panel menggunakan
//Flowlayout sebagai default
panel.add(textfield);
panel.add(button);
//menambahkan komponen-komponen contentpane
//menggunakan BorderLayout
contentpane.add(panel, BorderLayout.CENTER);
frame.pack();
//menyebabkan ukuran frame menjadi dasar pengaturan komponen
frame.setVisible(true);
}

public static void main(String[] args){
    FrameSwing f = new FrameSwing();
    f.launchframe();
}
}
```

Hasil :



Pembahasan :

Top-level containers seperti Jframe dan Japplet dalam Swing sangat tidak cocok dengan AWT. Ini adalah syarat menambahkan komponen ke dalam kontainer. Jika anda ingin menambahkan langsung sebuah komponen kedalam kontainer sebagai kontainer AWT, pertama-tama anda telah mendapatkan content pane dari kontainer. Untuk melakukan hal tersebut, anda akan menggunakan method `getContentPane` dari container.

Perlu diperhatikan pada package java.awt masih saja diimpor karena layout manager yang digunakan terdapat pada package tersebut. Juga, memberi judul pada frame dan mengepack komponen di dalam frame dapat juga dilakukan untuk frame AWT.

8.14 Percobaan12 (BorderLayoutSwing.java)

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

public class BorderLayoutSwing extends JFrame{
JButton Tombol1 = new JButton("Utara");
JButton Tombol2 = new JButton("Selatan");
JButton Tombol3 = new JButton("Timur");
JButton Tombol4 = new JButton("Barat");
JButton Tombol5 = new JButton("Tengah");

public BorderLayoutSwing(){
   super("Border Layout Beraksi");
```

```
    setSize(240,280);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLayout(new BorderLayout());
    add(Tombol1, BorderLayout.NORTH);
    add(Tombol2, BorderLayout.SOUTH);
    add(Tombol3, BorderLayout.EAST);
    add(Tombol4, BorderLayout.WEST);
    add(Tombol5, BorderLayout.CENTER);
}

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

Hasil :



Pembasahan :

Program diatas sebenarnya hampir sama dengan membuat BorderLayout pada AWT, karena untuk mengatur layout tombol baik untuk posisi NORTH, SOUTH, EAST, WEST, dan CENTER. Hanya saja untuk membuat frame disini kita menggunakan kelas JFrame yang ada pada Swing.

8.15 Latihan

a) Buatlah 5 buah tombol dan textfield dengan menggunakan Flow, Grid dan Border Layout baik menggunakan Java AWT maupun Swing !

BAB 9

GUI EVENT HANDLING

9.1 Tujuan

Menerangkan komponen-komponen delegation event model

Mengerti bagaimana delegation event model bekerja

Menciptakan aplikasi GUI yang berinteraksi dengan user

9.2 Latar Belakang

Pada bagian ini anda akan diperkenalkan dengan beberapa perintah kejadian yang sering dipakai dalam membuat program aplikasi. Ini sangat penting anda kuasai, supaya kualitas pemrograman anda bisa tambah sempurna.

9.3 Percobaan1

(a) Buat project baru dengan memilih perintah **File > New Project.. (Ctrl+Shift+N)**

(b) Muncul jendela **New Project,** pilih Java di kotak **Categories** dan pilih **Java Application** di kotak **Projects**

(c) Klik **Next >**, muncul jendela **New Java Project** dan ketik **Modul9a** di kotak **Project Name**, jangan pilih opsi **Create Main Class** kita akan menset main classnya setelah desain dan programnya selesai.

(d) Klik **Finish**

(e) Buat form baru, klik kanan pada Project **Modul9a > New > Jframe Form** beri nama pada **Class Name** yaitu **Event1**

(f) Tambahkan Jpanel pada form, kemudian tambahkan 3 Jtextfield, 3 JLabel dan 2 Jbutton pada Jpanel tersebut di tab **Design** dengan properties sebagai berikut :

No	Nama Kelas	Properties	Event
1	JLabel	Name = jLabel1 Text = Nama Mahasiswa	
2	JLabel	Name = jLabel2 Text = Nilai Ujian	
3	JLabel	Name = jLabel3 Text = Keterangan	
4	JTextField	Name = jTextField2 Text = null / dikosongkan	

5	JTextField	Name = jTextField2 Text = null / dikosongkan	FocusLost
6	JTextField	Name = jTextField3 Text = null / dikosongkan	
7	JButton	Name = jButton1 Text = Mulai	ActionPerformed
8	JButton	Name = jButton2 Text = Keluar	ActionPerformed
9	Jpanel	Name = jPanel1 TitleBorder = Program Event Handling 1	

(g) Klik kanan pada **jButton2** pilih menu **Events > Action > ActionPerformed,** maka akan muncul pada tab **Source,** sebuah method **Private void jButton2ActionPerformed(java.awt.event.ActionEvent evt)**, pada badan method tersebut tambahkan baris perintah berikut ini :

```
System.exit(0);
```

(h) Kembali ke tab **Design,** klik kanan pada **jButton1** pilih menu **Events > Action > ActionPerformed,** maka akan muncul pada tab **Source,** sebuah method **Private void jButton1ActionPerformed(java.awt.event.ActionEvent evt),** pada badan method tersebut tambahkan baris perintah berikut ini :

```
jTextField1.setText( “” );
jTextField1.setText( “” );
jTextField1.setText( “” );
```

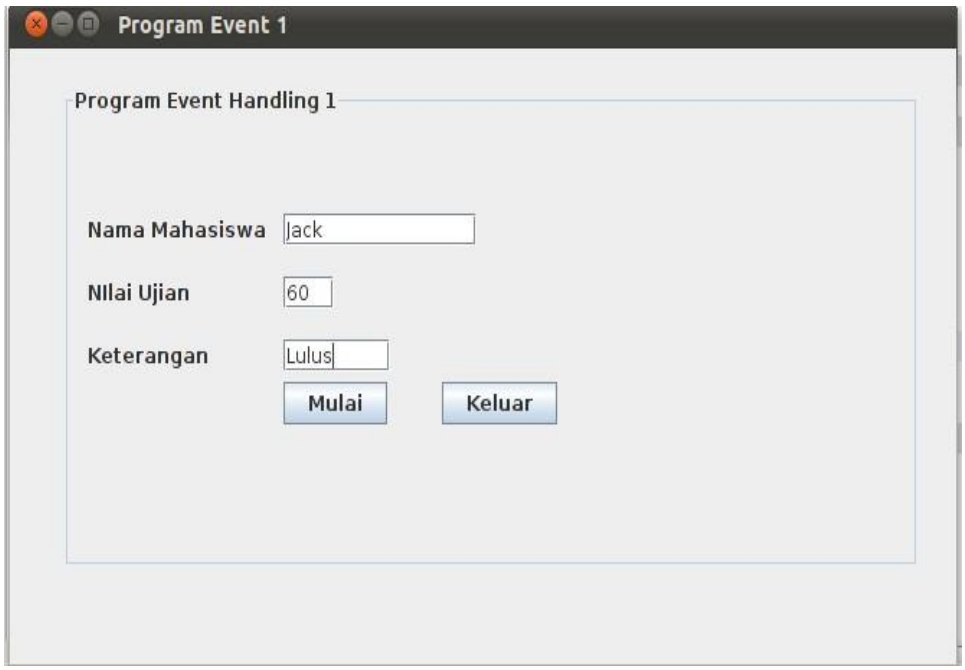
(i) Kembali ke tab **Design,** klik kanan pada **jTextField2** pilih menu **Events > Action > ActionPerformed,** maka akan muncul pada tab **Source** method **Private void JtextField2FocusLost (java.awt.event.FocusEvent evt),** pada badan method tersebut tambahkan baris perintah berikut ini :

```
float nilai;
nilai = Float.parseFloat(jTextField2.getText());
if(nilai>=55) {
jTextField3.setText( “Lulus” );
}
else {
```

```
jTexField3.setText( “Tidak lulus” ); }
```

(j) Set Main Class Project **Modul9a**, klik kanan pada Project **Modul9a** > **Properties.** Pilih **Categories > Run,** browse **Main Class** pilih **Event1.** Lalu jalankan program anda.

Hasil :



Pembahasan :

Salah satu interface **Event Listener** pada AWT yaitu **Action Listener** yang berfungsi ketika user melakukan klik pada **Button**, menekan tombol **Enter / Tab** saat memasukkan input dalam text field atau memilih menu item. Satu metode pada **Action Listener** yaitu **ActionPerformed,** dimana method **Private void jButton2ActionPerformed(java.awt.event.ActionEvent evt)** berisi perintah `System.exit(0);` dimana program akan berhenti jika tombol **Keluar** ditekan. Untuk method **Private void jButton1ActionPerformed(java.awt.event.ActionEvent evt)** berisi perintah `jTextField1.setText( “” );` dimana akan menset semua text field dengan string kosong lagi jika tombol `Mulai` ditekan.

Untuk method **Private void JtextField2FocusLost (java.awt.event.FocusEvent evt)** merupakan salah satu method yang terdapat pada interface **FocusListener** yang berfungsi ketika komponen memperoleh keyboard focus. Pada badan method tersebut

jika sebuah nilai yang dimasukkan pada **JtextField2** bernilai lebih atau sama dengan 55 maka akan menset sebuat text pada **JtextField3** yaitu sebuah String “Lulus” selain itu akan diset ke “Tidak Lulus”.

9.4 Percobaan2

(a) Buat project baru dengan memilih perintah **File > New Project.. (Ctrl+Shift+N)**

(b) Muncul jendela **New Project,** pilih Java di kotak **Categories** dan pilih **Java Application** di kotak **Projects**

(c) Klik **Next >**, muncul jendela **New Java Project** dan ketik **Modul9b** di kotak **Project Name**.

(d) Klik **Finish**

(e) Buat form baru, klik kanan pada Project **Modul9b > New > Jframe Form** beri nama pada **Class Name** yaitu **Event2**

(f) Tambahkan 2 jLabel, 1 jTextField, 1 jPasswordField, dan 1 jButton ke dalam form tersebut dengan properties sebagai berikut :

No	Nama Kelas	Properties	Event
1	JLabel	Name = jLabel1 Text = ID User	
2	JLabel	Name = jLabel2 Text = Password	
3	JTextField1	Name = jTextField1	KeyPressed
4	JPasswordField	Name = jPasswordField1	KeyPressed
5	JButton	Name = jButton1 Text = OK	ActionPerformed

(g) Tambahkan event klik pada tombol **OK** (**jButton1**), klik kanan pada tombol **OK > Events > Action > actionPerformed.** Maka akan muncul sebuah method **private void jButton1ActionPerformed** pada tab **Source**. Tambahkan baris perintah berikut pada badan method tersebut :

```
JOptionPane.showMessageDialog(this, “Selamat Datang “ + jTextField1.getText(), “Pesan Login”, JOptionPane.INFORMATION_MESSAGE);
```

(h) Tambahkan event yang kedua untuk **jTextField1.** Klik kanan pada **jTextField1** >

Events > Key > KeyPressed. Maka akan muncul sebuah method **private void jTextField1KeyPressed** pada tab **Source.** Tambahkan baris perintah berikut pada badan method tersebut :

```
if (evt.getKeyCode() == 10) {
jPasswordField1.requestFocus(); }
```

(i) Tambahkan event yang ketiga untuk **jPasswordField1.** Klik kanan pada **jPasswordField1** > **Events > Key > KeyPressed.** Maka akan muncul sebuah method **private void jPasswordField1KeyPressed** pada tab **Source.** Tambahkan baris perintah berikut pada badan method tersebut :

```
if (evt.getKeyCode() == 10) {
jButton1.requestFocus(); }
```

(j) Lakukan perubahan pada kelas **Main,java**, agar melakukan pemanggilan form **Event2** yaitu dengan baris perintah sebagai berikut :

```
new Event2().setVisible(true);
```

(k) Jalankan program anda.

Hasil :



Pembahasan :

Event pada **jButton1 (tombol OK)** yaitu **ActionPerformed** digunakan ketika user menekan tombol **jButton1 (tombol OK).** Kelas JoptionPane memiliki method showMessageDialog. Method ini bisa menampilkan berbagai jenis pesan / jendela dialog. Dalam hal ini dialognya adalah INFORMATION_MESSAGE, yaitu jenis

jendela pesan informasi saja yang dilengkapi dengan satu tombol OK. Parameter pertama adalah isi pesan yang ditampilkan, sedangkan parameter kedua adalah nama jendela pesan ini, parameter ketiga adalah jenis dialog / jendela.

Event kedua dan ketiga untuk jTextField1 dan jPasswoField1 yaitu KeyPressed, hal ini untuk memberikan respon ketika user memasukkan ID user dan menekan tombol **Enter,** maka kursor secara otomatis akan berpindah ke isian berikutnya (password). Hal ini dapat dilakukan dengan memanfaatkan kemampuan event, dengan kategori **Key** untuk interface **KeyListener** pada method **keyPressed().**

Perintah **getKeyCode()** adalah untuk mendapatkan nilai angka unik untuk setiap tombol pada keyboard. Dalam hal ini java mengenali enter sebagai angka 10. selanjutnya **requestFocus()** untuk memerintahkan kursor berpindah ke **jPasswordField1** (isian password).

Perintah `if(evt.getKeyCode() == 10)` adalah untuk melakukan pengujian kondisi dari nilai angka dari keyboard yang ditekan. Jika menekan **Enter**, kursor akan mengarah ke isian data berikutnya.

9.5 Percobaan3

(a) Buat project baru dengan memilih perintah **File > New Project.. (Ctrl+Shift+N)**

(b) Muncul jendela **New Project,** pilih Java di kotak **Categories** dan pilih **Java Application** di kotak **Projects**

(c) Klik **Next >**, muncul jendela **New Java Project** dan ketik **Modul9c** di kotak **Project Name**.

(d) Klik **Finish**

(e) Buat form baru, klik kanan pada Project **Modul9c > New > Jframe Form** beri nama pada **Class Name** yaitu **Event3**

(f) Tambahkan 3 jLabel, 2 jTextField dan 1 jCombobox ke dalam form tersebut dengan properties sebagai berikut :

No	Nama Kelas	Properties
1	JLabel	Name = jLabel1 Text = Nama Kota horizontalAlignment = LEFT
2	JLabel	Name = jLabel2 Text = Besar UMR horizontalAlignment = LEFT

3	JLabel	Name = jLabel3 Text = Makanan Khas horizontalAlignment = LEFT
4	JcomboBox	Name = jComboBox1 Model = Yogyakarta, Surabaya, Jakarta selectedIndex = 1
5	JTextField	Name = txtUMR
6	JTextField	Name = txtMakanan

(g) Untuk merubah nama variabel pada **jTextField1**, klik kanan pada **jTextField1 > Change Variable Name..** isikan dengan **txtUMR**. Lakukan hal yang sama pada **jTextField2** dengan nama variabel **txtMakanan**.

(h) Tambahkan sebuah event **ItemStateChanged** pada pilihan nama kota **(jComboBox1)**, dengan cara klik kanan **jComboBox1 > Events > Item > ItemStateChanged.**

(i) Perbaiki metode **jComboBox1ItemStateChanged** pada tab **Source** yaitu sebagai berikut :

```
if(jComboBox1.getSelectedIndex() == 0){
          txtUMR.setText("500.000");
          txtMakanan.setText("Gudeg");
         }
else if (jComboBox1.getSelectedIndex() == 1){
    txtUMR.setText("900.000");
    txtMakanan.setText("Rawon");
}
else if (jComboBox1.getSelectedIndex() == 2){
    txtUMR.setText("1.000.000");
    txtMakanan.setText("Ketoprak");
}
```

(j) Lakukan perubahan pada kelas **Main.java,** agar melakukan pemanggilan form Event3, yaitu dengan menambahkan baris berikut ini :

```
new Event3().setVisible(true);
```

(k) Jalankan program anda

Hasil :



Pembahasan :

Pada form ini akan menampilkan informasi mengenai kota-kota tertentu. Dengan memilih nama kotanya, akan ditampilkan informasi mengenai besarnya UMR (ipah minimum regional) dan informasi mengenai jenis makanan khasnya. Hal ini mendemonstrasikan event **ItemStateChanged,** yakni kejadian ketika user memilih pilihan pada ComboBox kota.

Pada method **jCombobox1ItemStateChanged** pada baris perintah **jComboBox1.getSelectedIndex()** fungsinya adalah untuk mendapatkan nilai indeks yang terpilih pada sebuah jComboBox. Indeks ini dimulai dari 0 (pilihan pertama) dan seterusnya. Sedangkan perintah **setText()** adalah untuk memberikan nilai text baru pada sebuah jTexField.

9.6 Percobaan4

(a) Buat project baru dengan memilih perintah **File > New Project.. (Ctrl+Shift+N)**

(b) Muncul jendela **New Project,** pilih Java di kotak **Categories** dan pilih **Java Application** di kotak **Projects**

(c) Klik **Next >**, muncul jendela **New Java Project** dan ketik **Modul9d** di kotak **Project Name**, jangan pilih opsi **Create Main Class** kita akan menset main classnya

setelah desain dan programnya selesai.

(d) Klik **Finish**

(e) Buat form baru, klik kanan pada Project **Modul9d > New > Jframe Form** beri nama pada **Class Name** yaitu **Event4.** Tambahkan **JToolBar** pada Form dengan posisi layout **Last.** Tambahkan **jLabel** pada pada toolbar tadi, dengan cara klik kanan pada **jToolbar > Add From Palete > Swing Controls > Label**

(f) Tambahkan Jpanel pada form, kemudian tambahkan 3 Jlabel, 3 jTextField dan 1 jTextButton pada Jpanel tersebut di tab **Design** dengan properties sebagai berikut :

No	Nama Kelas	Properties
1	JLabel	Name = jLabel1
2	JLabel	Name = jLabel2 Text = No MHS
3	JLabel	Name = jLabel3 Text = Nama
4	JLabel	Name = jLabel4 Text = Alamat
5	JTextField	Name = txtNoMhs
6	JTextField	Name = txtNama
7	JTextField	Name = txtAlamat
8	JButton	Name = jButton1 Text = Simpan
9	JPanel	Name = jPanel1 Layout Direction = Center
10	JToolBar	Name = jToolBar1 Layout Direction = Last

(g) Masukkan event **focusGained** yang pertama, dengan cara klik kanan **txtNoMhs > Events > Focus > focusGained.** Tambahkan baris perintah berikut ini pada badan method **private void txtNoMhsFocusGained (java.awt.event.FocusEvent evt)** :

```
jLabel1.setText(“Masukkan nomor mahasiswa”);
```

(h) Masukkan event **focusGained** yang kedua, dengan cara klik kanan **txtNama > Events > Focus > focusGained.** Tambahkan baris perintah berikut ini pada badan method **private void txtNamaFocusGained (java.awt.event.FocusEvent evt)** :

```
jLabel1.setText(“Masukkan nama lengkap mahasiswa”);
```

(i) Masukkan event **focusGained** yang pertama, dengan cara klik kanan **txtAlamat**

> Events > Focus > focusGained. Tambahkan baris perintah berikut ini pada badan method **private void txtAlamatFocusGained (java.awt.event.FocusEvent evt)** :

```
jLabel1.setText("Masukkan alamat domisili mahasiswa");
```

(j) Set Main Class Project **Modul9d**, klik kanan pada Project **Modul9a > Properties.** Pilih **Categories > Run,** browse **Main Class** pilih **Event4.** Lalu jalankan program anda.

Hasil :



Pembahasan :

Form ini akan menunkukkan manfaat dari event **focusGained** (salah satu dari event focus). **FocusGained** akan dibangkitkan ketika sebuah objek menerima focus (kursor aktif), dalam form EventFocus tersebut, ketika kursor masuk dalam kotak isian No MHS, maka status bar akan menampilkan pesan terkait dengan isian No.MHS tersebut ("Masukkan nomor Mahasiswa"). Begitu juga jika kursor aktif pada isian nama dan alamat, maka status bar akan menampilkan informasi / pesan terkait dengan isian tersebut.

9.7 Percobaan5

(a) Buat project baru dengan memilih perintah **File > New Project.. (Ctrl+Shift+N)** Muncul jendela **New Project,** pilih Java di kotak **Categories** dan pilih **Java**

Application di kotak **Projects**

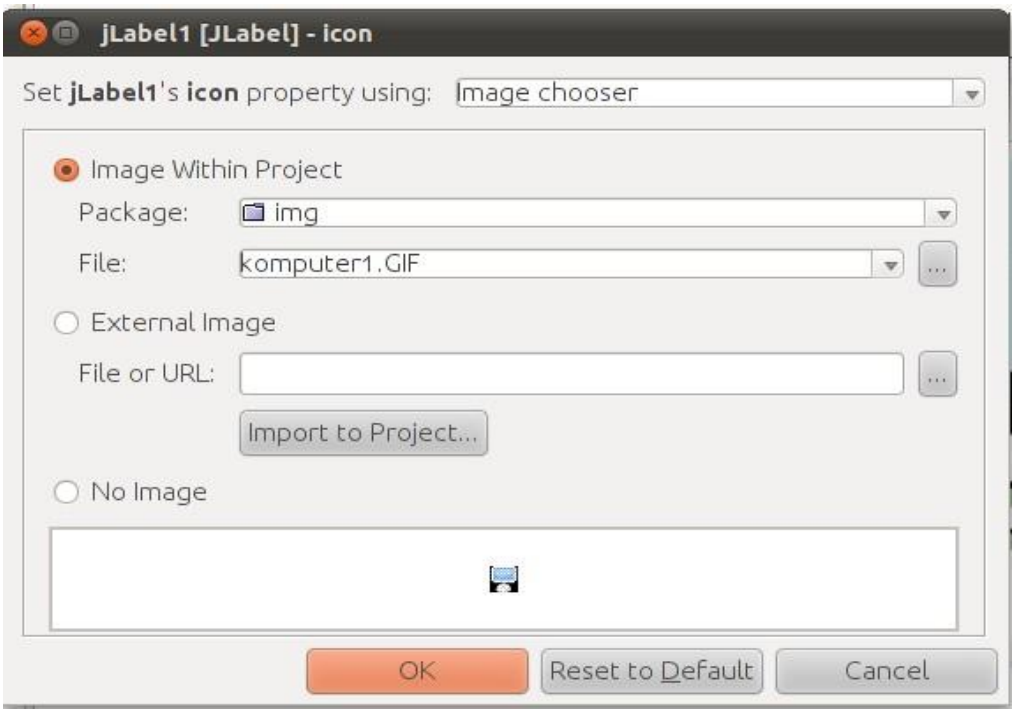
(b) Klik **Next >**, muncul jendela **New Java Project** dan ketik **Modul9e** di kotak **Project Name**, jangan pilih opsi **Create Main Class** kita akan menset main classnya setelah desain dan programnya selesai.

(c) Klik **Finish**

(d) Buat form baru, klik kanan pada Project **Modul9e > New > Jframe Form** beri nama pada **Class Name** yaitu **Event5**

(e) Sebelum merancang form, **copy** file-file gambar pada direktori / folder **src** pada project **Modul9e.** Letakkan file-file gambar tersebut dalam sebuah folder **img** didalam direktori/folder **src.**

(f) Tambahkan pada form sebuah JLabel, atur properti **icon,** arahkan file gambar ke **komputer1.GIF,** sehingga akan tampak seperti gambar dibawah ini :



(g) Tambahkan dua lagi jLabel dengan properti **icon,** arahkan ke file gambar **alat tulis1.JPG** dan **rumah1.JPG**

(h) Tambahkan satu lagi jLabel untuk menampilkan sebuah gambar yang lebih besar, letakkan disebelah kanan jLabel sebelumnya, dan masukan gambar **rumah2.JPG** dalam properti **icon.**

(i) Masukkan event **mouse motion** yang pertama, dengan cara klik kanan **jLabel1 > Events > MouseMotion > mouseMoved.** Selanjutnya tambahkan perintah dalam

event tersebut, sehingga menjadi seperti :

```
private void jLabel1MouseMoved (java.awt.event.MouseEvent evt) {
String imgSouce = "./img/komputer2.PNG" ;
jLabel4.setIcon(new ImageIcon(imgSource));
}
```

(j) Masukkan event **mouse motion** yang pertama, dengan cara klik kanan **jLabel12> Events > MouseMotion > mouseMoved.** Selanjutnya tambahkan perintah dalam event tersebut, sehingga menjadi seperti :

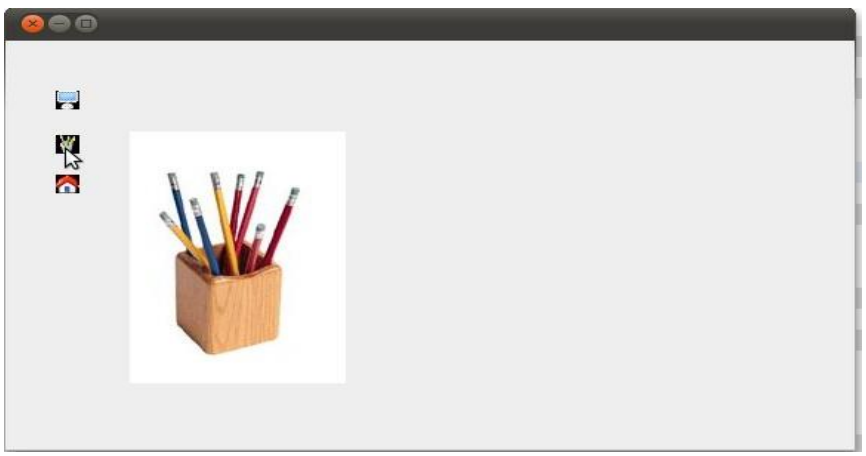
```
private void jLabel2MouseMoved (java.awt.event.MouseEvent evt) {
String imgSouce = "./img/alat tulis2.JPG" ;
jLabel4.setIcon(new ImageIcon(imgSource));
}
```

(k) Masukkan event **mouse motion** yang pertama, dengan cara klik kanan **jLabel3 > Events > MouseMotion > mouseMoved.** Selanjutnya tambahkan perintah dalam event tersebut, sehingga menjadi seperti :

```
private void jLabel3MouseMoved (java.awt.event.MouseEvent evt) {
String imgSouce = "./img/rumah2.PNG" ;
jLabel4.setIcon(new ImageIcon(imgSource));
}
```

(l) Set Main Class Project **Modul9e**, klik kanan pada Project **Modul9e > Properties.** Pilih **Categories > Run,** browse **Main Class** pilih **Event5.** Lalu jalankan program anda.

Hasil :



Pembahasan :

```
Pada baris perintah :
private void jLabel1MouseMoved (java.awt.event.MouseEvent evt) {
String imgSouce = "./img/komputer2.PNG";
jLabel4.setIcon(new ImageIcon(imgSource));
}
```

Event MouseMoved digunakan apabila mouse menuju pada gambar komputer1.GIF maka gambar pada jLabel4 akan memunculkan gambar dari file komputer2.PNG dengan perintah :

```
String imgSouce = "./img/komputer2.PNG";
jLabel4.setIcon(new ImageIcon(imgSource));
```

9.8 Percobaan6

(a) Buat project baru dengan memilih perintah **File > New Project.. (Ctrl+Shift+N)** Muncul jendela **New Project,** pilih Java di kotak **Categories** dan pilih **Java Application** di kotak **Projects**

(b) Klik **Next >**, muncul jendela **New Java Project** dan ketik **Modul9f** di kotak **Project Name**, jangan pilih opsi **Create Main Class** kita akan menset main classnya setelah desain dan programnya selesai.

(c) Klik **Finish**

(d) Buat form baru, klik kanan pada Project **Modul9e > New > Jframe Form** beri nama pada **Class Name** yaitu **Event6**

(e) Masuk dalam design form, pilih properties **title** dari form tersebut, masukkan "Aplikasi Pemesanan Menu"

(f) Tambahkan dua buah jLabel dengan properti **text** masing-masing "Daftar Menu" dan label kedua "Tambahan"

(g) Tambahkan sebuah jComboBox, sebuah jTextArea, dan sebuah jButton. Seperti pada gambar dibawah ini :



(h) Tambahkan event **windowOpened**, dibangkitkan ketika form ini dibuka, dengan cara klik kanan pada area **form > Events > Window > windowOpened.** Tambahkan perintah dibawah ini :

```
private void formWindowOpened(java.awt.event.WindowEvent evt) {
jComboBox1.removeAllitems();
jComboBox1.addItem( “Paket Sarapan” );
jComboBox1.addItem( “Paket Makan Siang ” );
jComboBox1.addItem( “Prasmanan ” );
jComboBox1.addItem( “Paket Keluarga” ); }
```

(i) Tambahkan **event windowClosing**, dibangkitkan ketika form ini ditutup, dengan cara klik kanan pada area **form > Events > Window > windowClosing**. Tambahkan perintah dibawah ini :

```
int closing;
closing=JOptionPane.showConfirmDialog(this, “Apakah anda yakin, mau keluar dari form ini ? “, “konfirmasi”, JOptionPane.YES_NO_OPTION);
if(closing==0) {
this.dispose(); }
```

(j) set pada properties **form**, **defaultCloseOperation = DO_NOTHING**

(k) Set Main Class Project **Modul9f**, klik kanan pada Project **Modul9f** > **Properties.** Pilih **Categories > Run,** browse **Main Class** pilih **Event6.** Lalu jalankan program

anda.

Hasil :



Pembahasan :

Pada events **windowOpened** yang didefinisikan pada form dengan perintah `removeAllItems()` artinya adalah untuk menghapus semua pilihan dalam jCombobox, sedangkan `addItem()` adalah untuk menambahkan sebuah pilihan dalam jCombobox.

Pada events **windowClosing,** pernyataan :

```
int closing;
closing=JOptionPane.showConfirmDialog(this, “Apakah anda yakin, mau keluar dari form ini ? “, “konfirmasi”, JOptionPane.YES_NO_OPTION);
if(closing==0) {
this.dispose(); }
```

Dideklarasikan terlebih dahulu sebuah variabel yang akan digunakan untuk mendefinisikan sebuah kotak dialog konfirmasi apabila sebuah window akan ditutup dengan menggunakan method pada **JOptionPane** yaitu **showConfirmDialog** yang menggunakan model **YES_NO_OPTION**.

BAB 10

JDBC

10.1 Tujuan

Mengkases database menggunakan JDBC

Melakukan operasi baca, tulis, dan hapus data dari database

Menggunakan interface dari yang sederhana sampai yang kompleks

10.2 Latar Belakang

Pemahaman JDBC ini adalah mutlak diperlukan bagi seorang programmer database, jika ingin bekerja dengan Java. Sebagai gambaran saja, jika programmer membuat aplikasi database dengan semisal visual basic / PHP dengan database SQL Server atau MySQL, apakah program tersebut bisa diganti databasenya dengan yang lain semisal PostgreSQL atau Oracle. Dalam hal ini JDBC mampu membuat aplikasi bisa berjalan diatas semua sistem database, hanya cukup dengan memasang driver JDBC dari database tersebut.

10.3 Percobaan1

(a) Terlebih dahulu buatlah database baru dengan nama **modul10a**

(b) Buat Project baru dengan nama **Modul10a**

(c) Edit **main.java** sehingga tampak sebagai berikut :

```
package Modul10a;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class Main {


public static void main(String[] args) {

   Connection con = null;

   try {

Class.forName("com.mysql.jdbc.Driver").newInstance();

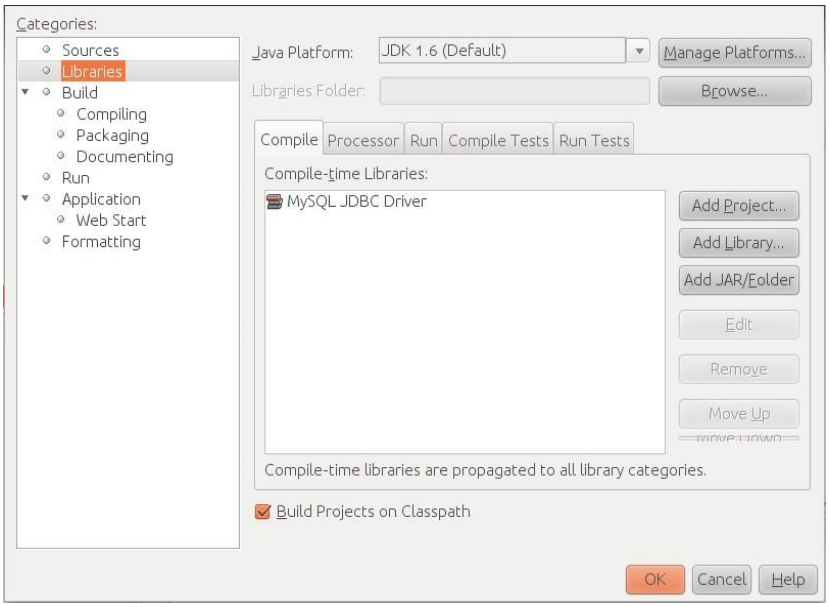
con=         DriverManager.getConnection("jdbc:mysql:///modul10a",

"root", "rahasia");

    if (!con.isClosed())
```

```
System.out.println("Sukses terhubung ke MySQL Server...");
        } catch(Exception e) {
System.err.println("Exception: " + e.getMessage());
        } finally {
            try {
                if (con != null)
                    con.close();  }
        catch(SQLException e) {}
        } } }
```

(d) Tambahkan MySQL JDBC driver dengan cara klik kanan pada project **Modul10a > Properties > Libraries** pada tab **Compile** pilih **Add Library** tambahkan **MySQL JDBC Driver** seperti terlihat pada gambar dibawah ini :



(e) Jalankan program anda

Hasil :

```
run:
Sukses terhubung ke MySQL Server...
BUILD SUCCESSFUL (total time: 1 second)
```

Pembahasan :

Pada deklarasi awal, program ini menggunakan beberapa class pada java.sql dengan perintah `import java.sql.Connection, java.sql.DriverManager` dan `java.sql.SQLException`. Kemudian dibuatlah objek `con` dari kelas `Connection` yang nantinya akan digunakan untuk mengakses database dengan perintah :

```
con= DriverManager.getConnection("jdbc:mysql:///modul10a", "root", "rahasia");
```

DriverManager adalah kelas untuk manajemen driver yang digunakan untuk mengakses database, dalam hal ini database yang ingin diakses adalah modul10a. Jika belum ditambahkan sebuah MySQL JDBC driver kedalam project ini, maka program akan mengeksekusi baris perintah :

```
catch(Exception e) {
System.err.println("Exception: " + e.getMessage()); }
```

10.4 Percobaan2

(a) Butlah tabel user dengan field **user_id (varchar (25)), password (varchar(10)) dan jabatan (varchar(25)).** Jadikan **user_id** sebagai **primary key.**

(b) Tambahkan **user_id : admin**, **password : rahasia** dan **jabatan : administrator** pada tabel user di database **Modul10a**

(c) Buat project baru dengan nama **Modul10b**

(d) Edit **main.java** sehingga tampak sebagai berikut :

```
package Modul10b;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
```

```
public class Main {
    private static String no;
    public static void main(String[] args) {
          Connection con = null;
        try {
           Class.forName("com.mysql.jdbc.Driver").newInstance();
con     =     DriverManager.getConnection("jdbc:mysql:///modul10a",     "root",
"rahasia");
    if (!con.isClosed())
    System.out.println("Sukses terhubung ke MySQL server...");
          Statement st = con.createStatement();
          ResultSet rs = st.executeQuery("Select * from user");
          while(rs.next()){ no=no+1;
              System.out.println(no +")");
     System.out.println("User ID : " +rs.getString("user_id"));
    System.out.println("Password : " +rs.getString("password"));
    System.out.println("Jabatan : " +rs.getString(3));
          }
          st.close();
          rs.close();
        }
        catch(Exception e) {
            System.err.println("Exception : " + e.getMessage());
        }
          finally {
            try {
                if (con != null)
                    con.close();
            } catch(SQLException e) {}
        }    }   }
```

(e) Tambahkan MySQL JDBC driver dengan cara klik kanan pada project **Modul10b > Properties > Libraries** pada tab **Compile** pilih **Add Library** tambahkan **MySQL JDBC Driver.**

(f) Jalankan program anda

Hasil :

```
run:
Sukses terhubung ke MySQL server...
null1)
User ID : admin
Password : rahasia
Jabatan : administrator
```

Pembahasan :

Kelas **Statement** digunakan untuk mengirimkan statemen SQL ke database, tanpa menggunakan paramater. Sebagai contoh pada baris perintah `Statement st = con.createStatement();`. Instance `st` tersebut membuat sebuah statement baca, yaitu "`select * from user`", melalui method `executeQuery` yang dimilikinya. Selain itu, statement tersebut juga bisa melakukan operasi baca dengan metode `executeUpdate()` yang juga dimiliki oleh kelas ini.

Sedangkan kelas **ResultSet** digunakan untuk menyimpan dataset (sekumpulan data) dari hasil statement query "SELECT". Seperti pada baris :

```
ResultSet rs = st.executeQuery("Select * from user");
while(rs.next()){ no=no+1;
      System.out.println(no +")");
System.out.println("User ID : " +rs.getString("user_id"));
```

Instance `rs`, akan menyimpan hasil query yang bisa diakses dengan cara memanggil method getter untuk setiap jenis datanya. Misalnya `getString()` adalah untuk mendapat nilai string, diikuti nama fieldnya. Sedangkan untuk membaca record berikutnya bisa dilakukan dengan method `next()`.

10.5 Percobaan3

(a) Buat project baru dengan nama **Modul10c**

(b) Edit **main.java** sehingga akan tampak sebagai berikut :

```
package Modul10c;
import java.sql.Connection; import
java.sql.DriverManager; import
java.sql.PreparedStatement; import
java.sql.SQLException; public
class Main {
    public static void main(String[] args) {
          Connection con = null;
        try {
      Class.forName("com.mysql.jdbc.Driver").newInstance();
con  =  DriverManager.getConnection("jdbc:mysql:///modul10a",  "root",
"rahasia");
       if (!con.isClosed())
 System.out.println("Sukses terhubung ke MySQL server...");

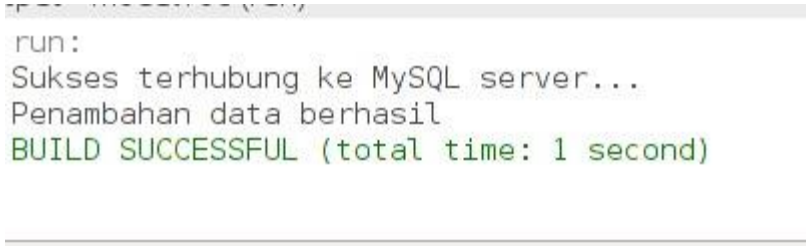
PreparedStatement pStatement = null;
String sql ="insert into user(user_id, password, jabatan)" +"Values
(?,?,?)";
            pStatement = con.prepareStatement(sql);
            pStatement.setString(1, "Joko");
            pStatement.setString(2, "jack");
            pStatement.setString(3, "Manajer Personalia");
            int intTambah= pStatement.executeUpdate();
            if (intTambah>0)
            System.out.println("Penambahan data berhasil");
            else
            System.out.println("Penambahan data gagal");
            pStatement.close();
            con.close();
        }
        catch(Exception e) {
```

```
System.err.println("Exception: " + e.getMessage());
}
  finally {
    try {
        if (con != null)
            con.close();
    } catch(SQLException e) {}
} } }
```

(c) Tambahkan MySQL JDBC driver dengan cara klik kanan pada project **Modul10c > Properties > Libraries** pada tab **Compile** pilih **Add Library** tambahkan **MySQL JDBC Driver.**

(d) Jalankan program anda.

Hasil :



Pembahasan :

Kelas **PreparedStatement** digunakan untuk mengirimkan statement SQL ke database, yang disertai dengan penggunaan parameter, seperti contoh :

```
PreparedStatement pStatement = null;
String sql ="insert into user(user_id, password, jabatan)" +"Values (?,?,?)";
     pStatement = con.prepareStatement(sql);
     pStatement.setString(1, "Joko");
     pStatement.setString(2, "jack");
     pStatement.setString(3, "Manajer Personalia");
     int intTambah= pStatement.executeUpdate();
```

Instance `pStatement` tersebut membuat sebuah statement tulis, yaitu "`insert into`

user (user_id,password,jabatan)", dan dijalankan melalui method executeUpdate() yang dimiliki oleh kelas ini.

Perhatikan method setString([nomor,kolom],[isi dari parameter]); nomor kolom dimulai dari 1, sehingga contoh diatas dapat disimpulkan user_id="Joko", password="jack", jabatan="Manajer Personalia"

10.6 Percobaan4

(a) Buat project baru dengan nama **Modul10d**

(b) Edit **main.java** sehingga akan tampak sebagai berikut :

```
package Modul10d;
import java.sql.*;
public class Main {
    public static void main(String[] args) {
    Connection con = null;
   try {
  Class.forName("com.mysql.jdbc.Driver").newInstance();
con = DriverManager.getConnection("jdbc:mysql:///modul10a", "root",
"rahasia");
     if (!con.isClosed())
 System.out.println("Sukses terhubung ke MySQL server...");
        PreparedStatement pStatement = null;
 String sql ="update user set password=?,"+       "jabatan=? where
user_id=? ";
            pStatement = con.prepareStatement(sql);
            pStatement.setString(1, "s3cr3t");
            pStatement.setString(2, "Manajer IT");
            pStatement.setString(3, "Joko");

            int intBaris= pStatement.executeUpdate();
            if (intBaris>0)
            System.out.println( "Perbaikan data berhasil");
            else
```

```
            System.out.println("Perbaikan data gagal");
            pStatement.close();
            con.close(); }
            catch(Exception e) {
        System.err.println("Exception: " + e.getMessage());
        }
          finally {
            try {
                if (con != null)
                    con.close();
            } catch(SQLException e) {}
        }  }
```

(c) Tambahkan MySQL JDBC driver dengan cara klik kanan pada project **Modul10d > Properties > Libraries** pada tab **Compile** pilih **Add Library** tambahkan **MySQL JDBC Driver.**

(d) Jalankan program anda.

Hasil :

```
run:
Sukses terhubung ke MySQL server...
Perbaikan data berhasil
BUILD SUCCESSFUL (total time: 1 second)
```

Pembahasan :

Pada program **Modul10d** ini tidak jauh berbeda dengan program pada **Modul10c** yang berbeda hanyalah pada baris perintah berikut :

```
String sql ="update user set password=?,"+     "jabatan=? where user_id=? ";
```

Dimana ini adalah perintah untuk melakukan update record yang akan menset password dan jabatan baru yang sesuai dengan user_id.

10.7 Percobaan5

(a) Buat project baru dengan nama **Modul10e**

(b) Edit **main.java** sehingga akan tampak sebagai berikut :

```
package Modul10e;
import java.sql.*;
public class Main {
public static void main(String[] args) {
Connection con = null;
try {
Class.forName("com.mysql.jdbc.Driver").newInstance();
con  =  DriverManager.getConnection("jdbc:mysql:///modul10a",  "root",
"rahasia");
            if (!con.isClosed())
System.out.println("Sukses terhubung ke MySQL server...");
   PreparedStatement pStatement = null;
   String sql ="delete from user " +  " where user_id=? ";
            pStatement = con.prepareStatement(sql);
            pStatement.setString(1, "Joko");
            int intBaris= pStatement.executeUpdate();
            if (intBaris>0)
           System.out.println("Penghapusan data berhasil");
            else
           System.out.println("Penghapusan data gagal");
            pStatement.close();
            con.close(); }
            catch(Exception e) {
       System.err.println("Exception: " + e.getMessage());
        }
          finally {
            try {
                if (con != null)
                    con.close();
            } catch(SQLException e) {}
        } }  }
```

(c) Tambahkan MySQL JDBC driver dengan cara klik kanan pada project **Modul10e > Properties > Libraries** pada tab **Compile** pilih **Add Library** tambahkan **MySQL JDBC Driver.**

(d) Jalankan program anda.

Hasil :

```
run:
Sukses terhubung ke MySQL server...
Penghapusan data berhasil
BUILD SUCCESSFUL (total time: 7 seconds)
```

Pembahasan :

Pada program **Modul10e** ini juga tidak jauh berbeda dengan program pada **Modul10d** yang berbeda hanyalah pada baris perintah berikut :

```
String sql =″delete from user ″ +  ″ where user_id=? ″;
```

Dimana ini adalah perintah untuk melakukan penghapusan record yang yang sesuai dengan user_id.

10.8 Percobaan6

(a) Buat project baru dengan nama **Modul10f**, **jangan pilih** opsi **Create Main Class.**

(b) Tambahkan satu package didalam **Modul10f** dengan cara klik kanan pada project **Modul10f > New > Java Package.** Beri nama **Modul10f**

(c) Tambahkan sebuah Jframe pada project **Modul10f** dengan cara klik kanan pada project **Modul10f > New > Jframe Form.** Beri nama **Tampildata**

(d) Tambahkan sebuah objek **JscrollPane** ke dalam form, atur melebar dibagian tengah form

(e) Edit **Tampildata.java,** tambahkan sebuah **method getData()** dibagian akhir **class Tampildata.java** sebelum tanda akhir kelas/kurung kurawal **"}"** :

```
private Object[][] getData(){
Connection con = null;
```

```
Object[][] data1=null;
  try {
      Class.forName("com.mysql.jdbc.Driver").newInstance();
con   =   DriverManager.getConnection("jdbc:mysql:///modul10a",   "root",
"rahasia");
  if (!con.isClosed())
 System.out.println("Sukses terhubung ke MySQL server...");
      Statement st = con.createStatement();
      ResultSet rs = st.executeQuery("Select * from user");

          rs.last();
          int rowCount=rs.getRow();
          rs.beforeFirst();
          data1= new Object[rowCount][3];
          int no=-1;
          while(rs.next()){ no=no+1;
              data1[no][0]=rs.getString("user_id");
              data1[no][1]=rs.getString("password");
              data1[no][2]=rs.getString("jabatan");
          }
          st.close();
          con.close();
        }
         catch(Exception e) {
       System.err.println("Exception : " + e.getMessage());
         }
           finally {
             try {
```

```
                if (con != null)
                    con.close();
            } catch(SQLException e) {}
        }
    return data1;
    }
```

(f) Tambahkan satu method lagi yaitu **tampilTabel()** untuk inisialisasi Jtable, dan letakkan dibawah method **getData()** :

```
private void tampilTabel(){
   String[] columnNames = {"user_id","password","jabatan"};
        JTable tabel = new JTable(getData(),columnNames);
        jScrollPane1.setViewportView(tabel);
    }
```

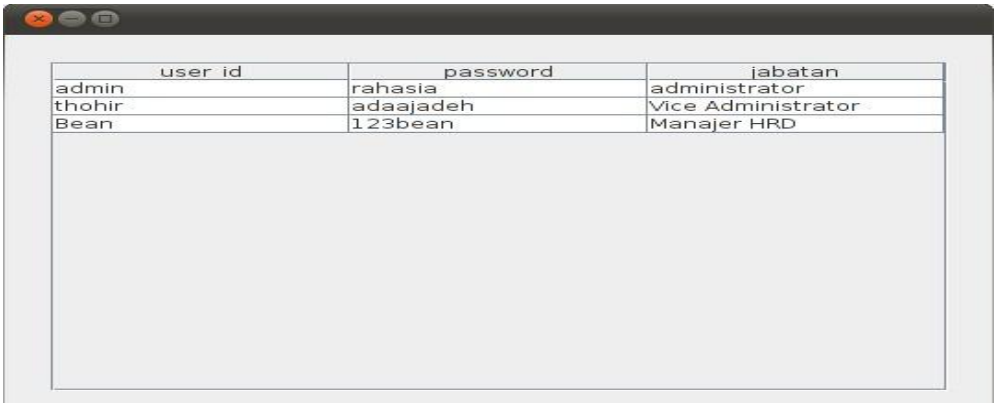
(g) Tambahkan method **tampilTabel()** pada metode konstruktor **Tampildata()**, letakkan dibawah **initComponents() :**

```
    public Tampildata() {
          initComponents();
          tampilTabel();
      }
```

(h) Set **Main Class** pada project **modul10f** dengan cara klik kanan pada project **modul10f > Properties > Run > Main Class > Browse > modul10f.Tampildata**

(i) Jangan lupa tambahkan MySQL JDBC Driver pada project **modul10f**.

Hasil :



Pembahasan :

Pernyataan `rs.last()` adalah menunjuk record paling terakhir

Pernyataan `int rowCount=rs.getRow();` adalah untuk mendapatkan nomor record sekarang/aktif. Hal ini untuk memperoleh jumlah recordnya karena ini akan dibutuhkan untuk melakukan inisialisasi ukuran array pada objek `Jtable` yang akan menampilkannya

Pernyataan `data1[no][0]` adalah untuk menentukan letaknya dalam array, variabel `no` adalah menyatakan baris (record), sedangkan 0 adalah kolomnya.

Parameter aktual inisialisasi Jtable() adalah getData(bertipe array dua dimensi) dan columnNames (berisi array dari judul tabel)

Method `setViewportView()` adalah untuk mengeset objek yang akan ditampilkan dalam ScrollPane. Objek ini memiliki kemampuan untuk menampilkan objek yang dilengkapi dengan layar gulung/scroll.

10.9 Percobaan7

(a) Buat project baru dengan nama **Modul10g,** jangan pilih opsi **Create Main Class**

(b) Tambahkan satu package ke dalam project **Modul10g,** dengan cara klik kanan pada project **Modul10g > New > Java Package.** Beri nama **Modul10g**

(c) Tambahkan sebuah Jframe pada project **Modul10g** dengan cara klik kanan pada project **Modul10g > New > Jframe Form.** Beri nama **FormInput**

(d) Tambahkan **3 buah JLabel, 3 JTextField dan satu JButton**

(e) Atur properties-nya sehingga akan tampil sebagai berikut :



(f) Tambahkan perintah **import** dibawah deklarasi **package Modul10g,** sehingga akan tampil sebagai berikut :

```
package modul10g;
import java.sql.*;
import javax.swing.JOptionPane;
```

(g) Tambahkan event ActionPerformed pada Jbutton dengan cara klik kanan pada **Jbutton1 > events > Action > actionPerformed.** Tambahkan baris perintah, sehingga akan tampak sebagai berikut :

```
    Connection con = null;
    try {
    Class.forName("com.mysql.jdbc.Driver").newInstance();
con = DriverManager.getConnection("jdbc:mysql:///modul10a", "root",
"rahasia");
if (!con.isClosed())
System.out.println("Sukses terhubung ke MySQL
    server...");
 PreparedStatement pStatement = null;
 String sql ="insert into user(user_id, password, jabatan)" +"Values
 (?,?,?)";
        pStatement = con.prepareStatement(sql);
```

```
        pStatement.setString(1,  jTextField1.getText());
        pStatement.setString(2,  jTextField2.getText());
        pStatement.setString(3, jTextField3.getText());

int intTambah= pStatement.executeUpdate();
            if (intTambah>0)
  JOptionPane.showMessageDialog(this,                    "Penambahan
sukses","Informasi",JOptionPane.INFORMATION_MESSAGE);
            else
 JOptionPane.showMessageDialog(this,                    "Penambahan
gagal","Informasi",JOptionPane.INFORMATION_MESSAGE);

            pStatement.close();
            con.close();

            jTextField1.setText("");
            jTextField2.setText("");
            jTextField3.setText("");
        }
        catch(Exception e) {
    System.err.println("Exception: " + e.getMessage());
        }
          finally {
            try {
                if (con != null)
                    con.close();
            } catch(SQLException e) {}
        } }
```

(h) Set **Main Class** pada project **modul10g** dengan cara klik kanan pada project **modul10g > Properties > Run > Main Class > Browse > modul10g.FormInput**

(i) Jangan lupa tambahkan MySQL JDBC Driver pada project **modul10g**.

Hasil :



Pembahasan :

Program ini hampir mirip dengan percobaan3, hanya saja yang membedakan disini menggunakan interaksi GUI Event Handling (**actionPerformed**) dengan konektifitas database (**JDBC**) dan sedikit memanfaatkan kelas JOptionPane untuk menampilkan informasi apakah penambahan data sukses atau tidak.

10.10 Percobaan8

(a) Buat project baru dengan nama **Modul10h,** jangan pilih opsi **Create Main Class**

(b) Tambahkan satu package ke dalam project **Modul10h,** dengan cara klik kanan pada project **Modul10h > New > Java Package.** Beri nama **Modul10h**

(c) Tambahkan sebuah Jframe pada project **Modul10h** dengan cara klik kanan pada project **Modul10h > New > Jframe Form.** Beri nama **FormUpdate**

(d) Tambahkan **3 buah JLabel, 3 JTextField dan dua Jbutton.** Sehingga akan tampak sebagai berikut :



(e) Ganti variabel **Jbutton1** menjadi **btnCekUser**, dengan cara klik kanan pada tombol **jbutton1 > Change Variable Name**

(f) Tambahkan perintah import dibawah project **modul10h.** Sehingga akan tampak sebagai berikut :

```
package modul10h;
import java.sql.*;
import javax.swing.JOptionPane;
```

(g) Tambahkan perintah berikut dibawah deklarasi Jframe :

```
Commenction con = null;
```

(h) Tambahkan perintah berikut dibawah konstruktor **FormUpdate() :**

```
  public FormUpdate() { initComponents(); try {
 Class.forName("com.mysql.jdbc.Driver").newInstance();
con    =    DriverManager.getConnection("jdbc:mysql:///modul10a",
"root", "rahasia");
     if (!con.isClosed())
System.out.println("Sukses terhubung ke MySQL Server...");
        } catch(Exception e) {
  System.err.println("Exception: " + e.getMessage());
        } finally {
            try {
```

```
                    if (con != null)
                        con.close();
                } catch(SQLException e) {}
            }
        }
```

(i) Tambahkan event klik **(actionPerformed)** pada **Jbutton2 (Tombol Simpan) :**

```
try {
    PreparedStatement pStatement = null;
    String sql ="update user" + "set password=?, jabatan=?" + "where user_id=?";
            pStatement = con.prepareStatement(sql);
            pStatement.setString(1, jTextField1.getText());
            pStatement.setString(2, jTextField2.getText());
            pStatement.setString(3, jTextField3.getText());
            int intTambah= pStatement.executeUpdate();
            if (intTambah>0)
    JOptionPane.showMessageDialog(this, "Update sukses","Informasi",JOptionPane.INFORMATION_MESSAGE);
            else
           JOptionPane.showMessageDialog(this, "Penambahan gagal","Informasi",JOptionPane.INFORMATION_MESSAGE);

            pStatement.close();
            con.close();

            jTextField1.setText("");
            jTextField2.setText("");
            jTextField3.setText("");
```

```
}
 catch(SQLException e){
System.out.println("Koneksi gagal "+e.toString());
 } }
```

(j) Tambahkan event klik **jbutton1(tombol Cek User).** Tambahkan baris perintah berikut :

(k)

Referensi

Avestro, Joyce. 2007. JENI Pengenalan Pemrograman 1. Jardiknas. __

Nugroho, Adi. 2008. Algoritma dan Struktur Data dalam Bahasa Java. Penerbit ANDI. Yogyakarta.

Wahono, S. Romi. 2008. Java Fundamentals. ___ . Jakarta