MODUL PEMROGRAMAN BERORIENTASI OBJEK

CTI 211

MODUL PERTEMUAN 3

CONTROL FLOW STATEMENT DALAM OOP

DISUSUN OLEH

7174 - SAWALI WAHYU, S.KOM, M.KOM

UNIVERSITAS ESA UNGGUL

FAKULTAS ILMU KOMPUTER

TAHUN 2020

SUBTOPIK 1 TOPIK SESI INI

A. Kemampuan Akhir Yang Diharapkan

Setelah mempelajari modul ini, diharapkan mahasiswa mampu :

1. Mahasiswa mampu memahami kontrol struktur kontrol dalam *java programming*
2. Mahasiswa mampu membuat dan mengimplementasikan fungsi dalam kontrol struktur ke dalam sebuah aplikasi
3. Mahasiswa mampu membuat fungsi struktur kontrol dalam logika program aplikasi

B. Uraian Materi

1) Konsep Dasar Struktur Kontrol
2) Struktur Kontrol Pemilihan
    i. Selection Statement
    ii. Iteration Statement
    iii. Jump Statement

1. Konsep Dasar Struktur Kontrol

Sebuah bahasa pemrograman menggunakan struktur atau statemen kontrol (*control statements*) untuk mengontrol jalannya aliran eksekusi. setiap pernyataan dieksekusi setelah pernyataan sebelumnya sesuai dengan urutannya. Pada bagian ini, akan mempelajari tentang struktur kontrol dimana dapat mengubah cara eksekusi pada pernyataan yang dibuat di program.

- 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 program.

2. Struktur Kontrol Pemilihan

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

Selection Statement

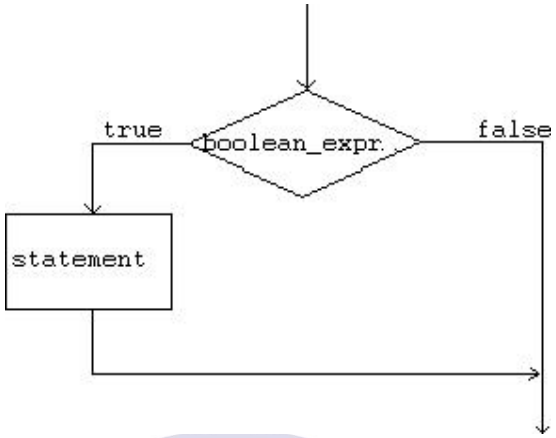
Statement if

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

Bentuk dari pernyataan if :

```
if( boolean_expression )
        statement;

 atau

if( boolean_expression ){ stat
        ement1; statement2;

        . . .

}
```

dimana, *boolean_expression* adalah sebuah penyataan logika *(true/false)* atau variabel bertipe *boolean*.



Berikut ini adalah potongan kode dari pernyataan if:

```
int grade = 68;
if( grade > 60 ) System.out.println("Congratulations!");
```

Atau

```
int grade = 68;
if( grade > 60 ){
System.out.println("Congratulations!");
System.out.println("You passed!");
}
```

Statement if-else

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

Bentuk statement if-else,

```
if( boolean_expression ) statement;
else
statement;
```

dapat juga ditulis seperti,

```
if( boolean_expression ){ statement1; statement2;
. . .
}
else{
```

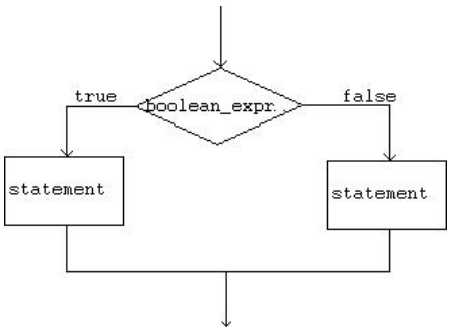
```
……
}
```

```
statement1; statement2;
. . .
```

Berikut ini contoh code statement if-else,

```
int grade = 68;
if( grade > 60)
{
System.out.println("Congratulations!");
}
else
System.out.println("Sorry you failed");
}
```

```
atau
int grade = 68;
if( grade > 60 )
{
System.out.println("Congratulations!");
System.out.println("You passed!");
}
else{
System.out.println("Sorry you failed");
}
```



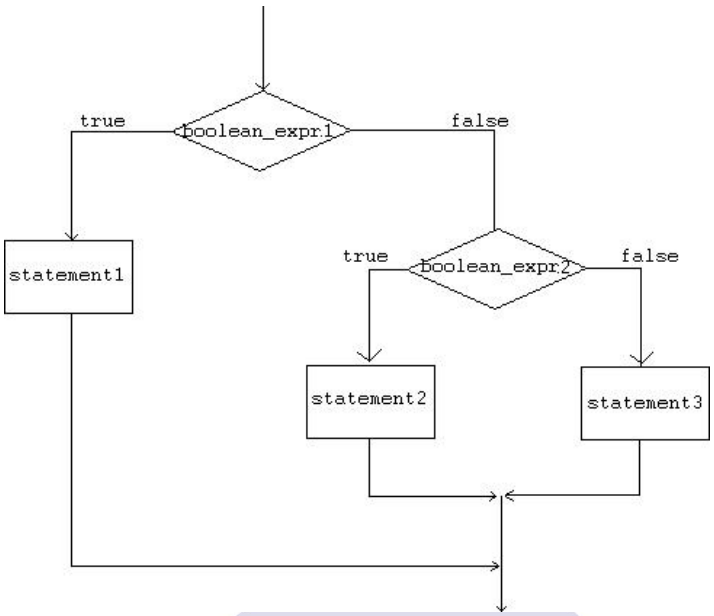
Statement if-else-if

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

Bentuk statement if-else if,

```
if( boolean_expression1 )
        statement1;
else if( boolean_expression2 )
        statement2;
else
        statement3;
```

Sebagai catatan : anda dapat memiliki banyak blok else-if sesudah pernyataan *if*. Blok *else* bersifat opsional dan dapat dihilangkan. Pada contoh yang ditampilkan di atas, jika *boolean_expression1* bernilai *true*, maka program akan mengeksekusi *statement1* dan melewati pernyataan yang lain. Jika *boolean_expression2* bernilai *true*, maka program akan mengeksekusi *statement2* dan melewati *statement3*.



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;   //
statement2;   //block 1
. . .      //
break;

case case_selector2:
statement1;   //
statement2;   //block 2
. . .      //
break;
. . .
default:
}
```

```
statement1;   //
statement2;   //block n
. . .     //
break;
```

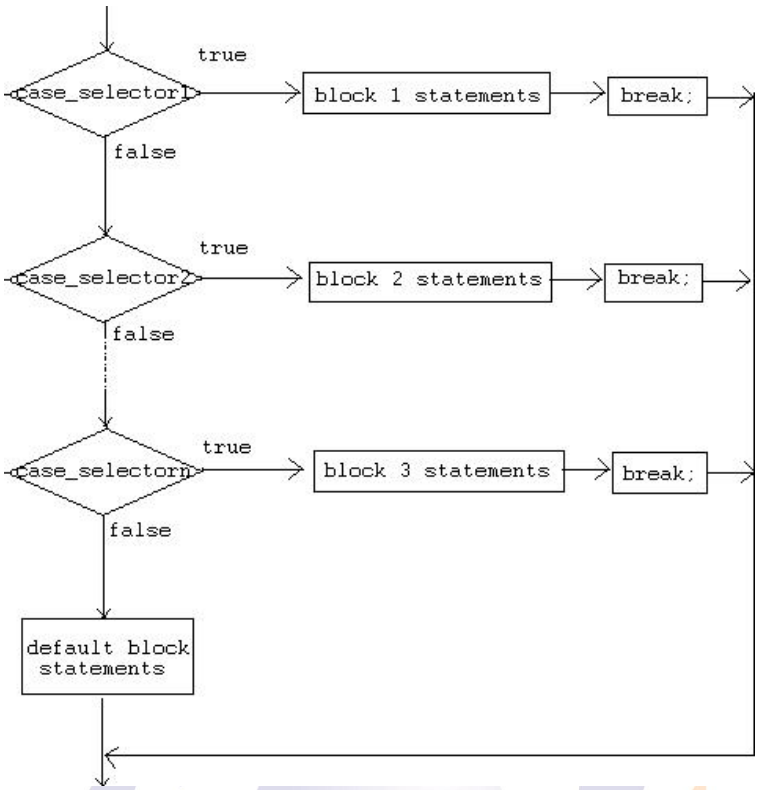
switch_expression adalah ekspresi integer atau karakter dan case_selector1, case_selector2 dan seterusnya adalah konstanta unik dari nilai integer atau karakter.

Ketika pernyataan switch ditemukan pada potongan kode program, java pertama kali akan memeriksa switch_expression, dan menuju ke case yang akan menyamakan nilai yang dimiliki oleh switch_expression. Selanjutnya program akan mengeksekusi pernyataan pada dari kode setelah case yang ditemukan sampai menemui pernyataan break, selanjutnya akan mengabaikan pernyataan yang lainnya hingga akhir dari struktur dari pernyataan switch.

Jika tidak ditemui case yang cocok, maka program akan mengeksekusi blok default. Sebagai catatan, bahwa bagian blok default adalah opsional. Sebuah pernyataan switch bisa jadi tidak memiliki blok kode default.

CATATAN:

i. Tidak seperti pada pernyataan if, beberapa pernyataan pada struktur pernyataan switch akan dieksekusi tanpa memerlukan tanda kurung kurawal ({}).
ii. Ketika sebuah case pada pernyataan switch menemui kecocokan, semua pernyataan pada case tersebut akan dieksekusi. Tidak hanya demikian, pernyataan lain yang berada pada case yang sesuai juga akan dieksekusi.
iii. Untuk menghindari program mengeksekusi pernyataan pada case berikutnya, kita menggunakan pernyataan break sebagai pernyataan akhir pada setiap blok case.



Iteration Statement

Bentuk statemen iterasi pada Java adalah for, while, dan do-while. Statemen ini sering disebut sebagai loop.

Statemen While

Statemen while merupakan salah satu statemen yang digunakan untuk memproses suatu statemen atau beberapa statemen beberapa kali. Bentuk statemen while ini adalah sebagai berikut.

while (LoopCondition)
Statemen;//statemen ini bisa lebih dari satu

Statemen Do While

Statemen do-while digunakan untuk mengulang proses. Bentuk statemen do-while ini adalah sebagai berikut

do
Statemen
while (LoopCondition);

Statemen For

Statemen for digunakan utuk mengulang pengeksekusian terhadap satu atau sejumlah pernyataan. Bentuk statemen for adalah sebagai berikut.

for (InitializationExpression; LoopCondition; StepExpression)

Statemen Nested Loops

Statemen loop juga mempunyai bentuk loop dalam loop (nested loop) atau dengan kata lain Java mengijinkan suatu loop berada dalam loop yang lain.

Statemen Jump

Java mendukung terhadap tiga jenis statemen jump: break, continue, dan return.

Statemen Break

Dalam Java, statemen break mempunyai tiga kegunaan, yaitu:.

a. Membatasi (terminates) sebuah urutan statemen dalam statemen switch.
b. Dapat digunakan untuk keluar dari sebuah loop.
c. Dapat digunakan sebagai sebuah pengganti bentuk dari goto.

Menggunakan Break Sebagai bentuk Goto

Pada bahasa Java, tidak mempunyai statemen goto seperti pada bahasa lain, tetapi sebagai penggantinya menggunakan statemen break. Break digunakan ketika kita ingin keluar dari satu atau lebih blok kode. Blok ini bukan menjadi bagian dari sebuah loop atau switch, dan break bekerja dengan sebuah label. Bentuk umum dari break ini adalah:

break label;

Label adalah sebuah nama dari label yang menunjukan sebuah blok kode. Penggunaan yang paling umum dari statemen break berlabel adalah untuk keluar dari nested loops.

Statemen Continue

Statemen continue merupakan kebalikan dari statemen break. Statemen continue digunakan untuk mengarahkan eksekusi ke iterasi berikutnya pada statemen pengulangan.

Statemen Return

Statemen kontrol return digunakan/biasanya secara eksplisit me-return (mengembalikan sebuah nilai) dari sebuah metode.

C. Latihan

Nasted If :

```
class NestedIfDemo
    {
    public static void main(String[] args)
    {
    int      a=1,b=2,c=3,d; if (a<b)
    if (b<c)

    d=c+b+a; else
    d=c-b-a;
    else
    d=c-b+a;
    System.out.println("Nilai D ="+d    );
    }
    }
```

Statement Bertingkat

```
class NestedIfDemo
{
public static void main(String[] args)
{
int      a=1,b=2,c=3,d; if (a<b)
if (b<c)
```

```
d=c+b+a; else
d=c-b-a;
else
d=c-b+a;
System.out.println("Nilai D ="+d    );
}
}
```

Selection Statement

```
Class MenuJurusanIfElse
{
public static void main(String args[]) throws java.io.IOException
{
char pilihan; do
{
System.out.println("PROGRAM MENGGUNAKAN IF-ELSE");
System.out.println("JURUSAN ELEKTRO POLINES:");
System.out.println(" 1. Prodi INFOKOM");
System.out.println(" 2. Prodi LISTRIK");
System.out.println(" 3. Prodi ELEKTRONIKA");
System.out.println(" 4. Prodi TELEKOMUNIKASI");
System.out.print("PILIH SALAH SATU:");
pilihan = (char) System.in.read();
}
while( pilihan < '1' || pilihan > '4'); System.out.println("¥n");

if (pilihan=='1')
{
System.out.println("FOKUS ILMU KOMPUTER");
}
else if (pilihan=='2')
{
System.out.println("FOKUS LISTRIK");
}
```

```
else if (pilihan=='3')
{
System.out.println("FOKUS ELEKTRONIKA");
}
else if (pilihan=='4')
{
System.out.println("FOKUS TELEKOMUNIKASI");
}
}
}
```

Looping Statement

```
class BreakNestedLoopDemo
{
public static void main(String args[])
{
for(int i=0; i<10; i++)
{
System.out.print("loop ke " + i + ": ");
if (i==5) break;// berhenti loop jika j = 5 for(int j=0; j<10; j++)
{
if(j == 8) break; // berhenti loop jika j = 8 System.out.print(j);
}
System.out.println();
}
System.out.println("Loops complete.");
}
}
```

A. Daftar Pustaka

1. Herbert Schildt, 2014, E-Book Java A Beginner Guide Sixth Edition : Create, Compile and Run Programs Today,by McGraw-Hill Education (Publisher).
2. Reges, Stuart; Stepp, Marty. 2016. Building Java Programs: A Back to Basic Approach 4th Edition. Pearson. http://www.buildingjavaprograms.com/
3. Java Platform Standard Edition Documentation. http://docs.oracle.com/javase/8/docs/