Bahasa Pemrograman

(Pemrograman Visual dengan Java)

Modul Praktikum

# Topik

*Structured Control* (Kendali terstruktur).

# Tujuan

Setelah menyelesaikan praktikum ini, mahasiswa diharapkan mampu:

1. Membuat program yang memerlukan penggunaan *selection control* seperti; *if statement* dan *switch statement*.
2. Membuat program yang memerlukan penggunaan *iteration control* seperti; *for statement, while statement*.
3. Membuat program yang memerlukan penggunaan *break statement* dan *continue statement*.
4. Membuat program yang dapat mengolah suatu kalimat atau *string*.

# Alat dan Bahan

* Komputer dengan **Java Development Kit (JDK)** dan aplikasi BlueJ sudah terinstall
* Informasi download dan instalasi JDK dan Netbeans dapat diperoleh di; <http://www.oracle.com/us/technologies/java/jdk-7-netbeans-download-432126.html>

# Rangkuman/Teori

## [Menjelaskan tentang *conditional statement*]

*Conditional statements* biasa digunakan untuk menguji suatu kondisi, kondisi dari statement tersebut biasanya untuk menentukan arah dari flow suatu algoritma.

*Conditional statements* sering digunakan dalam:

* Percabangan (selection/decision)
* Perulangan (repetition/looping)

Terdapat dua operator yang biasanya digunakan dalam *conditional staments*:

* *Relational operator*
* *Logical operator*

Berikut adalah tabel yang berisi daftar *relational operator* dan tabel yang berisi daftar *logical operator* yang digunakan dalam pemrograman java.

Tabel 1. Relational Operator

|  |  |
| --- | --- |
| NamaOperator | Operasi |
| **==** | Equality |
| **!=** | Inequality |
| **<** | Less than |
| **>** | Greater than |
| **<=** | Less than or equal |
| **>=** | Greater than or equal |

Tabel 2. Logical Operator

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **x** | **y** | **AND** | **OR** | **NOT** |
| **x && y** | **x || y** | **!x** |
| false | false | false | false | true |
| false | true | false | true | true |
| True | false | false | true | false |
| true | True | true | true | false |

## [Menjelaskan tentang *if statement*]

Salahsatu instruksi yang dapat digunakan untuk mengendalikan alur program adalah *if statement*. Dengan *if statement* kita bisa mengarahkan alur program sesuai kondisi (dari hasil *conditional* statement) yang ditentukan pada *if statement*.

*Syntax if statement*:

**if(** *<conditional statement>* **) {**

*<kumpulan instruksi jika kondisi terpenuhi>*

**}**

**else {**

*<kumpulan instruksi jika kondisi tidak terpenuhi>*

**}**

atau;

**if(** *<conditional statement>* **) {**

*<kumpulan instruksi jika kondisi terpenuhi>*

**}**

## [Menjelaskan tentang *switch statement*]

Salahsatu instruksi yang dapat digunakan untuk mengendalikan alur program adalah *switch statement*. Dengan *switch statement* kita bisa mengarahkan alur sesuai dengan kecocokan antara nilai dari variable uji dengan nilai uji.

*Syntax switch statement*:

**switch(** *<variable uji>* **) {**

**case** *<nilai uji\_1>: <kumpulan instruksi jika nilai variable cocok dengan nilai uji 1 terpenuhi>;*

 **break;**

**case** *<nilai uji\_2>: <kumpulan instruksi jika nilai variable cocok dengan nilai uji 2 terpenuhi>;*

 **break;**

**case** *<nilai uji\_3>: <kumpulan instruksi jika nilai variable cocok dengan nilai uji 1 terpenuhi>;*

 **break;**

dst….

**default:** *<kumpulan instruksi jika nilai variable tidak ada yang cocok dengan nilai ujii>;*

**}**

## [Menjelaskan tentang *for statement*]

Salahsatu instruksi yang dapat digunakan untuk mengendalikan alur program yang berupa perulangan adalah *for statement*. Dengan *for statement* kita bisa mengarahkan alur agar berulang sesuai dengan jumlah yang ditentukan.

*Syntax for statement*:

**for(***<inisialisasi>; <batas akhir>;**<multiplikasi>* **) {**

*<kumpulan instruksi yang akan diulang>*

**}**

Contoh:

for(int i=1; i<=5; i++) {

System.out.print(“Perulangan ke – “);

System.out.println(i);

}

## [Menjelaskan tentang *while statement*]

Salahsatu instruksi yang dapat digunakan untuk mengendalikan alur program yang berupa perulangan adalah *while statement*. Dengan *while statement* kita bisa mengarahkan alur agar berulang hingga *conditional statement* pada *while statement* tersebut terpenuhi.

*Syntax while statement*:

**while(***<conditional statement>* **) {**

*<kumpulan instruksi yang akan diulang>*

**}**

atau;

**do {**

*<kumpulan instruksi yang akan diulang>*

**} while(***<conditional statement>* **);**

Contoh:

x=0;

while(x<6) {

System.out.print(“Perulangan ke – “);

System.out.println(x);

x++;

}

## [Menjelaskan tentang *break dan continue statement*]

***break*** merupakan instruksi yang bisa digunakan untuk keluar dari suatu perulangan walaupun kondisi atau jumlah perulangan belum selesai.

***continue*** merupakan instruksi yang bisa digunakan untuk mem-bypass beberapa instruksi didalam perulangan.

## [Menjelaskan tentang tipe data *String*]

*String* dalam pemrograman Java adalah suatu class, dan penanganannya tidak sama dengan type data primitive (spt; int, char, double, boolean,..dsb).

Variable dengan tipe data String pada pemrograman Java untuk selanjutnya akan disebut sebagai object. Berikut adalah deklarasi untuk membuat object String;

**String *<nama\_object>*;**

Contoh:

**String x;** 🡨 perintah ini akan membuat object dengan nama x, dengan tipe String.

**String pesan = “Hallo”;** 🡨 perintah ini akan membuat object dengan nama pesan, dengan tipe String, dan mengisikan tulisan Hallo pada object pesan.

### Method pada *String*

Method yang dapat digunakan oleh object tipe String diantaranya adalah:

* **length()**

Untuk mengetahui panjang karakter dari suatu string.

Contoh:

String kalimat = “ABCDEFGHI”;

int panjang = kalimat.length();

* **charAt(***int* *n***)**

Untuk mengetahui karakter yang tersimpan pada posisi ke-n dalam suatu String.

Contoh:

String kalimat = “ABCDEFGHI”;

char huruf = kalimat.charAt(4);

* **substring(***int b***)**

Untuk mengetahui potongan string yang tersimpan pada suatu String mulai dari posisi ke-b sampai posisi akhir String.

* **substring(***int b, int e***)**

Untuk mengetahui potongan string yang tersimpan pada suatu String mulai dari posisi ke-b, sampai posisi ke-e.

Contoh:

String kalimat = “ABCDEFGHI”;

String potongan1 = kalimat.substring(4);

String potongan2 = kalimat.substring(4, 6);

* **lastIndexOf(***<karakterYangDicari>***)**

Untuk mengetahui posisi index terakhir dari suatu karakter yang dicari pada suatu String.

Contoh:

String kalimat = “AB#CDE#FGHI”;

int idx = kalimat.lastIndexOf(‘#’);

* **indexOf(***<karakterYangDicari>, <indexAwalPencarian>***)**

Untuk mengetahui posisi index dari suatu karakter yang dicari pada suatu String, pencarian dimulai pada posisi index seperti yang dituliskan pada *<indexAwalPencarian>*.

Contoh:

String kalimat = “AB#CDE#FGHI”;

int idx = kalimat.indexOf(‘#’, 3);

* **toCharArray()**

Untuk mengubah object string menjadi array karakter. Biasa digunakan untuk menyalin seluruh karakter pada object String ke object array.

Contoh:

String kalimat = “AB#CDE#FGHI”;

Char [] arrayKarakter = kalimat.toCharArray();

### Konversi pada String

Terdapat beberapa cara konversi angka menjadi string:

* **typeAsal.toString(***angka***)**

Konversi angka yang memiliki type seperti yang disebutkan pada typeAsal menjadi string.

Contoh:

String tipeDouble = Double.toString(14.2); //hasilnya adalah “14.2”

String tipeFloat = Float.toString(1420000000); //hasilnya adalah “1.42E9”

int nilai = 3;

System.out.println(Integer.toString(nilai\*4); //hasilnya adalah “12”

* **Integer.toString(***angka, format***)**

konversi dari integer menjadi string dengan format bilangan yang ditentukan

format: **2** (untuk bilangan biner), **8** (untuk bilangan oktal), **10** (untuk bilangan desimal), **16** (untuk bilangan heksadesimal).

Contoh:

int nilai = 3;

System.out.println(Integer.toString(nilai\*4, 2); //hasilnya adalah “1100”

System.out.println(Integer.toString(nilai\*4, 16); //hasilnya adalah “C”

Cara konversi dari String menjadi angka:

* **Integer.parseInt(***string***)**;

Melakukan konversi dari string (string suatu angka) menjadi angka tipe integer.

**Double.parseDouble(***string***)**;

Melakukan konversi dari string (string suatu angka) menjadi angka tipe double.

Contoh:

String diameter = "22";

String pi = "3.14";

Double kelilingLingkaran **= Integer.parseInt**(diameter) \* **Double.parseDouble**(pi);

System.out.println(kelilingLingkaran);

# Praktek

Pada bagian ini, akan dibahas mengenai praktek yang akan dilakukan. Untuk mempermudah pencatatan hasil praktek dan penarikan kesimpulan terhadap hasil praktek tersebut, Anda dapat menggunakan form pada **Appendix 1.**

## Praktek 1: Menggunakan *if statement*

1. Buat class dengan nama **Lab\_2\_if**.
2. Tambahkan kode program agar program:
	* 1. Membaca *input user*
		2. Mencocokkan apakah *input user* adalah kurang dari 10.
		3. Jika *input user* kurang dari 10 maka akan tampil dialog yang berisi pesan: “Input anda kurang dari 10”
		4. Jika *input user* lebih besar atau sama dengan10 maka akan tampil dialog yang berisi pesan: “Input anda sangat banyak”
3. Jalankan class yang sudah dimodifikasi tersebut, dengan cara cara **klik kanan pada nama file** 🡪 **Run File** (atau dengan menekan tombol ***Shift + F6*** pada keyboard).
4. Jika ada kesalahan, lakukan perbaikan yang diperlukan.

## Praktek 2: Menggunakan *switch statement*

1. Buat class dengan nama **Lab\_2\_switch**.
2. Tambahkan kode program agar program:
	* 1. Membaca *input user* dengan menggunakan dialog input (JoptionPane), dengan tulisan ***masukkan jenis membership anda:***
		2. Mencocokkan apakah *input user*.
		3. Jika *input user* adalah “GOLD” , maka akan tampil dialog yang berisi pesan; “Terima kasih atas *input* anda sebagai member” , kemudian menampilkan dialog lagi yang berisi pesan; “Karena membership anda adalah GOLD, anda mendapatkan poin *voucher* sebesar 50 ribu rupiah” .
		4. Jika *input user* adalah “SILVER” , maka akan tampil dialog yang berisi pesan; “Terima kasih atas *input* anda sebagai member” , kemudian menampilkan dialog lagi yang berisi pesan; “Karena membership anda adalah SILVER, anda mendapatkan poin *voucher* sebesar 30 ribu rupiah” .
		5. Jika *input user* adalah “BRONZE” , maka akan tampil dialog yang berisi pesan; “Terima kasih atas *input* anda sebagai member” , kemudian menampilkan dialog lagi yang berisi pesan; “Karena membership anda adalah BRONZE, anda mendapatkan poin *voucher* sebesar 10 ribu rupiah” .
		6. Jika *input user* adalah bukan dari jenis membership yang ada (GOLD, SILVER, , BRONZE) maka akan tampil dialog yang berisi pesan; “Terima kasih atas *input* anda sebagai konsumen” , kemudian menampilkan dialog lagi yang berisi pesan; “Meskipun andan bukan member, anda berhak mendapat potongan sebesar 5000 rupiah dari total belanja” .
3. Jalankan class yang sudah dimodifikasi tersebut, dengan cara cara **klik kanan pada nama file** 🡪 **Run File** (atau dengan menekan tombol ***Shift + F6*** pada keyboard).
4. Jika ada kesalahan, lakukan perbaikan yang diperlukan.

## Praktek 3: Mengolah data dengan String

1. Buat class baru dengan nama **Lab\_2\_PeriksaKarakterString** dan lengkapi hingga seperti berikut:

*import javax.swing.JOptionPane;*

*public class Lab\_2\_PeriksaKarakterString {*

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

 *String kalimat;*

*char cari;*

*int jumlah=0;*

*kalimat = JOptionPane.showInputDialog(null,"Masukkan kalimat anda! ");*

 *cari = JOptionPane.showInputDialog(null,"Masukkan karakter yg ingin dihitung jumlahnya! ").charAt(0);*

*for (int i = 0; i < kalimat.length(); i++) {*

 *if (cari == kalimat.charAt(i)) {*

 *jumlah++;*

 *}*

*}*

*JOptionPane.showMessageDialog(null, "jumlah huruf " + cari + "= " + jumlah);*

 *}*

*}*

1. Pelajari kode program tersebut dan buat analisis proses apa saja yang dilakukan oleh program tersebut!

**Praktek 4: While statement**

1. Buat class baru dengan nama **Lab\_2\_PeriksaKarakterString\_2**.
2. Salin isi dari main program pada praktek 3
3. Modifikasi program sehingga perulangannyamenggunakan while – *statement*
4. Jalankan class yang sudah dimodifikasi tersebut, dengan cara cara **klik kanan pada nama file** 🡪 **Run File** (atau dengan menekan tombol ***Shift + F6*** pada keyboard).
5. Jika ada kesalahan, lakukan perbaikan yang diperlukan.

# Analisa Hasil Praktek

Berdasarkan praktek yang telah Anda lakukan, buat kesimpulan:

# Tugas

1. Lanjutkan praktek jika pelaksanaan di Lab belum selesai.

# Appendix 1: Form Data Praktek

Nama : ……………………….

NIM : ……………………….

Tgl : ……………………….

|  |  |
| --- | --- |
| **Penggalan *Source code*** | **Keluaran / *Output*** |
|  |  |
|  |  |
|  |  |