MODUL V-I
SELEKSI KONDISI & PERULANGAN

Pertemuan : 5
Waktu : 8 x 60 Menit (Online)

1.1 Tujuan Modul V-1

Setelah mahasiswa mempelajari materi ini, diharapkan dapat :

1. Memahami seleksi kondisi.

1.2 Landasan Teori

1.2.1 Seleksi Kondisi

Seleksi kondisi digunakan untuk mengontrol alur program. Analoginya mirip seperti fungsi rambu lalu lintas di jalan raya. Kapan kendaraan diperbolehkan melaju dan kapan harus berhenti diatur oleh rambu tersebut. Seleksi kondisi pada program juga kurang lebih sama, kapan sebuah blok kode akan dieksekusi dikontrol.

Yang dijadikan acuan oleh seleksi kondisi adalah nilai bertipe bool, bisa berasal dari variabel, ataupun hasil operasi perbandingan. Nilai tersebut menentukan blok kode mana yang akan dieksekusi.

Go memiliki 2 macam keyword untuk seleksi kondisi, yaitu if else dan switch.

1.2.1.1 Seleksi Kondisi Menggunakan Keyword if, else if, & else

Cara penerapan if-else di Go sama seperti pada bahasa pemrograman lain. Yang membedakan hanya tanda kurungnya *(parentheses)*, di Go tidak perlu ditulis. Kode berikut merupakan contoh penerapan seleksi kondisi if else, dengan jumlah kondisi 4 buah.

Contoh penggunaan dari seleksi kondisi diatas sebagai berikut :

```
var point = 8

if point == 10 {
    fmt.Println("lulus dengan nilai sempurna")
} else if point > 5 {
    fmt.Println("lulus")
} else if point == 4 {
    fmt.Println("hampir lulus")
} else {
    fmt.Printf("tidak lulus. nilai anda %d\n", point)
}
```

Dari ke-empat kondisi di atas, yang terpenuhi adalah `if point > 5`, karena nilai variabel `point` memang lebih besar dari `5`. Maka blok kode tepat dibawah kondisi tersebut akan dieksekusi (blok kode ditandai kurung kurawal buka dan tutup), hasilnya text `"lulus"` muncul sebagai output.

Skema if else Go sama seperti pada pemrograman umumnya. Yaitu di awal seleksi kondisi menggunakan `if`, dan ketika kondisinya tidak terpenuhi akan menuju ke `else` (jika ada). Ketika ada banyak kondisi, gunakan `else if`.

1.2.1.2 Variabel Temporary Pada if - else

Variabel temporary adalah variabel yang hanya bisa digunakan pada blok seleksi kondisi dimana ia ditempatkan saja. Penggunaan variabel ini membawa beberapa manfaat, antara lain:

- Scope atau cakupan variabel jelas, hanya bisa digunakan pada blok seleksi kondisi itu saja
- Kode menjadi lebih rapi
- Ketika nilai variabel tersebut didapat dari sebuah komputasi, perhitungan tidak perlu dilakukan di dalam blok masing-masing kondisi.

Adapun contoh dari Variabel Temporary Pada if - else , seperti dibawah ini :

```
var point = 8840.0

if percent := point / 100; percent >= 100 {

    fmt.Printf("%.1f%s perfect!\n", percent, "%")

} else if percent >= 70 {

    fmt.Printf("%.1f%s good\n", percent, "%")

} else {

    fmt.Printf("%.1f%s not bad\n", percent, "%")

}
```

Variabel `percent` nilainya didapat dari hasil perhitungan, dan hanya bisa digunakan di deretan blok seleksi kondisi itu saja.

Deklarasi variabel temporary hanya bisa dilakukan lewat metode type inference yang menggunakan tanda `:=`. Penggunaan keyword `var` disitu tidak diperbolehkan karena akan menyebabkan error.

1.2.1.3 Seleksi Kondisi Menggunakan Keyword switch - case

Switch merupakan seleksi kondisi yang sifatnya fokus pada satu variabel, lalu kemudian di-cek nilainya. Contoh sederhananya seperti penentuan apakah nilai variabel `x` adalah: `1`, `2`, `3`, atau lainnya.

```
var point = 6
switch point {
case 8:
    fmt.Println("perfect")
case 7:
    fmt.Println("awesome")
default:
    fmt.Println("not bad")
}
```

Pada kode di atas, tidak ada kondisi atau `case` yang terpenuhi karena nilai variabel `point` tetap `6`. Ketika hal seperti ini terjadi, blok kondisi `default` dipanggil. Bisa dibilang bahwa `default` merupakan `else` dalam sebuah switch.

Perlu diketahui, switch pada pemrograman Go memiliki perbedaan dibanding bahasa lain. Di Go, ketika sebuah case terpenuhi, tidak akan dilanjutkan ke pengecekkan case selanjutnya, meskipun tidak ada keyword `break` di situ. Konsep ini berkebalikan dengan switch pada umumnya, yang ketika sebuah case terpenuhi, maka akan tetap dilanjut mengecek case selanjutnya kecuali ada keyword `break`.

1.2.1.4 Pemanfaatan case Untuk Banyak Kondisi

Sebuah `case` dapat menampung banyak kondisi. Cara penerapannya yaitu dengan menuliskan nilai pembanding-pembanding variabel yang di-switch setelah keyword `case` dipisah tanda koma (`,`).

```
var point = 6

switch point {
case 8:
    fmt.Println("perfect")
case 7, 6, 5, 4:
    fmt.Println("awesome")
default:
    fmt.Println("not bad")
}
```

Kondisi `case 7, 6, 5, 4:` akan terpenuhi ketika nilai variabel `point` adalah 7 atau 6 atau 5 atau 4.

1.2.1.5 Kurung Kurawal Pada Keyword case & default

Tanda kurung kurawal (`{ }`) bisa diterapkan pada keyword `case` dan `default`. Tanda ini opsional, boleh dipakai boleh tidak. Bagus jika dipakai pada blok kondisi yang didalamnya ada banyak statement, kode akan terlihat lebih rapi dan mudah di-maintain.
Perhatikan kode berikut, bisa dilihat pada keyword `default` terdapat kurung kurawal yang mengapit 2 statement didalamnya.

Adapun contoh dari penggunaan kurung kurawal pada keyword case & default, bisa dilihat dan diimplementasikan.

```
var point = 6

switch point {
case 8:
    fmt.Println("perfect")
case 7, 6, 5, 4:
    fmt.Println("awesome")
default:
    {
        fmt.Println("not bad")
        fmt.Println("you can be better!")
    }
}
```

1.2.1.6 Switch Dengan Gaya if - else

Uniknya di Go, switch bisa digunakan dengan gaya ala if-else. Nilai yang akan dibandingkan tidak dituliskan setelah keyword `switch`, melainkan akan ditulis langsung dalam bentuk perbandingan dalam keyword `case`.
Pada kode di bawah ini, kode program switch di atas diubah ke dalam gaya `if-else`. Variabel `point` dihilangkan dari keyword `switch`, lalu kondisi-kondisinya dituliskan di tiap `case`.

```
var point = 6

switch {
case point == 8:
    fmt.Println("perfect")
case (point < 8) && (point > 3):
    fmt.Println("awesome")
default:
    {
        fmt.Println("not bad")
        fmt.Println("you need to learn more")
    }
}
```

1.2.1.7 Penggunaan Keyword fallthrough Dalam switch

Seperti yang sudah dijelaskan sebelumnya, bahwa switch pada Go memiliki perbedaan dengan bahasa lain. Ketika sebuah `case` terpenuhi, pengecekkan kondisi tidak akan diteruskan ke case-case setelahnya.

Keyword `fallthrough` digunakan untuk memaksa proses pengecekkan diteruskan ke `case` selanjutnya dengan **tanpa menghiraukan nilai kondisinya**, jadi case di pengecekan selanjutnya tersebut selalu dianggap benar (meskipun aslinya adalah salah).

```
var point = 6

switch {

case point == 8:

    fmt.Println("perfect")

case (point < 8) && (point > 3):

    fmt.Println("awesome")

    fallthrough

case point < 5:

    fmt.Println("you need to learn more")

default:

    {

        fmt.Println("not bad")

        fmt.Println("you need to learn more")

    }

}
```

Setelah pengecekkan `case (point < 8) && (point > 3)` selesai, akan dilanjut ke pengecekkan `case point < 5`, karena ada `fallthrough` di situ.

1.2.1.8 Seleksi Kondisi Bersarang

Seleksi kondisi bersarang adalah seleksi kondisi, yang berada dalam seleksi kondisi, yang mungkin juga berada dalam seleksi kondisi, dan seterusnya. Seleksi kondisi bersarang bisa dilakukan pada `if` - `else`, `switch`, ataupun kombinasi keduanya.

```
var point = 10

if point > 7 {
    switch point {
    case 10:
        fmt.Println("perfect!")
    default:
        fmt.Println("nice!")
    }
} else {
    if point == 5 {
        fmt.Println("not bad")
    } else if point == 3 {
        fmt.Println("keep trying")
    } else {
        fmt.Println("you can do it")
        if point == 0 {
            fmt.Println("try harder!")
        }
    }
}
```

1.3 Praktikum

1.3.1 Latihan Praktikum

```
package main

import (
     "fmt"
)

func main() {
     var numA, numB, numMenu int

     fmt.Println("========================")
     fmt.Print("Masukan Angka ke 1 : ")
     fmt.Scan(&numA)
     fmt.Print("Masukan Angka ke 2 : ")
     fmt.Scan(&numB)
     fmt.Println("========================")
     fmt.Println("Pilih menu dibawah :")
     fmt.Println("1. Penjumlahan")
     fmt.Println("2. Pengurangan")
     fmt.Println("========================")
     fmt.Print("Pilih menu angka diatas : ")
     fmt.Scan(&numMenu)

     if numMenu == 1 {

          fmt.Println("Hasil penjumlahan", numA, "+", numB,
"=", numA+numB)

     } else {

          fmt.Println("Hasil pengurangan", numA, "+", numB,
"=", numA-numB)
     }

}
```

Untuk latihan praktikum hari ini, silahkan menambah menu untuk perkalian dan pembagian pada program diatas dan membuat sebuah fungsi untuk mengulangi pilihan dari user untuk dapat memilih menu kembali.

1.3.2 Tugas Praktikum

Coming Soon