

TEKNOLOGI MULTIMEDIA

Modul 3

J A V A

Oleh

Dr. Ing. Melvi, S.T., M. T.
Aryanto, S.T., M.T>

MODUL 1

PENGENALAN JAVA, EDITOR DAN SINTAKS JAVA

A. Target Pembelajaran

1. Mengetahui bahasa pemrograman JAVA.
2. Mampu membuat kelas sederhana dengan bahasa pemrograman JAVA.
3. Mengetahui penggunaan komentar dalam bahasa pemrograman JAVA.
4. Mengetahui cara menggunakan Netbeans.
5. Mengetahui Penggunaan Character Literal.

B. Materi Dasar

1. Pengenalan JAVA

Java adalah salah satu bahasa pemrograman yang bersifat *open source* yang merupakan produk dari Sun Microsystems dan sekarang di pegang oleh oracle.

Bahasa Java adalah bahasa modern yang telah diterima masyarakat komputasi dunia. Hampir semua perusahaan perangkat lunak dan komputer besar mendukung dan mengembangkan aplikasi sistem berbasis Java.

Java adalah salah satu bahasa pemrograman *Multiplatform* (Bisa berjalan di berbagai macam sistem operasi) karena pada dasarnya java mempunyai Jre (java runtime environment) atau dapat kita artikan sebagai mesin tersendiri untuk mengeksekusi binary code hasil dari kompilasi program yang telah kita buat, berbeda dengan bahasa pemrograman vb, c++ yang memanfaatkan komponen sistem dalam Windows untuk mengeksekusi binary code hasil kompilasi program..

Pada Tahun 1996, Sun Microsystems secara resmi merilis versi awal Java yang kemudian terus berkembang hingga muncul JDK 1.1 (*Java Development Kit versi 1.1*). Perkembangan terus dilakukan hingga muncul versi baru yang disebut Java 2. Perubahan utama antara versi sebelumnya adalah adanya Swing yang merupakan teknologi *Graphical User Interface* (GUI) yang mampu menghasilkan aplikasi *desktop* yang benar-benar baik.

Untuk mengembangkan aplikasi berbasis java, kita memerlukan Java Development Kit(JDK), saat ini java telah mencapai versi 2 yang dikenal dengan java 2 platform. Java 2 platform, dibagi menjadi tiga kategori yaitu:

1. Java 2 Standard Edition (J2SE)
2. Java 2 Enterprise Edition(J2EE)
3. Java 2 Micro Edition(J2ME)

2. Instalasi JDK

Sebelum anda membuat aplikasi dengan java maka anda harus menginstal software bahasa pemrograman java. java dapat anda dapatkan dengan cara download di alamat link berikut <http://java.sun.com>. Kemudian setelah anda mendownloadnya maka mulai dengan menginstall, pada dasarnya proses insalasi java sangat mudah yaitu dengan cara mengklik ganda pada file master java kemudian ikuti prosedur yang diperintahkan.

3. Editor Java

Ada banyak editor yang bisa kita gunakan untuk menuliskan soce code bahasa pemrograman java, bahkan semua editor dimungkinkan selama editor tersebut mampu menyimpan file dengan ekstensi .java yang merupakan ekstensi dari file untuk menyimpan source code bahasa pemrograman java. Beberapa contoh editr yang bias kita pakai misalnya :

- Netbens
- Notepad
- Eclips
- dll

4. Program Java Pertama

Kita akan coba untuk menganalisa program Java pertama pada contoh berikut:

```
public class Hello
{
    /**
     * My first java program
     */
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

Baris pertama kode :

public class Hello

Menandakan nama class yaitu Hello. Dalam Java semua kode seharusnya ditempatkan didalam deklarasi class. kita melakukannya dengan menggunakan kata kunci class.

Baris berikutnya yaitu yang terdiri atas kurung kurawal { menandakan awal blok. kita dapat juga meletakkan kurung kurawal ini setelah baris pertama dari kode yang kita tulis. Jadi, kita dapat menulis kode kita sebagai berikut :

```
public class Hello
{ }
atau
public class Hello { }
```

Tiga baris selanjutnya menandakan adanya komentar Java. Komentar adalah sesuatu yang digunakan untuk mendokumentasikan setiap bagian dari kode yang ditulis. Komentar bukan merupakan bagian dari program itu sendiri, tetapi digunakan untuk tujuan dokumentasi. Komentar itu sendiri dapat ditambahkan pada kode yang Anda tulis sebagai petunjuk yang dapat membantu proses pembelajaran pemrograman yang baik.

```
/**
 * My first java program
 */
```

Komentar dinyatakan dengan tanda “/” dan “*/”. Segala sesuatu yang ada diantara tanda tersebut diabaikan oleh compiler Java, dan mereka hanya dianggap sebagai komentar sedangkan untuk membuat sebuah atau satu baris komentar dengan cara berikut ini

// ini adalah program pertamaku. Semua tulisan yang berada satu baris dengan tan da // akan diabaikan oleh compiler.

Baris selanjutnya,

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

atau dapat juga ditulis sebagai berikut,

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

mengindikasikan nama suatu method dalam class Hello yang bertindak sebagai method utama. Method utama adalah titik awal dari suatu program Java. Semua program kecuali applet yang ditulis dalam bahasa Java dimulai dengan method utama. Yakinkan untuk mengikuti kaidah penulisan tanda yang benar.

MODUL 2

TIPE DATA, KEYWORD, VARIABEL, OPERATOR DAN TEKNIK KONFERSI

A. Target Praktikum

- Mahasiswa mengetahui berbagai macam tipe data dan penggunaannya
- Mahasiswa mengetahui berbagai macam keyword
- Mahasiswa mengetahui cara pembuatan variable dan penggunaannya.
- Mahasiswa mengetahui konfersi tipe data
- Mahasiswa mampu menyelesaikan permasalahan terkait penggunaan variable dan tipe data
- Mahasiswa mengetahui berbagai macam operator java dan penggunaannya
- Mahasiswa mampu menyelesaikan permasalahan terkait penggunaan operator

B. Materi Dasar

1. Keyword dalam Java

Kata kunci adalah identifier yang telah dipesan untuk didefinisikan sebelumnya oleh Java untuk tujuan tertentu. Anda tidak dapat menggunakan keyword sebagai nama variabel, class, method.

Berikut Ini Keyword Java :

abstract	default	if	private	throws
boolean	do	import	public	try
break	double	int	return	void
byte	else	static	short	while
case	extends	long	super	const
catch	final	native	switch	for
char	finally	new	this	continue
class	float	package	throw	transient

2. Tipe Data Primitif

Bahasa pemrograman Java mendefinisikan delapan tipe data primitif. Mereka diantaranya adalah boolean (untuk bentuk logika), char (untuk bentuk tekstual), byte, short, int, long (integral), double and float (floating point).

3. Variabel

Variabel adalah item yang digunakan data untuk menyimpan pernyataan objek. Variabel memiliki **tipe data** dan **nama**. Tipe data menandakan tipe nilai yang dapat dibentuk oleh variabel itu sendiri. Nama variabel harus mengikuti aturan untuk identifier.

Berikut Aturan penamaan variable :

- Diawali dengan
 - huruf/abjad
 - karakter mata uang
 - underscore (_)
- Terdiri dari huruf/abjad, angka dan underscore
- Tidak boleh mengandung karakter khusus atau spasi
- Tidak boleh diawali dengan angka

3.1 Deklarasi dan Inisialisasi Variabel

Untuk deklarasi variabel adalah sebagai berikut,

<data tipe> <name> [=initial value];

Nilainya berada diantara <> adalah nilai yang disyaratkan, sementara nilai dalam tanda [] bersifat optional sesuai dengan tipe data yang dipakai untuk deklarasi variabel.

3.2 Menampilkan Data Variabel

Untuk mengeluarkan nilai dari variabel yang diinginkan, kita dapat menggunakan perintah sebagai berikut, System.out.println() atau System.out.print()

Contoh variabel :

```
a. public class ContohVariabel1
{
    public static void main(String args[])
    {
        int a=0; // local variable
        boolean b=true; // local variable
        float c=23; // local variable
        System.out.println(a);
        System.out.println(b);
        System.out.println(c);
    }
}
```

```

b. public class ContohVariabel2
{
    static int a; // member variable
    static boolean b; // member variable
    static float c; // member variable
    public static void main(String args[])
    {
        System.out.println(a);
        System.out.println(b);
        System.out.println(c);
    }
}

```

4. Teknik Konfersi Data

Konfersi adalah cara untuk memberikan nilai suatu data dari satu tipe data ke tipe data yang lain.

Contoh, perhatikan class berikut :

```

class konversi
{
    public static void main(String[]args)
    {
        int a=50;
        double b=40.5;
        String konfersi_a =String.valueOf(a); // konfersi dari integer ke string
        String konfersi_b =String.valueOf(b); //konfersi dari double ke string
        int pindah_keinteger =Integer.parseInt(konfersi_a); // konfersi dari string ke integer
        double pindah_kedouble =Double.parseDouble(konfersi_b); // konfersi dari string ke double
    }
}

```

Yang perlu diketahui, bahwa konfersi data dilakukan jika kita membutuhkan suatu tipe data yang sama untuk menjalankan sebuah proses, jika data tersebut sudah sama tipe datanya maka tidak perlu dilakukan proses konfersi data.

Untuk konfersi data harus memenuhi dua criteria berikut ini :

1. Dua tipe data yang kompatibel

2. tipe data awal lebih kecil dibandingkan dengan tipe data tujuan.

Jika dua syarat tersebut tidak terpenuhi maka bisa dilakukan teknik casting, yaitu melakukan perubahan tipe data dengan cara dipaksa. Casting ini bisa langsung dilakukan otomatis oleh java dengan cara menambahkan tipe data casting dikurung kemudian dilanjutkan dengan proses yang mau dijalankan, contoh : `int a=4; int b=5; double c=(double)a/b;`

5. Operator

Dalam Java, ada beberapa tipe operator. Ada operator **aritmatika**, operator **relasi**, operator **logika**, dan operator **kondisi**. 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.

5.1 Operator Aritmatika

Operator aritmatika merupakan operator yang berhubungan dengan fungsi matematika.

contoh	Nama	Hasil
<code>a + b</code>	Penjumlahan	Jumlah dari \$a dan \$b
<code>a - b</code>	Pengurangan	Selisih dari \$a dan \$b
<code>a * b</code>	Perkalian	Hasil kali dari \$a dan \$b
<code>a / b</code>	Pembagian	Pembagian dari \$a dan \$b
<code>a % b</code>	Modulus	Sisa dari \$a dibagi dengan \$b

Penulisan Ekspresi Aritmatika

$$\frac{a+b}{a-b} - pq = (a+b)/(a-b)-p*q$$

$$\frac{a+b/p}{ab} - q = (a+b/p)/(a*b)-q$$

Contoh Operator Aritmatika :

```
public class OperasiAritmatika
{
    public static void main(String args[])
    {
        short x = 6;
        int y = 4;
        float a = 12.5f;
```



```

float b = 7f;
System.out.println("x= "+ x + ", y = "+y);
System.out.println("x + y =" + (x + y));
System.out.println("x - y =" + (x - y));
System.out.println("x / y =" + (x / y));
System.out.println("x % y =" + (x % y));
System.out.println("a = "+ a + ", b = "+b);
System.out.println("a / b =" + (a / b));
System.out.println("x / b =" + (x / b));
System.out.println("b / x =" + (b / x));
}
}

```

5.2 Operator Relasi

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

contoh	Nama	Hasil
a == b	sama dengan	benar jikaa sama dengan b
a != b	tidak sama	benar jika a tidak sama dengan b
a < b	lebih kecil	benar jika a lebih kecil b
a > b	lebih besar	benar jika a lebih besar b
a <= b	lebih kecil atau sama dengan	benar jika a lebih kecil atau sama dengan b
a >= b	lebih besar atau sama dengan	benar jikaa lebih besar atau sama dengan b

Contoh Operator Relasi :

```

public class operator {
    public static void main(String args[])
    {
        int a = 4;
        int b = 3;
        System.out.println(a==b);
        System.out.println(a!=b);
        System.out.println(a<b);
        System.out.println(a>b);
    }
}

```

```

    System.out.println(a<=b);
    System.out.println(a>=b);
}
}

```

5.3 Operator Tambahan dan Pengurangan

Tabel operator Increment/decrement adalah :

Operator	Operasi	Contoh
++a	Pre-increment	Nilai a ditambah satu, kemudian operasi berjalan
a++	Post-increment	Operasi dilaksanakan, kemudian a ditambah satu
--a	Pre-decrement	Nilai a dikurang satu, kemudian operasi dilaksanakan satu
a--	Post-decrement	Operasi dilaksanakan, kemudian a dikurangi satu

Contoh Operator Tambahan dan Pengurangan :

```

public class IncrementDecrement
{
    public static void main(String args[])
    {
        int i = 1;
        System.out.println("i : " + i );
        // Pre-increment
        System.out.println("++i : " + ++i );
        // Post-increment
        System.out.println("i++ : " + i++ );
        System.out.println("i : " + i );
        // Pre-decrement
        System.out.println("--i : " + --i );
        // Post-decrement
        System.out.println("i-- : " + i-- );
        System.out.println("i : " + i );
    }
}

```

5.4 Operator Logika

Operator logika memiliki satu atau lebih operand boolean yang menghasilkan nilai boolean. Terdapat enam operator logika yaitu: && (logika AND), & (boolean logika AND), || (logika OR), | (boolean logika inclusive OR), ^ (boolean logika exclusive OR), dan ! (logika NOT).

contoh	Nama	Hasil
a &&b	AND	benar jika a dan b benar
a b	OR	benar jika a atau b benar
a ^b	XOR	benar jika a atau b benar tetapi tidak keduanya

Contoh Operator Logika:

```
public class operator {  
    public static void main(String args[])  
    {  
        System.out.println(a&&b);  
        System.out.println(a||b);  
        System.out.println(a^b);  
        System.out.println(a&b);  
        System.out.println(a|b);  
        System.out.println(a!=b);  
    }  
}
```

Operator Perbandingan/Kondisi

Operator perbandingan ini digunakan pada struktur kontrol program, seperti if, elseif, di mana dilakukan perbandingan antar dua nilai.

contoh	Nama	Hasil
a == b	sama dengan	benar jika a sama dengan b
a != b	tidak sama	benar jika a tidak sama dengan b
a < b	lebih kecil	benar jika a lebih kecil b
a > b	lebih besar	benar jika a lebih besar b
a <= b	lebih kecil atau sama dengan	benar jika a lebih kecil atau sama dengan b
a >= b	lebih besar atau sama dengan	benar jika a lebih besar atau sama dengan b

MODUL 3

INPUT DARI KEYBOARD DAN STRUKTUR KONTROL

A. Target Praktikum

- Membuat program Java yang interaktif yang bisa mendapatkan input dari keyboard
- Menggunakan class `BufferedReader` untuk mendapatkan input dari keyboard melalui layar console
- Menggunakan class `JOptionPane` untuk mendapatkan input dari keyboard menggunakan GUI
- Menggunakan struktur kontrol pemilihan (`if`, `else`, `switch`) yang digunakan untuk memilih blok kode yang akan dieksekusi

B. Materi Dasar

Setelah kita mempelajari konsep dasar pada Java dan menulis beberapa program sederhana. Sekarang kita akan mencoba membuat program kita lebih interaktif dengan menggunakan input dari keyboard. Pada bab ini, kita akan mempelajari dua cara memberikan input, yang pertama adalah menggunakan class `BufferedReader` dan melalui GUI (Graphical User Interface) dengan menggunakan class `JOptionPane`

1. Menggunakan `BufferedReader`

Pada bagian ini, kita akan menggunakan class `BufferedReader` yang berada di package `java.io` untuk mendapatkan input dari keyboard.

Berikut ini adalah langkah-langkah yang diperlukan untuk mendapatkan input dari keyboard:

1. Tambahkan library tambahan di bagian paling atas code Anda:

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

2. Tambahkan statement berikut:

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

3. Deklarasikan variabel `String` temporary untuk mendapatkan input, dan gunakan fungsi `readLine()` untuk mendapatkan input dari keyboard. Anda harus mengetikkannya di dalam blok `try-catch`:

```
try {  
    String temp = dataIn.readLine(); }  
catch( IOException e ){  
    System.out.println("Error in getting input"); }
```

Berikut ini adalah contoh source code lengkapnya:

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

```

import java.io.InputStreamReader;
import java.io.IOException;
public class GetInputFromKeyboard
{
    public static void main( String[] args ){
        BufferedReader dataIn = new BufferedReader(new InputStreamReader( System.in) );
        String name = "";
        System.out.print("Silahkan Masukkan Nama Anda:");
        try{
            name = dataIn.readLine();
        }catch( IOException e ){
            System.out.println("Error!");
        }
        System.out.println("Hallo " + name +"Anda Berhasil Membuatnya!!");
    }
}

```

Berikutnya akan penjelasan setiap baris dari code tersebut:

- Statement,

```

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.

- blok di bawah ini merupakan try-catch block,

```

try{
    name = dataIn.readLine();
}catch( IOException e ){
    System.out.println("Error!");
}

```

Pada baris ini menjelaskan bahwa kemungkinan terjadi error pada pernyataan,

```

name = dataIn.readLine();

```

akan ditangkap. Kita akan membahas tentang penanganan exception pada bab selanjutnya dari pembahasan ini, tetapi untuk sekarang, Anda cukup mencatat bahwa Anda perlu menambahkan kode ini untuk menggunakan method `readLine()` dari `BufferedReader` untuk mendapatkan input dari user.

- pernyataan,

```
name = dataIn.readLine();
```

method diatas memanggil `dataIn.readLine()`, mendapatkan input dari user dan memberikan sebuah nilai `String`. Nilai ini akan disimpan ke dalam variabel `name`, yang akan kita gunakan pada statement akhir untuk menyambut user,

```
System.out.println("Hello " + name + "!");
```

2. JOptionPane

Cara selanjutnya untuk mendapatkan input dari user adalah dengan menggunakan class `JOptionPane` yang didapatkan dari `javax.swing` package. `JOptionPane` memudahkan memunculkan dialog box standard yang memberikan kepada user sebuah nilai atau menginformasikan sesuatu.

Berikut Contoh Kode yang digunakan :

```
import javax.swing.JOptionPane;
public class GetInputFromKeyboard2 {
    public static void main( String[] args )
    {
        String name = "";
        name = JOptionPane.showInputDialog("Please enter your name");
        String msg = "Hello " + name + "!";
        JOptionPane.showMessageDialog(null, msg);
    }
}
```

Berikut penjelasannya

- Statement pertama,

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

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

Bisa juga ditulis seperti,

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

- Pernyataan,

```
name = JOptionPane.showInputDialog("Please enter your name");
```

membuat sebuah input dialog JOptionPane, yang akan menampilkan dialog dengan sebuah pesan, sebuah textfield dan tombol OK seperti pada gambar. Hasil dari dialog tersebut adalah String dan disimpan ke dalam variabel name. Sekarang kita membuat pesan selamat datang, yang akan disimpan ke dalam variabel msg,

```
String msg = "Hello " + name + "!";
```

Baris selanjutnya adalah menampilkan sebuah dialog yang berisi sebuah pesan dan tombol OK,

```
JOptionPane.showMessageDialog(null, msg);
```

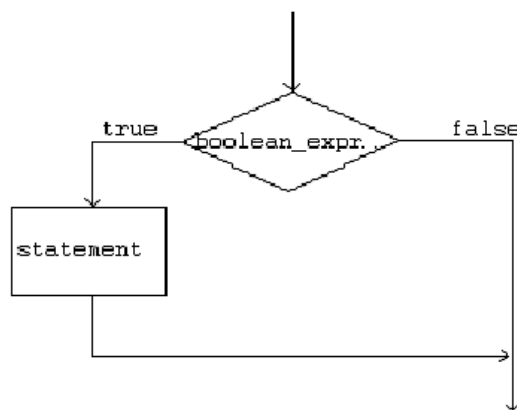
3. Struktur Kontrol

Pada bagian ini, kita akan mempelajari tentang struktur kontrol dimana kita dapat mengubah cara eksekusi pada pernyataan yang dibuat di program kita.

Struktur kontrol adalah pernyataan dari Java yang memungkinkan user untuk memilih dan mengeksekusi blok kode spesifik dan mengabaikan blok kode yang lain.

3.1 Statement if

Pernyataan if akan menentukan sebuah pernyataan (atau blok kode) yang akan eksekusi jika dan hanya jika persyaratan bernilai benar(true).



Gambar 1: Flowchart Statement If

Contoh Program Dengan Statement If :

```
public class Logika_if {  
    public static void main (String args[]){  
        int targetnilai=60;  
        boolean lulus=false;  
        int nilaianda=70;
```

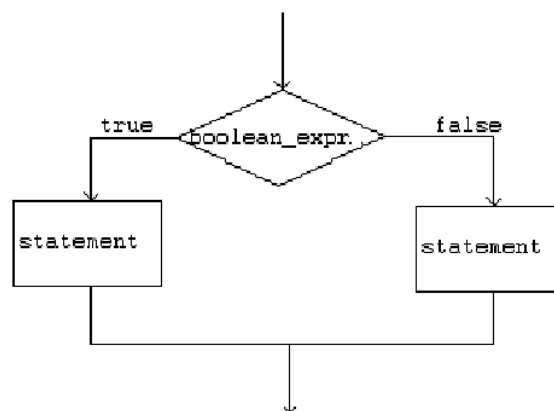
```

        if (nilaianda > targetnilai){
            lulus = true;
        }
        System.out.println("Status kelulusan :"+lulus);
    }
}

```

3.2 Statement if-else

Pernyataan if-else digunakan apabila kita ingin mengeksekusi beberapa pernyataan dengan kondisi true dan pernyataan yang lain dengan kondisi false.



Gambar 2: Flowchart Statement If-Else

Contoh Program Dengan Statement If Else:

```

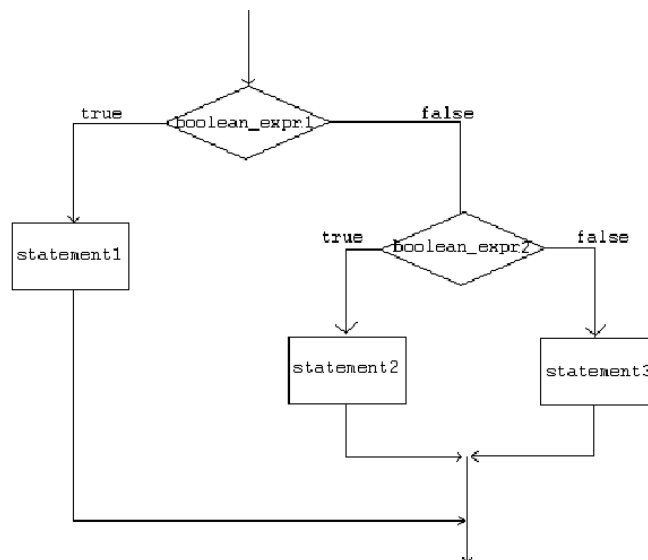
public class Statement_If_Else {
    public static void main (String args[]) {
        int targetnilai=60;
        String hadiah="";
        int nilaianda=50;
        if (nilaianda > targetnilai){
            hadiah = "ponsel";
        } else{
            hadiah = "radio FM";
        }
        System.out.println("Jenis hadiah :"+hadiah);
    }
}

```


3.3 Statement if-else-if

Pernyataan pada bagian kondisi else dari blok if-else dapat menjadi struktur if-else yang lain. Kondisi struktur seperti ini memungkinkan kita untuk membuat seleksi persyaratan yang lebih kompleks.

Bentuk statement if-else if,
`if( boolean_expression1 )`
`statement1;`
`else if( boolean_expression2 )`
`statement2;`
`else statement3`



Gambar 3: Flowchart Statement If-Else-If

Contoh Program Dengan Statement If Else If :

```
import javax.swing.*;

public class Statement_If_Else_if {
    public static void main(String args[]) {
        String input=JOptionPane.showInputDialog("Nilai anda ?");
        int nilai=Integer.parseInt(input); String grade=" ";
        if(nilai>85){
            grade="A";
        }else if(nilai>75){
            grade="B";
        }else{
            grade="E";
        }
    }
}
```

```

    }
    System.out.println("Nilai Anda : "+nilai);
    System.out.println("Grade Anda : "+grade);
    System.exit(0);
}
}

```

3.4 Statement switch

Cara lain untuk membuat cabang adalah dengan menggunakan kata kunci switch. Switch mengkonstruksikan cabang untuk beberapa kondisi dari nilai.

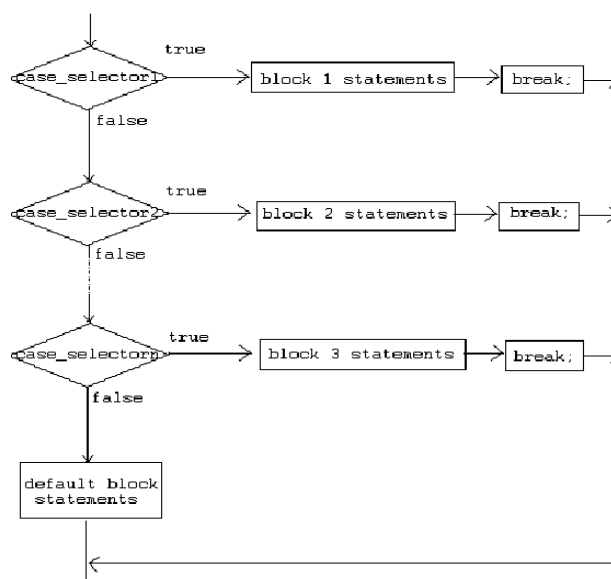
Bentuk statement switch,

```

switch( switch_expression )
{
    case case_selector1: statement1;
    break;
    case case_selector2: statement2;
    break;
    default: statement3;
    break;
}

```

switch_expression adalah ekspresi integer atau karakter dan case_selector1, case_selector2 dan seterusnya adalah konstanta unik dari nilai integer atau karakter.



Contoh Program Dengan Ststatement Switch Case:

```
import javax.swing.*;
public class AlurSwitch {
    public static void main(String args[]) {
        String pilihan=JOptionPane.showInputDialog("Menu :\n1. Sate Kambing\n2.
            Gule\n3. Sop Kaki Kambing\n4. Tongseng");
        int menu=Integer.parseInt(pilihan);
        switch(menu)
        {
            case 1: System.out.println("pilih sate kambing");
                break;
            case 2: System.out.println("pilih gule");
                break;
            case 3: System.out.println("pilih sop kaki kambing");
                break;
            case 4: System.out.println("pilih tongseng");
                break;
            default: System.out.println("Maaf, menu tidak ada");
                break;
        } System.exit(0);
    }
}
```

MODUL 4

PERULANGAN/LOOPING

A. Target Praktikum

- Mahasiswa mengetahui struktur dasar statement perulangan(looping)
- Mahasiswa mengetahui cara pengoperasian statement perulangan
- Mahasiswa mampu menyelesaikan permasalahan terkait penggunaan statement perulangan

B. Materi Dasar

pengulangan adalah berupa pernyataan dari Java yang memungkinkan kita untuk mengeksekusi blok code berulang-ulang sesuai dengan jumlah tertentu yang diinginkan. Ada tiga macam jenis dari struktur pengulangan yaitu while, do-while, dan for-loops.

1. while loop

Pernyataan while loop adalah pernyataan atau blok pernyataan yang diulang-ulang sampai mencapai kondisi yang cocok.

Bentuk pernyataan while,

```
while( boolean_expression )  
{  
    statement1;  
    statement2;  
}
```

Pernyataan di dalam while loop akan dieksekusi berulang-ulang selama kondisi boolean_expression bernilai benar (true). Contoh, pada kode dibawah ini,

```
public class contoh_while {  
    public static void main(String args[]) {  
        int i = 4;  
        while( i > 0 ){  
            System.out.print(i);  
            i--;  
        }  
    }  
}
```

Contoh diatas akan mencetak angka 4321 pada layar.

2. do-while loop

Do-while loop mirip dengan while-loop. Pernyataan di dalam do-while loop akan dieksekusi 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.

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{  
    statement1;  
    statement2;  
}while( boolean_expression );
```

Pernyataan di dalam do-while loop akan dieksekusi pertama kali, dan akan dievaluasi kondisi dari boolean_expression. Jika nilai pada boolean_expression tersebut bernilai true, pernyataan di dalam do-while loop akan dieksekusi lagi.

Berikut ini beberapa contoh do-while loop:

Contoh 1:

```
public class contoh_while {  
    public static void main(String args[]) {  
        int i = 4;  
        do{  
            System.out.print(i);  
            i--;  
        }while( i > 0 );  
    }  
}
```

Contoh ini akan memberikan output 4321 pada layar.

3. for loop

Pernyataan for loop memiliki kondisi hampir mirip seperti struktur pengulangan sebelumnya yaitu melakukan pengulangan untuk mengeksekusi kode yang sama sebanyak jumlah yang telah ditentukan.

Bentuk dari for loop,

```
for (InitializationExpression; LoopCondition; StepExpression){  
    statement1;
```

```
statement2;
```

```
}
```

- **InitializationExpression** – inisialisasi dari variabel loop.
- **LoopCondition** - membandingkan variabel loop pada nilai batas tertentu
- **StepExpression** - melakukan update pada variabel loop.

Berikut ini adalah contoh dari for loop,

```
public class cthfor {  
    public static void main(String args[]){  
        for(int i=4;i>=0;i--)  
        {  
            System.out.print(i);  
        }  
    }  
}
```

Pada contoh ini, pernyataan `i=4` merupakan inisialisasi dari variabel. Selanjutnya, kondisi `i>=0` diperiksa. Jika kondisi bernilai `true`, pernyataan di dalam for loop dieksekusi. Kemudian, ekspresi `i--` dieksekusi, lalu akan kembali pada bagian pemeriksaan terhadap kondisi `i>=0` lagi. Kondisi ini akan dilakukan berulang-ulang sampai mencapai nilai yang salah (`false`).

4. Percabangan dalam Perulangan

Pernyataan percabangan memungkinkan kita untuk mengatur aliran eksekusi program. Java memberikan tiga bentuk pernyataan percabangan: `break`, `continue` dan `return`.

4.2 Pernyataan break

Pernyataan `break` memiliki dua bentuk: tidak berlabel (`unlabeled`) dan berlabel (`labeled`).

4.2.1 Pernyataan break tidak berlabel

Pernyataan `break` tidak berlabel (`unlabeled`) digunakan untuk menghentikan jalannya pernyataan `switch`. Selain itu pernyataan `break unlabeled` juga bisa digunakan untuk menghentikan pernyataan-pernyataan `for`, `while` atau `do-while` loop.

```
for( int i=0; i< 5; i++ ){  
    if( i==3){  
        System.out.println(i);  
        break;  
    }  
}
```

Dari pernyataan diatas didapatka jika i berposisi pada nilai maka perulangan akan dihentikan. Dan akan melakukan pencetakan angka i, yaitu angka 3.

4.2.2 Pernyataan break berlabel

Bentuk label dari pernyataan break akan menghentikan pernyataan di luarnya, dimana sebelumnya harus diberikan label yang sudah di spesifikasikan pada program pada pernyataan break. Ketika sebuah nilai ditemukan, break akan menghentikan pernyataan yang diberi label searchLabel yang terletak di luar pernyataan for loop.

```
searchLabel:
for( int i=0; i<6; i++ )
{
    if(i==5 )
    {
        System.out.println(i);
        break searchLabel;
    }
}
```

Pernyataan break menghentikan pernyataan yang diberi label; dan tidak menjalankan aliran kontrol apapun pada label. Aliran kontrol pada label akan diberikan secara otomatis pada pernyataan yang terletak dibawah label. Jadi jika pernyataan yang tidak dibawahnya maka tidak akan mengenali label yang telah dituliskan.

4.3 Pernyataan Continue

Pernyataan continue memiliki dua bentuk: berlabel dan tidak berlabel. Anda dapat menggunakan pernyataan continue untuk melanjutkan pengulangan yang sedang dijalankan oleh pernyataan for, while, atau do- while loop.

4.3.1 Pernyataan continue tidak berlabel

Bentuk pernyataan continue tidak berlabel (unlabeled) akan melewati bagian pernyataan setelah pernyataan ini dituliskan dan memeriksa eksepresi logika (boolean) yang mengontrol pengulangan. Jika ekspresi logika (boolean) masih bernilai true, maka pengulangan tetap dilanjutkan. Pada dasarnya pernyataan ini akan melanjutkan bagian pengulangan pada pernyataan loop.

```
for( int i=0; i<6; i++ ){
    if(i==3 ){
```

```

        continue;
    }
    System.out.println(i);
}

```

4.3.2 Pernyataan continue berlabel

Bentuk pernyataan continue berlabel (labeled) akan melanjutkan pengulangan yang sedang terjadi dan dilanjutkan ke pengulangan berikutnya dari pernyataan pengulangan yang diberi label (tanda).

```

search:
for( int i=0; i<6; i++ ) {
    if(i==3){
        continue search;
    }
    System.out.println(i);
}

```

4.4 Pernyataan Return

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

Untuk memberikan sebuah 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;`

4.5 Nested Loop

Nested Loop adalah suatu bentuk perulangan didalam perulangan, Perhatikan contoh berikut ini untuk memahaminya :

```

public class nested{
    public static void main(String[]args){
        for(int i=0;i<4;i++){
            for(int j=0;j<5;j++){
                System.out.print(j);
            }
            System.out.println();
        }
    }
}

```