LAPORAN PRAKTIKUM

ALGORITMA DAN PEMROGRAMAN 1

MODUL 10

“SWITCH-CASE”

DISUSUN OLEH:

RAIHAN ADI ARBA

103112400071

S1 IF-12-01

DOSEN:

Yohani Setiya Rafika Nur, M. Kom.

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024/2025

DASAR TEORI

Switch-case dalam Golang merupakan struktur kontrol yang menyediakan cara efisien untuk mengevaluasi multiple kondisi dalam sebuah program. Struktur ini menjadi salah satu fitur penting dalam bahasa pemrograman Go yang memungkinkan pengembang untuk menulis kode yang lebih terorganisir dan mudah dipahami, terutama ketika berhadapan dengan banyak kondisi yang perlu dievaluasi. Berbeda dengan bahasa pemrograman lain, Golang memiliki implementasi switch-case yang lebih fleksibel dan powerful dengan fitur automatic break yang mencegah fall-through yang tidak diinginkan, sehingga mengurangi potensi bug dan meningkatkan keandalan kode.

Dalam penggunaannya, switch-case di Golang memiliki keunikan tersendiri karena dapat mengevaluasi berbagai tipe data dan kondisi, bahkan dapat berfungsi tanpa ekspresi eksplisit layaknya if-else chain dengan sintaks yang lebih bersih. Fleksibilitas ini memberikan kebebasan kepada pengembang untuk mengimplementasikan logika yang kompleks dengan cara yang lebih efisien. Switch-case juga mendukung multiple case values dalam satu case, memungkinkan pengelompokan beberapa nilai yang memerlukan penanganan yang sama, sehingga mengurangi duplikasi kode.

Salah satu fitur canggih dari switch-case dalam Golang adalah dukungan untuk type switch, yang memungkinkan pemeriksaan tipe data interface secara runtime. Ini sangat berguna dalam pemrograman berorientasi objek dan penanganan data dinamis. Meskipun Golang secara otomatis menambahkan break di setiap akhir case, bahasa ini juga menyediakan fitur fallthrough yang dapat digunakan jika diperlukan eksekusi case berikutnya. Namun, penggunaan fallthrough tidak umum dan harus digunakan dengan hati-hati karena dapat membuat alur program menjadi kurang jelas.

Struktur kontrol ini sangat berguna ketika kita perlu membandingkan satu variabel dengan beberapa nilai yang berbeda, membuat kode lebih mudah dibaca dan dipelihara dibandingkan dengan rangkaian if-else yang panjang. Dalam konteks performa, switch-case di Golang dioptimalkan oleh compiler untuk menghasilkan kode yang efisien, terutama ketika menangani banyak kondisi. Penggunaan switch-case yang tepat dapat meningkatkan efisiensi dan kejelasan kode, terutama dalam situasi yang memerlukan penanganan multiple kondisi atau pemeriksaan nilai-nilai spesifik.

Keunggulan lain dari switch-case dalam Golang adalah kemampuannya untuk menangani berbagai jenis ekspresi dan tipe data, termasuk string, integer, dan bahkan fungsi. Ini membuat switch-case menjadi alat yang sangat versatile dalam pengembangan aplikasi. Struktur ini juga mendukung penggunaan expression switch, di mana setiap case dapat berisi ekspresi boolean yang kompleks, memberikan fleksibilitas tambahan dalam implementasi logika bisnis yang rumit.

A. UNGUIDED

1. Latihan 1

Buatlah program dengan bahasa Go yang digunakan untuk menentukan apakah kadar pH pada air yang diinput termasuk air yang layak diminum atau tidak.

• Masukan terdiri dari satu nilai float, yaitu kadar pH.

• Keluaran berupa teks:

– "Air Layak Minum" (jika pH >= 6.5 dan pH <= 8.6),

– "Air Tidak Layak Minum" (jika pH < 6.5 ataupun pH > 8.6),

– "Input tidak valid, rentang pH 0 - 14" jika nilai pH melampaui 14 atau bernilai negatif.

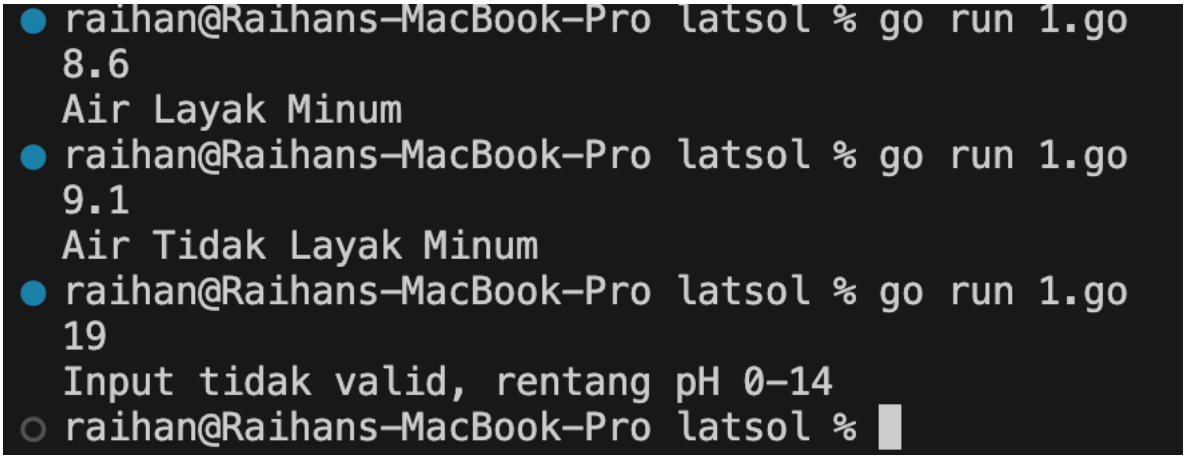
Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	8.6	Air layak minum
2	9	Air tidak layak minum
3	16	Nilai pH tidak valid. Nilai pH harus antara 0 dan 14.

Source Code:

```
 package main

import "fmt"

func main() {
	var pH float64
	fmt.Scan(&pH)

	switch {
	case pH < 0 || pH > 14:
		fmt.Println("Input tidak valid, rentang pH 0-14")
	case pH >= 6.5 && pH <= 8.6:
		fmt.Println("Air Layak Minum")
	case pH < 6.5 || pH > 8.6:
		fmt.Println("Air Tidak Layak Minum")
	}
}
```

Output :



Penjelasan Program:

Program ini dibuat untuk mengecek kelayakan air minum berdasarkan nilai pH yang dimasukkan oleh pengguna. Program dimulai dengan mendeklarasikan variabel pH bertipe float64 yang akan menyimpan input dari pengguna melalui fungsi fmt.Scan(). Selanjutnya, program menggunakan struktur switch-case tanpa ekspresi untuk melakukan evaluasi terhadap tiga kondisi berbeda. Pertama, program akan memeriksa apakah nilai pH berada di luar rentang valid (0-14), jika benar maka akan menampilkan pesan "Input tidak valid, rentang pH 0-14". Kedua, program memeriksa apakah nilai pH berada dalam rentang air layak minum (6.5-8.6), jika benar maka akan menampilkan "Air Layak Minum". Ketiga, jika nilai pH berada di luar rentang air layak minum namun masih dalam rentang valid, program akan menampilkan "Air Tidak Layak Minum". Hal ini dibuktikan dengan beberapa contoh output dimana input 8.6 menghasilkan "Air Layak Minum", input 9.1 menghasilkan "Air Tidak Layak Minum", dan input 19 menghasilkan "Input tidak valid, rentang pH 0-14".

2. Latihan 2

Buatlah program dalam bahasa Go untuk menghitung tarif parkir berdasarkan jenis kendaraan dan durasi parkir yang dimasukkan oleh pengguna.

Ada tiga jenis kendaraan:
– motor,
– mobil,
– truk.Masing-masing jenis kendaraan memiliki tarif parkir yang berbeda: • Motor: Rp 2.000 per jam
• Mobil: Rp 5.000 per jam
• Truk: Rp 8.000 per jam

Program harus:

• Menentukan tarif per jam berdasarkan jenis kendaraan.
• Mengalikan tarif tersebut dengan jumlah jam parkir untuk mendapatkan total biaya.
• Jika durasi parkir kurang dari 1 jam, maka durasi dianggap 1 jam penuh.

Masukan : berupa jenis kendaraan dan durasi parkir (dalam jam).
Keluaran : berupa total biaya parkir berdasarkan jenis kendaraan dan durasi parkir.

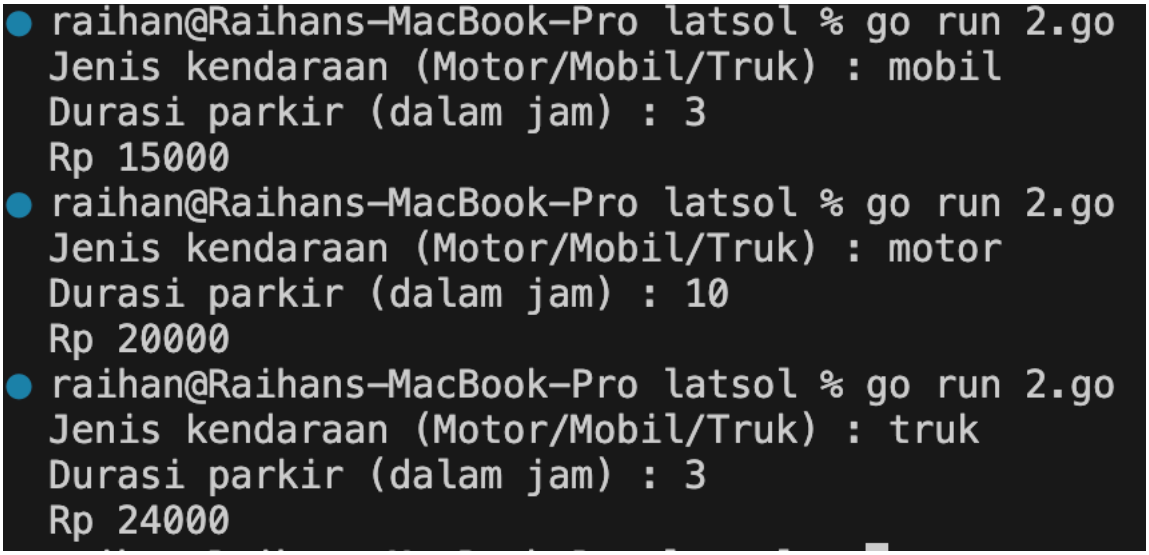
Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	motor 3 jam	Rp 6000
2	mobil 1 jam	Rp 5000
3	truk 5 jam	Rp 40000

Source Code :

```
package main

import "fmt"

func main() {
	var durasi, tarif int
	var tipe string

	fmt.Print("Jenis kendaraan (Motor/Mobil/Truk) : ")
	fmt.Scan(&tipe)
	fmt.Print("Durasi parkir (dalam jam) : ")
	fmt.Scan(&durasi)

	switch {
	case tipe == "mobil":
		tarif = 5000 * durasi
	case tipe == "motor":
		tarif = 2000 * durasi
	case tipe == "truk":
		tarif = 8000 * durasi
	default:
		fmt.Println("Jenis kendaraan atau durasi parkir tidak valid")
	}
	fmt.Println("Rp", tarif)
}
```

Output :



Pembahasan :

Program ini dirancang untuk menghitung biaya parkir berdasarkan jenis kendaraan dan durasi parkir. Program dimulai dengan mendeklarasikan tiga variabel: durasi dan tarif bertipe integer, serta tipe bertipe string. Program meminta input dari pengguna berupa jenis kendaraan (Motor/Mobil/Truk) dan durasi parkir dalam jam menggunakan fmt.Print() untuk menampilkan prompt dan fmt.Scan() untuk menerima input. Selanjutnya, program menggunakan struktur switch-case untuk menentukan tarif berdasarkan jenis kendaraan: mobil dikenakan tarif Rp 5.000 per jam, motor Rp 2.000 per jam, dan truk Rp 8.000 per jam. Jika jenis kendaraan yang dimasukkan tidak sesuai dengan pilihan yang tersedia, program akan menampilkan pesan "Jenis kendaraan atau durasi parkir tidak valid". Akhirnya, program menampilkan total biaya parkir dengan menampilkan "Rp" diikuti nilai tarif yang telah dihitung berdasarkan durasi parkir dikalikan dengan tarif per jam sesuai jenis kendaraan.

3. Latihan 3

Buatlah program dengan bahasa Go yang digunakan untuk mengidentifikasi pola aritmatika berdasarkan bilangan yang diinputkan dan melakukan operasi matematika yang sesuai. Beberapa ketentuan kategori di antaranya:
a) Bilangan Ganjil: Menghitung penjumlahan antara bilangan yang diinput dengan bilangan berikutnya.
b) Bilangan Genap: Menghitung perkalian antara bilangan yang diinput dengan bilangan berikutnya.
c) Bilangan Kelipatan 5: Menghitung hasil kuadrat dari bilangan yang diinputkan.
d) Bilangan Kelipatan 10: Membagi bilangan yang diinputkan dengan bilangan 10.• Masukan terdiri dari satu bilangan bulat.
• Keluaran:
– "Kategori: Bilangan Ganjil" diikuti dengan "Hasil penjumlahan dengan bilangan berikutnya %(input) + %(input+1) = %(hasil)" untuk Bilangan Ganjil.

– "Kategori: Bilangan Genap" diikuti dengan "Hasil perkalian dengan bilangan berikutnya %(input) * %(input+1) = %(hasil)" untuk Bilangan Genap.
– "Kategori: Bilangan Kelipatan 5" diikuti dengan "Hasil kuadrat dari %(input)^2 = %(hasil)" untuk Bilangan Kelipatan 5.
– "Kategori: Bilangan Kelipatan 10" diikuti dengan "Hasil pembagian antara %(input) / 10 = %(hasil)" untuk Bilangan Kelipatan 10.
Contoh masukan dan keluaran:

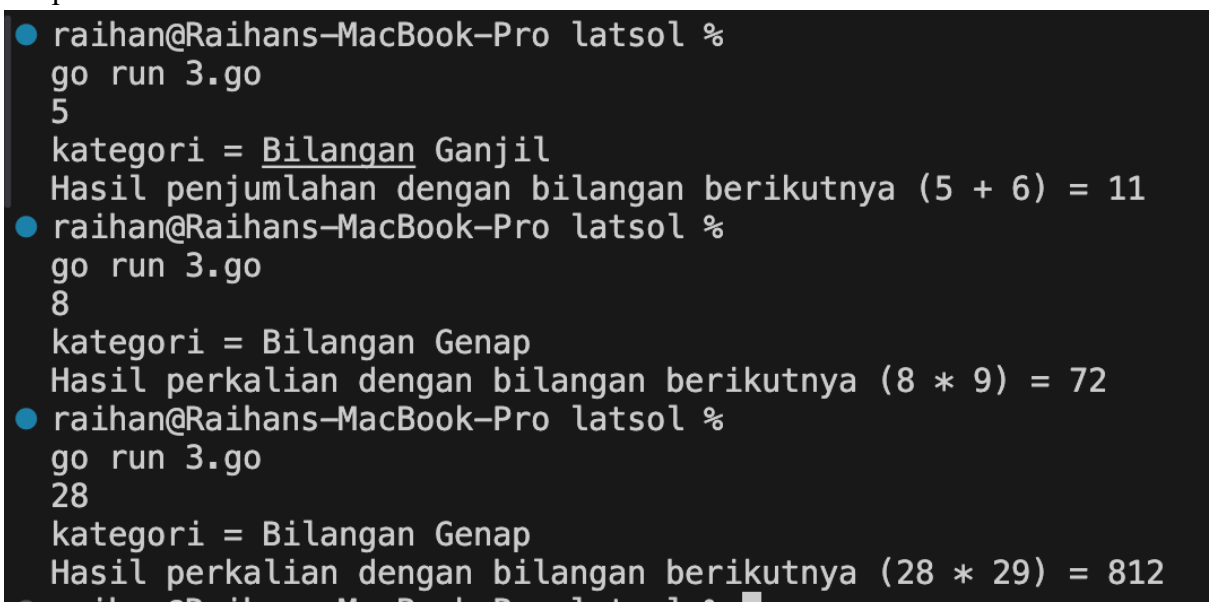
No	Masukan	Keluaran
1	5	Kategori: Bilangan Ganjil Hasil penjumlahan dengan bilangan berikutnya 5 + 6 = 11
2	8	Kategori: Bilangan Genap Hasil perkalian dengan bilangan berikutnya 8 * 9 = 72
3	25	Kategori: Bilangan Kelipatan 5 Hasil kuadrat dari 25 ^2 = 625
4	20	Kategori: Bilangan Kelipatan 10 Hasil pembagian antara 20 / 10 = 2

Source Code:

```
package main

import "fmt"

func main() {
	var input, hasil int
	fmt.Scan(&input)

	switch {
	case input%10 == 0:
		fmt.Println("kategori = Bilangan Kelipatan 10")
		hasil = input / 10
		fmt.Printf("Hasil pembagian antara (%d / 10) = %d\n", input, hasil)
	case input%5 == 0 && input != 5:
		fmt.Println("kategori = Bilangan Kelipatan 5")
		hasil = input * input
		fmt.Printf("Hasil kuadrat dari  (%d ^ 2) = %d\n", input, hasil)
	case input%2 == 0:
		hasil = input * (input + 1)
		fmt.Println("kategori = Bilangan Genap")
		fmt.Printf("Hasil perkalian dengan bilangan berikutnya (%d * %d) =
%d\n", input, input+1, hasil)
	case input%2 != 0:
		hasil = input + (input + 1)
		fmt.Println("kategori = Bilangan Ganjil")
```

```
                fmt.Printf("Hasil penjumlahan dengan bilangan berikutnya (%d + %d)
= %d\n", input, input+1, hasil)
        }
}
```

Output:



Deskripsi Program:

Program untuk mengklasifikasikan bilangan bulat yang diinputkan pengguna berdasarkan sifatnya dan menghitung hasil operasi tertentu sesuai dengan kategorinya. Jika bilangan merupakan kelipatan 10, program akan membaginya dengan 10 dan menampilkan hasil pembagiannya. Untuk bilangan yang merupakan kelipatan 5 tetapi bukan 5, program akan menghitung kuadrat bilangan tersebut dan menampilkan hasilnya. Jika bilangan adalah bilangan genap, program akan mengalikan bilangan tersebut dengan bilangan berikutnya, sementara untuk bilangan ganjil, program akan menjumlahkannya dengan bilangan berikutnya. Hasil perhitungan ini kemudian ditampilkan bersama dengan kategori bilangan yang sesuai. Program ini dirancang untuk memberikan informasi yang jelas dan hasil perhitungan yang sesuai berdasarkan sifat bilangan yang diinputkan.

DAFTAR PUSTAKA

Prayogo, N. A. (2021). ***Dasar Pemrograman Go. Ebook***