MODUL 7

PEMILIHAN 2

KOMPETENSI

1. Mahasiswa memahami tentang operator logika
2. Mahasiswa mampu menyelesaikan permasalahan dengan menggunakan sintaks pemilihan bersarang
3. Mahasiswa mampu membuat sebuah program Java yang memanfaatkan sintaks pemilihan bersarang

MATERI DASAR

a. Percabangan dalam percabangan (Nested if)

Kita telah mempelajari penggunaan pernyataan IF untuk memilih sebuah tidak, pernyataan IF-ELSE untuk memilih antara dua tindakan, serta pernyataan IF-ELSE IF-ELSE dan SWITCH-CASE untuk memilih beberapa tindakan (3 atau lebih).

Terkadang kita membutuhkan pengambilan keputusan dalam bentuk level (bertingkat) sehingga di dalam suatu pernyataan IF (atau IF-ELSE) bisa saja terdapat pernyataan IF (atau IF-ELSE) yang lain. Jenis percabangan seperti ini disebut NESTED IF (percabangan bersarang).

Secara umum, bentuk penulisan pernyataan NESTED IF adalah sebagai berikut:

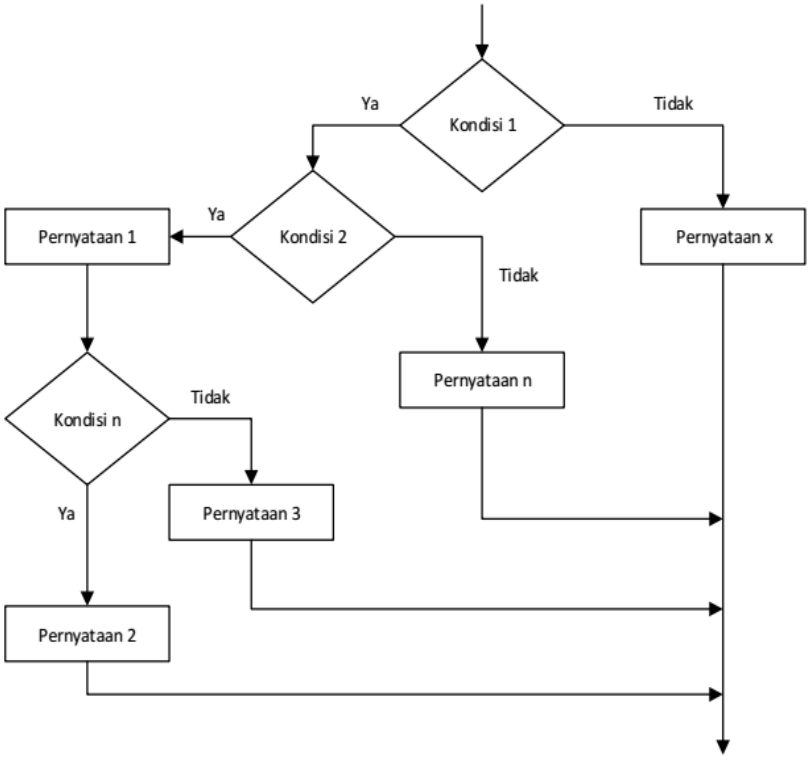
```
if (kondisi 1){
  if (kondisi 2){
      pernyataan 1;
      …
      …
      if (kondisi n){
          pernyataan 2;
      } else {
          pernyataan 3;
      }
  } else {
      pernyataan n;
  }
} else {
      pernyataan x;
}
```

Pada bentuk penulisan pernyataan NESTED-IF tersebut, kondisi yang akan diseleksi pertama kali adalah kondisi IF yang berada di posisi terluar (kondisi 1).

- Jika kondisi 1 bernilai salah, maka pernyataan ELSE terluar (pasangan dari IF yang bersangkutan) yang akan diproses. Namun, jika pernyataan ELSE (pasangan dari IF) tidak ditulis, maka penyeleksian kondisi akan dihentikan.
- Jika ternyata kondisi 1 bernilai benar, maka kondisi berikutnya yang lebih dalam (kondisi 2) akan diseleksi. Jika kondisi 2 bernilai salah, maka pernyataan ELSE (pasangan dari IF yang bersangkutan) yang akan diproses. Namun, jika pernyataan ELSE (pasangan dari IF) tidak ditulis, maka penyeleksian kondisi akan dihentikan.

Dengan cara yang sama, penyeleksian kondisi akan dilakukan sampai dengan kondisi n, jika kondisi-kondisi sebelumnya bernilai benar.

Flowchart sintaks pemilihan bersarang ditunjukkan pada Gambar berikut.



Flowchart Sintaks Pemilihan Bersarang

Berikut ini adalah contoh penggunaan NESTED IF ketika seseorang akan melakukan pembayaran di kasir. Kasir akan memberikan pertanyaan sebagai berikut:

Apakah pelanggan mempunyai kartu anggota?

- TRUE: Pelanggan mempunyai kartu anggota
  - Apakah total harga barang belanjaan lebih dari Rp 500.000?
    - ❖ TRUE: Total harga barang belanjaan lebih dari Rp 500.000

Pelanggan mendapatkan diskon Rp 50.000

- ❖ FALSE: Total harga barang belanjaan tidak lebih dari Rp 500.000

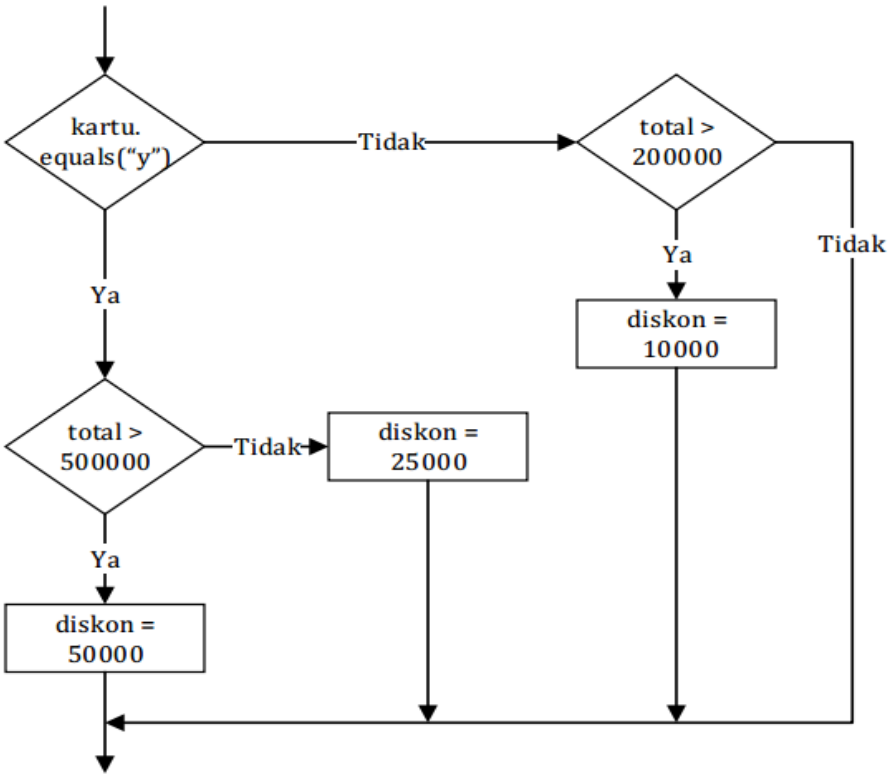
  Pelanggan mendapatkan diskon Rp 25.000

- FALSE: Pelanggan tidak mempunyai kartu anggota
  - Apakah total harga barang belanjaan lebih dari Rp 200.000?
    - ❖ TRUE: Total harga barang belanjaan lebih dari Rp 200.000

      Pelanggan mendapatkan diskon Rp 10.000
    - ❖ FALSE: Total harga barang belanjaan tidak lebih dari Rp 200.000

      Pelanggan tidak mendapatkan diskon

Untuk lebih memperjelas alur percabangan pada contoh kasus tersebut, perhatikan flowchart pada Gambar berikut:



Contoh Flowchart

Berikut ini adalah contoh kode program untuk penggunaan NESTED IF pada contoh kasus pembayaran di kasir:

```
import java.util.Scanner;
public class kasir {
    public static void main(String[] args) {
        int total, diskon, bayar;
        String kartu;
        Scanner sc = new Scanner (System.in);
        System.out.print("Apakah pelanggan mempunyai kartu anggota (y atau t)? ");
        kartu = sc.nextLine();
        System.out.print("Berapa total harga barang belanjaan? Rp ");
        total = sc.nextInt();
        if (kartu.equals("y")) {
            if (total > 500000) {
                diskon = 50000;
            } else {
                diskon = 25000;
            }
        } else {
            if (total > 200000) {
                diskon = 10000;
            } else {
                diskon = 0;
            }
        }
        bayar = total - diskon;
        System.out.println("Total yang harus dibayar: Rp " + bayar);
    }
}
```

Pada kode program tersebut, kasir diminta untuk memasukkan input, apakah pelanggan mempunyai kartu anggota atau tidak. Selanjutnya kasir juga perlu memasukkan total harga barang belanjaan. Kondisi yang akan diseleksi pertama kali adalah nilai dari variabel “kartu”. Jika pengguna memasukkan input “y”, maka seleksi kondisi ini bernilai benar, dan selanjutnya dilakukan penyeleksian total harga barang belanjaan untuk menentukan diskon yang diperoleh. Gambar 4 menunjukkan hasil keluaran program ketika dijalankan.

Berikut adalah contoh hasil output program:

```
run:
Apakah pelanggan mempunyai kartu anggota (y atau t)? y
Berapa total harga barang belanjaan? Rp 250000
Total yang harus dibayar: Rp 225000
BUILD SUCCESSFUL (total time: 5 seconds)

run:
Apakah pelanggan mempunyai kartu anggota (y atau t)? t
Berapa total harga barang belanjaan? Rp 300000
Total yang harus dibayar: Rp 290000
BUILD SUCCESSFUL (total time: 5 seconds)
```

b. Penggunaan operator logika dalam percabangan

Operator logika merupakan jenis operator yang akan membandingkan logika hasil dari Relational operator.

Tabel Operator Logika

Operator	Deskripsi
&& (AND)	Jika semua operand bernilai benar (TRUE), maka kondisi tersebut dinyatakan bernilai benar.
|| (OR)	Jika salah satu operand bernilai benar (TRUE), maka kondisi tersebut dinyatakan bernilai benar.
! (NOT)	Operator ini juga disebut sebagai invers atau negasi. Operator ini akan membalikan nilai boolean true menjadi false dan begitu juga sebaliknya.

Contoh implementasi operator logika dalam percabangan:

Operator logika digunakan untuk menggabungkan beberapa kondisi. Jika Anda ingin program yang menghasilkan output "Welcome!" hanya ketika usia variabel lebih besar dari 18 dan uang variabel lebih besar dari 500. Dalam kasus tersebut, Anda bisa menggunakan logika AND (&&) (deretan kode bagian atas) untuk mempersingkat percabangan bersarang yang ada pada deretan kode bagian bawah.

```
int age = 30;
int money = 600;

if (age > 18) {
      if (money > 500) {
            System.out.println("Welcome!");
            }
      } else {
            System.out.println("Not eligible!");
      }

//Outputs "Welcome!"
```

```
int age = 30;
int money = 600;
if (age > 18 && money > 500) {
      System.out.println("Welcome!");
}

//Outputs "Welcome!"
```

Percobaan 1

1. Buatlah sebuah project di Netbeans dan simpan dengan nama Pemilihan2.java
2. Tambahkan import library Scanner ke dalam program dengan cara mengetik:

```
import java.util.Scanner;
```

3. Mendeklarasikan Scanner dengan nama **sc** sebagai berikut:

```
Scanner sc = new Scanner(System.in);
```

4. Membuat variabel bertipe int dengan nama nilai, dengan cara mengetik:

```
int nilai;
```

5. Tambahkan kode berikut untuk menerima input dari keyboard:

```
System.out.print("Masukkan nilai ujian (0 - 100) : ");
nilai = sc.nextInt();
```

6. Buatlah struktur pengecekan kondisi bersarang.

   Pengecekan pertama digunakan untuk memastikan bahwa nilai yang dimasukkan berada pada rentang 0 – 100.

   Jika nilai berada pada rentang 0 – 100, maka akan dilakukan pengecekan status kelulusan mahasiswa, yaitu:

   - jika nilai di antara 90 – 100 maka nilainya A
   - jika nilai di antara 80 – 89 maka nilainya B
   - jika nilai di antara 60 – 79 maka nilainya C
   - jika nilai di antara 50 – 59 maka nilainya D
   - jika nilai di antara 0 – 49 maka nilainya E

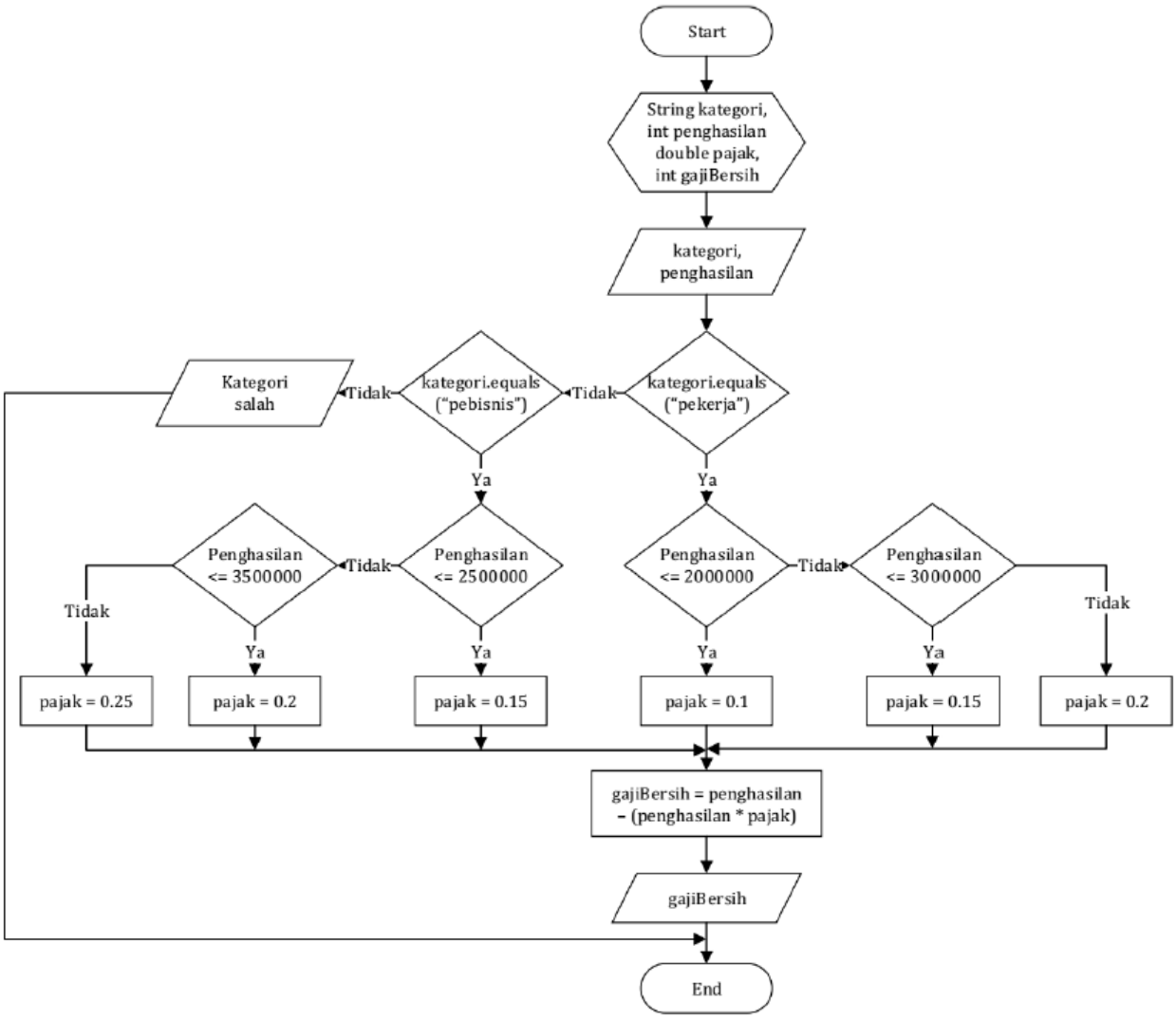
   Sedangkan jika nilai berada di luar rentang 0 – 100, maka ditampilkan informasi bahwa nilai yang dimasukkan tidak valid.

```
if (nilai >= 0 && nilai <= 100){
    if (nilai >= 90 && nilai <= 100){
        System.out.println("Nilai A, EXCELLENT!");
    } else if (nilai >= 80 && nilai <= 89){
        System.out.println("Nilai B, pertahankan prestasi anda!");
    } else if (nilai >= 60 && nilai <= 79){
        System.out.println("Nilai C, tingkatkan prestasi anda!");
    } else if (nilai >= 50 && nilai <= 59){
        System.out.println("Nilai D, tingkatkan belajar anda");
    } else {
        System.out.println("Nilei E, anda tidak lulus!");
    }
} else {
    System.out.println("Nilai yang anda masukkan tidak valid!");
}
```

7. Jalankan program tersebut dan amati apa yang terjadi?

Percobaan 2

1. Perhatikan flowchart berikut ini:



Flowchart tersebut digunakan untuk menghitung gaji bersih seseorang setelah dipotong pajak sesuai dengan kategorinya (pekerja dan pebisnis) dan besarnya penghasilan.

2. Buka text editor, lalu simpan file dengan nama Pemilihan2_2.java
3. Tambahkan import library Scanner.

```
import java.util.Scanner;
```

4. Deklasrasikan Scanner dengan nama **sc**.

```
Scanner sc = new Scanner(System.in);
```

5. Deklarasikan variabel **kategori**, **penghasilan**, **gajiBersih**, dan **pajak**;

```
String kategori;
int penghasilan, gajiBersih;
double pajak = 0;
```

6. Tambahkan kode berikut ini untuk menerima input dari keyboard.

```
System.out.print("Masukkan kategori : ");
kategori = sc.nextLine();
System.out.print("Masukkan besarnya penghasilan : ");
penghasilan = sc.nextInt();
```

7. Buatlah struktur pengecekan kondisi bersarang. Pengecekan pertama digunakan untuk mengecek kategori (pekerja atau pebisnis). Selanjutnya dilakukan pengecekan kedua untuk menentukan besarnya pajak berdasarkan penghasilan yang telah dimasukkan. Kemudian tambahkan kode program untuk menghitung gaji bersih yang diterima setelah dipotong pajak

```
if (kategori.equalsIgnoreCase("pekerja")){
    if (penghasilan <= 2000000){
        pajak = 0.1;
    } else if (penghasilan <= 3000000){
        pajak = 0.15;
    } else {
        pajak = 0.2;
    }
    gajiBersih = (int)(penghasilan - (penghasilan * pajak));
    System.out.println("Gaji bersih yang anda terima : " +gajiBersih);
} else if (kategori.equalsIgnoreCase("pebisnis")){
    if (penghasilan <= 2500000){
        pajak = 0.15;
    } else if (penghasilan <= 3500000){
        pajak = 0.2;
    } else {
        pajak = 0.25;
    }
    gajiBersih = (int)(penghasilan - (penghasilan * pajak));
    System.out.println("Gaji bersih yang anda terima : " +gajiBersih);
} else {
    System.out.println("Kategori yang anda masukkan salah");
}
```

8. Jalankan program tersebut dan amati apa yang terjadi.

TUGAS

1. Lakukan percobaan diatas. Pahami seluruh percobaan yang telah anda lakukan. Capture seluruh listing program dan hasil running program. Tampilkan hasil capture tersebut dalam laporan praktikum anda.