**Modul 6 \_Ridhatul Nafsi**

**Struktur Kontrol**

**1. Struktur Kontrol KeputusanStatement if**

**a. Statement-if**

menentukan sebuah statement (atau blok kode) yang akan dieksekusi jika dan hanya jika persyaratan boolean (boolean statement) bernilai true.Bentuk dari statement if,

if( boolean\_expression )

statement;

atau

if( boolean\_expression ){

statement1;

statement2;

. . .

}

try{

String temp = dataIn.readLine();

}

catch( IOException e ){

System.out.println(“Error in getting input”);

}

**b. Statement if-else**

digunakan apabila kita ingin mengeksekusi sebuah statement dengan kondisi true dan statement 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");

**c. Statement if-else-if**

Statement pada bagian else dari blok if-else dapat menjadi struktur if-else yang lain.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;

contoh code statement if-else-if

int grade = 68;

if( grade > 90 ){

System.out.println("Very good!");

}

else if( grade > 60 ){

System.out.println("Very good!");

}

else{

System.out.println("Sorry you failed");

}

**d. Statement switch**

Cara lain untuk membuat percabangan adalah dengan menggunakan kata kunci switch. Dengan menggunakan switch kita bisa melakukan percabangan dengan persyaratan yang beragam. 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 persyaratan integer atau character dan case\_selector1, case\_selector2 dan seterusnya adalah konstanta nilai integer yang unique (unik).

**2. Struktur Kontrol Perulangan**

Struktur kontrol pengulangan adalah statement dari Java dimana kita bisa mengeksekusi blok code berulang-ulang dalam kurun nilai tertentu. Ada tiga macam jenis struktur kontrol pengulangan yaitu while, do-while, dan for-loops.

**a. while loop**

Statement while loop adalah statement atau blok statement yang diulang-ulang sampai mencapai kondisi yang cocok. Bentuk statement while,

while( boolean\_expression ){

statement1;

statement2;

. . .

}

Statement di dalam while loop akan dieksekusi berulang-ulang selama

boolean\_expression bernilai true.

Contoh, pada code dibawah ini,

int i = 4;

while ( i > 0 ){

System.out.print(i);

i--;

}

Contoh diatas akan mencetak angka 4321 pada layar.

**b. do-while loop**

Do-while loop mirip dengan while-loop. Statement di dalam do-while loop akan dieksekusi beberapa kali selama kondisi bernilai true. Perbedaan antara while dan do-while loop adalah dimana statement di dalam do-while loop dieksekusi sedikitnya satu kali.Bentuk statement do-while,

do{

statement1;

statement2;

. . .

}while( boolean\_expression );

Statement di dalam do-while loop akan dieksekusi pertama kali, dan dilakukan pengecekan kondisi dari boolean\_expression. Jika nilai tersebut belum mencapai nilai yang diinginkan, statement akan dieksekusi lagi.

contoh do-while loop:

int x = 0;

do

{

System.out.println(x);

x++;

}while (x<10);

Contoh ini akan memberikan output 0123456789 pada layar.

**c. for loop**

Seperti pada struktur pengulangan sebelumnya yaitu melakukan pengulangan eksekusi code beberapa kali. Bentuk dari for loop,

for (InitializationExpression; LoopCondition; StepExpression){

statement1;

statement2;

. . .

}

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

contoh dari for loop,

int i;

for( i = 0; i < 10; i++ ){

System.out.print(i);

}

Pada contoh ini, statement i=0 merupakan inisialisasi dari variabel. Selanjutnya, kondisi i<10 diperiksa. Jika kondisi bernilai true, statement di dalam for loop dieksekusi. Kemudian, statement i++ dieksekusi, dan dilakukan pengecekan kondisi. Kondisi ini akan dilakukan berulang-ulang sampai mencapai nilai yang salah (false).

**3. Branching Statements**

**A. break statement**

Statement break memiliki dua bentuk: unlabeled dan labeled.

**a. Unlabeled break statement**

Unlabeled menghentikan jalannya statement switch. Anda bisa juga menggunakan bentuk unlabeled untuk menghentikan for, while atau do-while loop. Contohnya,

String names[] = {"Beah", "Bianca", "Lance", "Belle",

"Nico", "Yza", "Gem", "Ethan"};

String searchName = "Yza";

boolean foundName = false;

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

if( names[i].equals( searchName )){

foundName = true;

break;

}

}

if( foundName ){

System.out.println( searchName + " found!" );

}

else{

System.out.println( searchName + " not found." );

}

Pada contoh ini, jika string “Yza” ditemukan, pengulangan pada for loop akan dihentikan dan akan melanjutkan ke proses berikutnya.

**b. Labeled break statement**

Bentuk labeled form dari statement break akan menghentikan statement luar, dimana diidentifikasikan berupa label pada statement break. Program berikut ini akan mencari nilai dalam array dua dimensi. Terdapat dua pengulangan bersarang (nested loop). Ketika sebuah nilai ditemukan, labeled break akan menghentikan statement yang diberi label searchLabel, dimana label ini berada di luar.

int[][] numbers = {{1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

int searchNum = 5;

boolean foundNum = false;

searchLabel:

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

for( int j=0; j<numbers[i].length; j++ ){

if( searchNum == numbers[i][j] ){

foundNum = true;

break searchLabel;

}

}

}

if( foundNum ){

System.out.println( searchNum + " found!" );

}

else{

System.out.println( searchNum + " not found!" );

}

Statement break menghentikan sementera labeled statement; ia tidak lagi menjalankan flow control pada label. Flow control pada label akan di-transfer secara otomatis mengikuti labeled statement.

**B. Continue statement**

**a. Unlabeled continue statement**

Bentuk unlabeled akan melewati akhir statement pada bagian yang dalam dan memeriksa boolean expression yang mengkontrol loop, pada dasarnya akan melewati bagian pengulangan pada loop. Berikut ini adalah contoh dari penghitungan angka dari “Beah” dalam suatu array.

String names[] = {"Beah", "Bianca", "Lance", "Beah"};

int count = 0;

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

if( !names[i].equals("Beah") ){

continue; //skip next statement

}

count++;

}

System.out.println("There are " + count + " Beahs in the

list");

**b. Labeled continue statement**

Bentuk labeled akan melanjutkan sebuah statement dengan melewati pengulangan yang sedang berjalan dari loop terluar yang diberi label (tanda). outerLoop:

for( int i=0; i<5; i++ ){

for( int j=0; j<5; j++ ){

System.out.println("Inside for(j) loop"); //message1

if( j == 2 ) continue outerLoop;

}

System.out.println("Inside for(i) loop"); //message2

}

Pada contoh ini, pesan ke-2 tidak dicetak, karena statement continue akan melewati pengulangan yang sedang berjalan.

**C. Return statement**

Statement return digunakan untuk keluar dari sebuah fungsi (method). Statement return memiliki dua bentuk: menggunakan sebuah nilai, dan tidak memberikan nilai. Untuk memberikan sebuah nilai, cukup berikan nilai (atau ekspresi yang menghasilkan sebuah nilai) sesudah return. Contohnya,

return ++count;

atau

return "Hello";

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

return;

Pembuat : Ridhatul Nafsi