

LAPORAN PRAKTIKUM ALGORITMA

DAN PEMROGRAMAN 1

MODUL 11

SWITCH-CASE



Disusun oleh:

MUHAMMAD FIRDAUS ARDIANSYAH

109082500126

S1IF-13-07

Asisten Praktikum

Adithana dharma putra

Apri pandu wicaksono

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

LATIHAN KELAS – GUIDED

1. Guided 1

Source Code

```
package main

import "fmt"

func main() {
    var jam12, jam24 int
    var label string

    if _, err := fmt.Scan(&jam24); err != nil {
        fmt.Println("failed to read input:", err)
        return
    }

    switch {
    case jam24 == 0:
        jam12 = 12
        label = "AM"
    case jam24 < 12:
        jam12 = jam24
        label = "AM"
    case jam24 == 12:
        jam12 = 12
        label = "PM"
    case jam24 > 12 && jam24 < 24:
        jam12 = jam24 - 12
    }
}
```

```

        label = "PM"

    default:

        fmt.Println("invalid hour")

    return

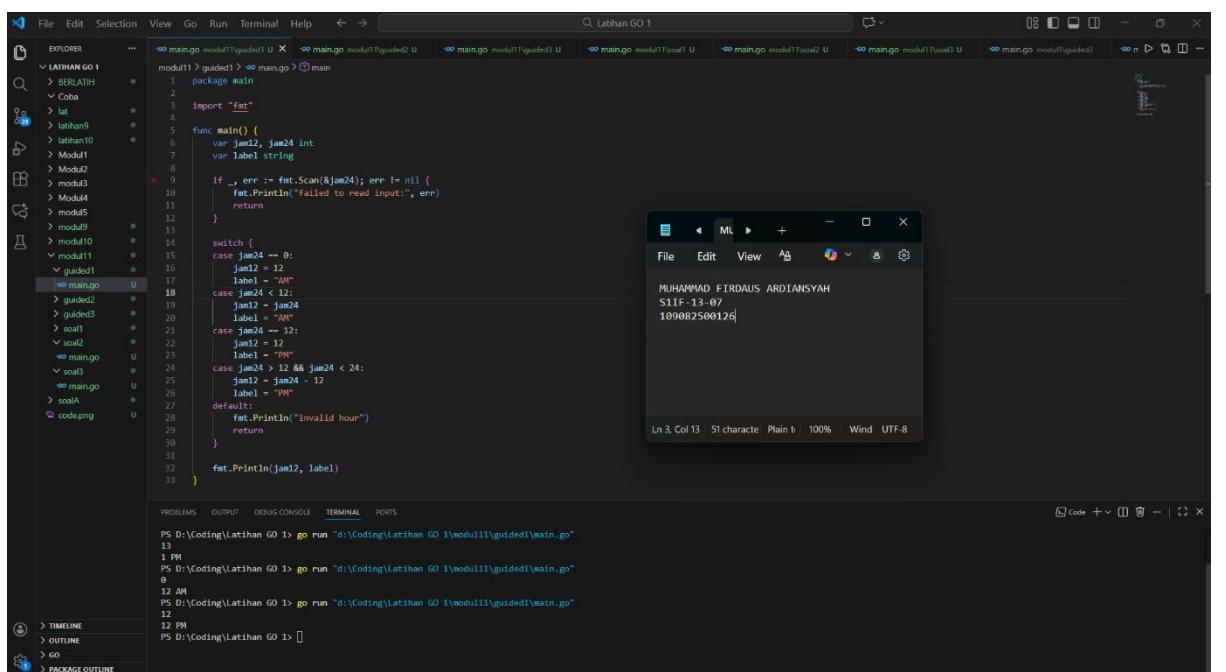
}

fmt.Println(jam12, label)

}

```

Screenshot program



Deskripsi program

Program ini bertujuan untuk mengubah format jam dari sistem 24 jam (0-23) menjadi format 12 jam (AM/PM). Program mendeklarasikan variabel jam24 untuk menampung input kita, serta jam12 dan label untuk menyimpan hasil konversi. Terdapat juga mekanisme *error handling* saat proses fmt.Scan; jika input yang dimasukkan bukan angka yang valid, program akan mencetak pesan kesalahan dan menghentikan eksekusi menggunakan perintah return.

Logika inti konversi menggunakan struktur kontrol switch tanpa ekspresi (*tagless switch*), yang berfungsi mirip dengan rangkaian if-else. Blok case menangani berbagai skenario waktu: mengubah jam 0 menjadi 12 AM, mempertahankan angka

untuk jam di bawah 12 (AM), menetapkan jam 12 sebagai PM, dan melakukan operasi aritmatika (mengurangi 12) untuk jam di atas 12 agar menjadi format PM. Jika input berada di luar rentang 0-23, blok default akan aktif untuk memberitahu bahwa jam tidak valid.

2. Guided 2

Source Code

```
package main

import "fmt"

func main() {
    var nama_tanaman string
    fmt.Scan(&nama_tanaman)

    switch nama_tanaman {
        case "nepenthes", "drosera":
            fmt.Println("Termasuk Tanaman Karnivora.")
            fmt.Println("Asli Indonesia.")
        case "venus", "sarracenia":
            fmt.Println("Termasuk Tanaman Karnivora.")
            fmt.Println("Tidak Asli Indonesia.")
        default:
            fmt.Println("Tidak termasuk Tanaman Karnivora.")
    }
}
```

Screenshot program

The screenshot shows a Go development environment with several windows:

- EXPLORER**: Shows a file tree with folders like `LATIHAN GO 1`, `BERLATIH`, and `Coba`. Under `modul11\guided2`, there are files `main.go`, `guided3.go`, `soal1.go`, `soal2.go`, `soal3.go`, and `soalA.go`.
- CODE**: Displays the `main.go` file content:

```

package main

import "fmt"

func main() {
    var nama_tanaman string
    fmt.Scan(&nama_tanaman)

    switch nama_tanaman {
    case "nepenthes", "drosera":
        fmt.Println("Termasuk Tanaman Karnivora.")
        fmt.Println("Asli Indonesia.")
    case "venus", "sarracenia":
        fmt.Println("Termasuk Tanaman Karnivora.")
        fmt.Println("Tidak Asli Indonesia.")
    default:
        fmt.Println("Tidak termasuk Tanaman Karnivora.")
    }
}

```
- TERMINAL**: Shows command-line output for running the program with different inputs:

```

PS D:\Coding\Latihan GO 1> go run "d:\Coding\Latihan GO 1\modul11\guided2\main.go"
nepenthes
Termasuk Tanaman Karnivora.
Asli Indonesia.
PS D:\Coding\Latihan GO 1> go run "d:\Coding\Latihan GO 1\modul11\guided2\main.go"
venus
Termasuk Tanaman Karnivora.
Tidak Asli Indonesia.
PS D:\Coding\Latihan GO 1> go run "d:\Coding\Latihan GO 1\modul11\guided2\main.go"
Karedok
Tidak termasuk Tanaman Karnivora.
PS D:\Coding\Latihan GO 1>

```
- OUTPUT**: Shows the output of the program execution.
- DEBUG CONSOLE**: Shows the output of the debug console.
- PROBLEMS**: Shows no problems.
- OUTPUT**: Shows the output of the terminal.
- DEBUG CONSOLE**: Shows the output of the debug console.
- TERMINAL**: Shows the terminal interface with the user's name and student ID.
- PORTS**: Shows the network ports.

Deskripsi program

Program ini fokus utamanya adalah mengidentifikasi jenis dan asal usul tanaman berdasarkan input nama yang diberikan oleh kita. Di dalam fungsi `main`, program mendeklarasikan variabel `nama_tanaman` bertipe `string`, kemudian menggunakan fungsi `fmt.Scan` untuk menangkap input teks dari terminal dan menyimpannya ke dalam variabel tersebut.

Logika pengelompokan diimplementasikan menggunakan struktur kontrol `switch` yang memeriksa nilai variabel `nama_tanaman`. Keunikan kode ini terletak pada penggunaan multiple values in case, di mana satu blok `case` dapat menangani beberapa kondisi string sekaligus (seperti `"nepenthes"` dan `"drosera"` dalam satu grup). Program membagi output menjadi tiga kemungkinan: tanaman karnivora asli Indonesia, tanaman karnivora non-lokal (seperti `"venus"` atau `"sarracenia"`), dan blok `default` untuk menangani input tanaman lain yang tidak terdaftar sebagai karnivora.

3. Guided 3

Source Code

```

package main

import "fmt"

func main() {

    var kendaraan string

    var lamaParkir, biaya int
}

```

```
fmt.Print("Masukkan jenis kendaraan  
(mobil/motor/truk) : ")  
  
fmt.Scan(&kendaraan)  
  
fmt.Print("Masukkan lama parkir (dalam jam) : ")  
  
fmt.Scan(&lamaParkir)  
  
switch kendaraan {  
  
case "motor":  
  
    if lamaParkir <=2 {  
  
        biaya = 7000  
  
        fmt.Println("Biaya parkir: Rp", biaya)  
  
    } else {  
  
        biaya = 9000  
  
        fmt.Println("Biaya parkir: Rp", biaya)  
  
    }  
  
case "mobil":  
  
    if lamaParkir <=2 {  
  
        biaya = 15000  
  
        fmt.Println("Biaya parkir: Rp", biaya)  
  
    } else {  
  
        biaya = 20000  
  
        fmt.Println("Biaya parkir: Rp", biaya)  
  
    }  
  
case "truk":  
  
    if lamaParkir <=2 {  
  
        biaya = 25000  
  
        fmt.Println("Biaya parkir: Rp", biaya)
```

```

        } else {

            biaya = 35000

            fmt.Println("Biaya parkir: Rp", biaya)

        }

        default:

            fmt.Println("Jenis kendaraan atau durasi tidak
valid.")

            fmt.Println("Biaya parkir: Rp 0")

    }

}

```

Screenshot program

The screenshot shows a Go development environment with the following details:

- Code Editor:** The main window displays the source code for a Go program named `modul11.go`. The code defines a function `BiayaParkir` that takes a string `JenisKendaraan` and an integer `IamaParkir` as parameters. It calculates the parking fee based on the duration and vehicle type.
- Terminal:** A terminal window titled "Latihan GO 1" shows the command `go run "D:\Coding\Latihan\001\modul11\guide1\main.go"` being run, followed by the output: "Biaya parkir: Rp 7000".
- Modal Window:** A modal dialog box is open, displaying student information: MUHAMMAD FIRDAUS ARDIANSYAH, S11F-13-07, and 109082500126.
- Bottom Status Bar:** Shows the current file is `Ln 3, Col 13`, character count is `51`, and the font size is `140%`.

Deskripsi program

Program ini meminta kita untuk memasukkan dua data utama: jenis kendaraan (sebagai *string*) dan lama durasi parkir dalam satuan jam (sebagai *integer*). Variabel biaya disiapkan untuk menampung nominal harga yang harus dibayarkan berdasarkan aturan yang berlaku.

Logika perhitungan tarif menggunakan struktur kontrol bertingkat (*nested control structures*). Di bagian luar, program menggunakan switch untuk menyeleksi kategori kendaraan (motor, mobil, atau truk). Kemudian, di dalam setiap blok case, terdapat percabangan if-else untuk membedakan tarif berdasarkan durasi: tarif dasar untuk parkir ≤ 2 jam, dan tarif berbeda (lebih mahal) jika parkir lebih dari 2 jam. Penting dicatat bahwa skema tarif di sini bersifat *flat* (harga tetap) untuk durasi di atas 2 jam, bukan dihitung secara akumulatif per jam.

TUGAS

1. Tugas 1

Source code

```
package main

import "fmt"

func main() {
    var pH float64

    fmt.Print("Masukkan pH air: ")
    fmt.Scan(&pH)

    switch {
        case pH < 0 || pH > 14:
            fmt.Println("Nilai pH tidak valid. Nilai pH harus antara 0 dan 14")
        case pH >= 6.5 && pH <= 8.6:
            fmt.Println("Air Layak Minum")
        default:
```

```

        fmt.Println("Air Tidak Layak Minum")

    }

}

```

Screenshot program

The screenshot shows a Go development environment with the following details:

- File Explorer:** Shows a project structure under "LATIHAN GO 1" with files like main.go, modul1, modul2, modul3, modul4, modul5, modul9, modul10, modul11, guided1, guided2, guided3, and soal1.
- Code Editor:** Displays the content of main.go:

```

package main
import "fmt"
func main() {
    var pH float64
    fmt.Print("Masukkan pH air: ")
    fmt.Scan(&pH)
    switch {
    case pH < 0 || pH > 14:
        fmt.Println("Nilai pH tidak valid. Nilai pH harus antara 0 dan 14")
    case pH >= 6.5 && pH <= 8.6:
        fmt.Println("Air Layak Minum")
    default:
        fmt.Println("Air Tidak Layak Minum")
    }
}

```
- Terminal:** Shows the command "go run" being executed in the terminal, followed by user input "Masukkan pH air: 8.6" and the output "Air Layak Minum".
- Output Panel:** Shows the command "go run" being executed again, followed by user input "Masukkan pH air: 9" and the output "Air Tidak Layak Minum".
- Code Outline:** Shows the outline of the main.go file with sections for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS.

Deskripsi program

Program ini fungsinya adalah mengevaluasi kelayakan konsumsi air berdasarkan input tingkat keasaman (pH). Variabel pH dideklarasikan dengan tipe data float64 agar mampu menerima angka desimal yang presisi (seperti 6.5), di mana inputnya diambil dari kita menggunakan fungsi fmt.Scan.

Logika pengambilan keputusan menggunakan struktur switch tanpa ekspresi (*tagless switch*) yang fleksibel. Blok pertama (case) bertugas memvalidasi angka menggunakan operator logika OR (||) untuk menolak nilai yang tidak masuk akal (di bawah 0 atau di atas 14). Blok kedua menggunakan operator AND (&&) untuk menetapkan rentang aman (antara 6.5 sampai 8.6) sebagai "Air Layak Minum". Jika input valid namun tidak masuk dalam rentang aman tersebut (misalnya terlalu asam atau terlalu basa), program akan menjalankan blok default dan menyatakan air tidak layak minum.

2. Tugas 2

Source code

```
package main

import "fmt"

func main() {
    var durasiParkir int
    var jenisKendaraan string
    var biayaParkir int

    fmt.Print("Masukkan jenis kendaraan
(mobil/motor/truk): ")

    fmt.Scan(&jenisKendaraan)

    fmt.Print("Masukkan durasi parkir (dalam jam): ")

    fmt.Scan(&durasiParkir)

    switch jenisKendaraan {
        case "motor":
            biayaParkir = 2000
            if durasiParkir <= 1 {
                fmt.Println("Biaya parkir: Rp", biayaParkir
                * durasiParkir)
            } else {
                fmt.Println("Biaya parkir: Rp", biayaParkir
                * durasiParkir)
            }
        case "mobil":
            biayaParkir = 5000
            if durasiParkir <= 1 {
                fmt.Println("Biaya parkir: Rp", biayaParkir
                * durasiParkir)
            }
    }
}
```

```

        } else {

            fmt.Println("Biaya parkir: Rp", biayaParkir
            * durasiParkir)

        }

    case "truk":

        biayaParkir = 8000

        if durasiParkir <= 1 {

            fmt.Println("Biaya parkir: Rp", biayaParkir
            * durasiParkir)

        } else {

            fmt.Println("Biaya parkir: Rp", biayaParkir
            * durasiParkir)

        }

    }

}

```

Screenshot program

The screenshot shows a Go development environment with the following details:

- File Explorer (EXPLORER):** Shows the project structure with files like `main.go`, `modul11\soal2\main.go`, and `modul11\soal2\main.go`.
- Code Editor:** Displays the content of `main.go` which contains a function `main()` that prints parking fees based on vehicle type and duration.
- Terminal (LATHAN GO 1):**
 - Shows the command `PS D:\Coding\LATHAN GO 1> go run "d:\Coding\LATHAN GO 1\modul11\soal2\main.go"`
 - Asks for vehicle type: `Masukkan jenis kendaraan (mobil/motor/truk): motor`
 - Asks for duration: `Masukkan durasi parkir (dalam jam): 3`
 - Outputs the result: `Biaya parkir: Rp 12000`
- Output Window:** Shows the output of the program execution.
- Code Editor (right side):** Displays a snippet of code with student information:


```

MUHAMMAD FIRDAUS ARDIANSYAH
S1TF-13-07
109082500126
      
```

Deskripsi program

Program ini merupakan kalkulator biaya parkir berbasis tarif per jam, program meminta input dari kita berupa jenis kendaraan (string) dan durasi parkir dalam jam (integer). Variabel biayaParkir digunakan untuk menyimpan harga dasar per jam yang berbeda-beda tergantung jenis kendaraannya.

Logika program menggunakan struktur switch untuk memisahkan kategori kendaraan. Untuk "motor" tarifnya 2.000, "mobil" 5.000, dan "truk" 8.000. Di dalam setiap blok case, terdapat percabangan if-else yang mengecek apakah durasi parkir kurang dari atau sama dengan 1 jam. Namun, jika diperhatikan dengan saksama, rumus perhitungan di dalam blok if maupun else adalah identik (biayaParkir * durasiParkir). Artinya, program ini menghitung biaya secara linier (tarif dikali jam) tanpa membedakan apakah parkir itu 1 jam pertama atau jam-jam berikutnya.

3. Tugas 3

Source code

```
package main

import "fmt"

func main() {
    var num int
    fmt.Print("Masukkan bilangan: ")
    fmt.Scan(&num)

    switch {
    case num == 5:
        hasil := num + (num + 1)
        fmt.Println("Kategori: Bilangan Ganjil")
        fmt.Printf("Hasil penjumlahan dengan bilangan
berikutnya %d + %d = %d\n", num, num+1, hasil)
    case num%10 == 0:
        hasil := num / 10
        fmt.Println("Kategori: Bilangan Kelipatan 10")
        fmt.Printf("Hasil pembagian antara %d / 10 =
%d\n", num, hasil)
    case num%5 == 0:
```

```

        hasil := num * num

        fmt.Println("Kategori: Bilangan Kelipatan 5")
        fmt.Printf("Hasil kuadrat dari %d ^2 = %d\n",
num, hasil)

    case num%2 == 0:

        hasil := num * (num + 1)
        fmt.Println("Kategori: Bilangan Genap")
        fmt.Printf("Hasil perkalian dengan bilangan
berikutnya %d * %d = %d\n", num, num+1, hasil)

    default:

        hasil := num + (num + 1)
        fmt.Println("Kategori: Bilangan Ganjil")
        fmt.Printf("Hasil penjumlahan dengan bilangan
berikutnya %d + %d = %d\n", num, num+1, hasil)

    }
}

```

Screenshot program

The screenshot shows a Go development environment with the following details:

- Code Editor:** The main editor window displays the source code for a Go program named `main.go`. The code uses a `switch` statement to categorize a number based on its remainder when divided by 2. It handles cases for odd numbers (Ganjil), even numbers (Genap), and multiples of 5 (Kelipatan 5). The code also includes comments and imports for `fmt`.
- Terminal:** A separate terminal window shows the execution of the program. It prompts the user for a number, then prints the category and performs the specified calculation.
- Output:** The terminal output is as follows:

```

PS D:\Coding\Latihan GO 1> go run "d:\Coding\Latihan GO 1\modulIII\soal3\main.go"
Masukkan bilangan: 5
Kategori: Bilangan Ganjil
Hasil penjumlahan dengan bilangan berikutnya 5 + 6 = 11
PS D:\Coding\Latihan GO 1> go run "d:\Coding\Latihan GO 1\modulIII\soal3\main.go"
Masukkan bilangan: 8
Kategori: Bilangan Genap
Hasil perkalian dengan bilangan berikutnya 8 * 9 = 72
PS D:\Coding\Latihan GO 1> go run "d:\Coding\Latihan GO 1\modulIII\soal3\main.go"
Masukkan bilangan: 25
Kategori: Bilangan Kelipatan 5
Hasil kuadrat dari 25 ^2 = 625
PS D:\Coding\Latihan GO 1>

```

Deskripsi program

Program ini adalah aplikasi pengolah bilangan bulat yang melakukan operasi aritmatika berbeda tergantung pada kategori angka yang diinputkan. Program meminta satu input *integer* num, lalu mengevaluasinya menggunakan struktur kontrol switch tanpa kondisi (*tagless switch*). Setiap kategori memiliki rumus

perhitungan unik, seperti penjumlahan, pembagian, pengkuadratan, atau perkalian, yang hasilnya kemudian diformat dan ditampilkan menggunakan `fmt.Println`.

Logika program ini sangat bergantung pada urutan prioritas pengecekan di dalam blok `switch`. Program mengecek kondisi dari atas ke bawah: pertama apakah angka tersebut spesifik 5, lalu apakah kelipatan 10, kemudian kelipatan 5, dan terakhir apakah angka tersebut genap (kelipatan 2). Karena sifat `switch` di Go yang otomatis berhenti setelah menemukan kondisi yang benar, angka seperti 20 (yang sebenarnya memenuhi syarat kelipatan 10, kelipatan 5, dan genap) hanya akan dieksekusi oleh `case "Kelipatan 10"` saja, sedangkan kondisi di bawahnya akan diabaikan. Angka yang tidak memenuhi semua kriteria tersebut akan masuk ke blok `default` dan dikategorikan sebagai bilangan ganjil.