

LAPORAN PRAKTIKUM ALGORITMA

DAN PEMROGRAMAN 1

MODUL 11

SWITCH-CASE



Disusun oleh:

Didi Hermawanto

109082500088

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

    fmt.Scan(&jam24)

    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:
            jam12 = jam24 - 12
            label = "PM"
    }

    fmt.Println(jam12, label)
}
```

Screenshot program

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

- Code Editor:** A file named `modul1.go` is open, containing Go code for a program that reads a 24-hour time input and outputs it in AM/PM format.
- Terminal:** The terminal window shows the command `go run "c:\coding\didid\golang\Laprak 11\modul1.go"` being run three times, resulting in the output "1 PM", "0", and "12 PM".
- Status Bar:** The status bar at the bottom right indicates the current line (Ln 3), column (Col 23), character count (60), encoding (Plain t), zoom level (100%), window size (Wind), and file encoding (UTF-8).

Deskripsi program

Program ini membaca sebuah angka dari pengguna sebagai jam dalam format 24 jam. Dua variabel disiapkan di awal, yaitu `jam12` untuk menampung hasil jam dalam format 12 jam dan `label` untuk menyimpan informasi AM atau PM. Setelah nilai `jam24` dibaca, program masuk ke bagian `switch` tanpa ekspresi, yang berarti setiap `case` akan dicek berdasarkan kondisi yang bernilai benar. Pada kondisi pertama, jika `jam24` bernilai 0, program mengubahnya menjadi jam 12 dan memberi label AM karena pukul 00:00 dalam format 24 jam setara dengan 12 AM. Jika angkanya kurang dari 12, jam tersebut langsung dipakai tanpa perubahan dan tetap diberi label AM. Bila nilai yang masuk tepat 12, program menganggapnya sebagai 12 PM. Sedangkan untuk nilai lebih dari 12, jamnya dikurangi 12 agar sesuai format 12 jam dan ditandai sebagai PM. Setelah kondisi yang sesuai ditemukan, program menampilkan hasil akhir berupa jam dalam format 12 jam beserta label AM atau PM. Alur kerja programnya sangat sederhana : baca input, cocokkan kondisi waktu, tentukan label, tampilkan hasil konversi.

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("Bukan Asli Indonesia")
        default:
            fmt.Println("Bukan Termasuk Tanaman
Karnivora")
        }
    }
}

```

Screenshot program

The screenshot shows a code editor interface with two tabs: 'modul1.go' and 'modul2.go'. The 'modul2.go' tab is active, displaying the Go code provided above. A terminal window is open at the bottom, showing the execution of the program. The output of the program is displayed in a separate window, showing the classification of various plants based on user input.

NIM	:109082500088
KELAS	:S1IF-13-07
NAMA	:Didi Hermawanto

Terminal Output:

```

nepenthes
Termasuk Tanaman Karnivora
Asli Indonesia
PS C:\coding didi\golang\Laprak 11> go run "c:\coding didi\golang\Laprak 11\modul2.go"
venus
Termasuk Tanaman Karnivora
Bukan Asli Indonesia
PS C:\coding didi\golang\Laprak 11> go run "c:\coding didi\golang\Laprak 11\modul2.go"
karedok
Bukan Termasuk Tanaman Karnivora

```

Deskripsi program

Program ini digunakan untuk mengenali apakah sebuah tanaman termasuk tanaman karnivora dan sekaligus menentukan apakah tanaman tersebut berasal dari Indonesia atau bukan. Setelah pengguna mengetikkan nama tanaman, program langsung

memeriksa nilai input tersebut melalui struktur *switch*. Pada pilihan pertama, jika nama yang dimasukkan adalah “nepenthes” atau “drosera”, program menampilkan bahwa tanaman tersebut merupakan tanaman karnivora dan berasal dari Indonesia. Pada pilihan kedua, jika nama yang diberikan adalah “venus” atau “sarracenia”, program tetap menganggapnya sebagai tanaman karnivora, tetapi bukan tanaman asli Indonesia. Jika nama tanaman tidak cocok dengan semua pilihan yang ada, program langsung masuk ke bagian *default* dan memberi informasi bahwa tanaman tersebut bukan termasuk tanaman karnivora. Alur program berjalan ketika: pengguna memasukkan nama → program mencocokkan dengan daftar kasus → menentukan kategori → menampilkan hasil.

3. Guided 3

Source Code

```
package main

import "fmt"

func main() {
    var kendaraan string
    var durasi int
    var tarif int

    fmt.Print("Masukkan jenis kendaraan
(Motor/Mobil/Truk): ")

    fmt.Scan(&kendaraan)

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

    switch {
        case kendaraan == "Motor" && durasi >= 1 &&
durasi <= 2:
            tarif = 7000
        case kendaraan == "Motor" && durasi > 2:
            tarif = 9000
        case kendaraan == "Mobil" && durasi >= 1 &&
durasi <= 2:
            tarif = 15000
    }
}
```

```

        case kendaraan == "Mobil" && durasi > 2:
            tarif = 20000

        case kendaraan == "Truk" && durasi >= 1 &&
durasi <= 2:
            tarif = 25000

        case kendaraan == "Truk" && durasi > 2:
            tarif = 35000

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

    }

    fmt.Printf("Tarif parkir untuk %s selama %d jam
adalah: Rp%d\n", kendaraan, durasi, tarif)

}

```

Screenshot program

The screenshot shows a code editor interface with two tabs: `modul1.go` and `modul3.go`. The `modul3.go` tab is active and contains the following Go code:

```

func main() {
    var kendaraan string
    var durasi int

    fmt.Print("Masukkan jenis kendaraan (Motor/Mobil/Truk): ")
    fmt.Scan(&kendaraan)
    fmt.Print("Masukkan durasi parkir (dalam jam): ")
    fmt.Scan(&durasi)

    switch {
    case kendaraan == "Motor" && durasi >= 1 && durasi <= 2:
        tarif = 7000
    case kendaraan == "Motor" && durasi > 2:
        tarif = 9000
    case kendaraan == "Mobil" && durasi >= 1 && durasi <= 2:
        tarif = 15000
    case kendaraan == "Mobil" && durasi > 2:
        tarif = 20000
    }
}

func printTarif(kendaraan string, durasi int) {
    fmt.Printf("Tarif parkir untuk %s selama %d jam adalah: Rp%d\n", kendaraan, durasi, tarif)
}

```

To the right of the code editor is a terminal window showing the execution of the program. The user inputs the type of vehicle and parking duration, and the program outputs the calculated fare.

NIM	:109082500088
KELAS	:S1IF-13-07
NAMA	:Didi Hermawanto

Terminal Output:

```

PS C:\coding\didil\golang\Laprak 11> go run "c:\coding\didil\golang\Laprak 11\modul3.go"
● Masukkan jenis kendaraan (Motor/Mobil/Truk): Motor
Masukkan durasi parkir (dalam jam): 2
Tarif parkir untuk Motor selama 2 jam adalah: Rp7000
● PS C:\coding\didil\golang\Laprak 11> go run "c:\coding\didil\golang\Laprak 11\modul3.go"
Masukkan jenis kendaraan (Motor/Mobil/Truk): Mobil
Masukkan durasi parkir (dalam jam): 4
Tarif parkir untuk Mobil selama 4 jam adalah: Rp20000
● PS C:\coding\didil\golang\Laprak 11> go run "c:\coding\didil\golang\Laprak 11\modul3.go"
Masukkan jenis kendaraan (Motor/Mobil/Truk): Truk

```

Deskripsi program

Program ini dibuat untuk menentukan biaya parkir berdasarkan jenis kendaraan dan lamanya waktu parkir. Setelah pengguna memasukkan jenis kendaraan dan durasi parkir, program langsung mengecek kombinasi keduanya melalui struktur *switch* yang berisi beberapa kondisi. Setiap *case* mewakili tarif yang berbeda, misalnya motor dengan durasi 1–2 jam dikenai tarif tertentu, motor lebih dari 2 jam memiliki tarif lain, begitu juga dengan mobil dan truk yang masing-masing memiliki aturan tarif berbeda. Jika input tidak sesuai dengan kategori yang tersedia, program menampilkan pesan bahwa data yang dimasukkan tidak valid. Setelah kondisi yang cocok ditemukan, program menghitung tarif dan menampilkannya kepada pengguna. Alur programnya simpel : membaca input, mencocokkan kondisi kendaraan dan durasi , menentukan tarif , menampilkan hasil akhir.

TUGAS

1. Tugas 1

Source code

```
package main

import "fmt"

func main() {
    var ph float64

    fmt.Print("Masukkan kadar 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 in Visual Studio Code. The code editor displays a file named `soal1.go` with the following content:

```

10
11     switch {
12     case ph < 0 || ph > 14:
13         fmt.Println("Nilai pH tidak valid. Nilai pH harus antara 0 dan 14.")
14
15     case ph >= 6.5 && ph <= 8.6:
16         fmt.Println("Air Layak Minum")
17
18     default:
19         fmt.Println("Air Tidak Layak Minum")
20 }

```

The terminal pane shows several runs of the program:

- PS C:\coding didi\golang\Laprak 11> go run "c:\coding didi\golang\Laprak 11\soal1.go"
 Masukkan kadar pH air: 8.6
 Air Layak Minum
- PS C:\coding didi\golang\Laprak 11> go run "c:\coding didi\golang\Laprak 11\soal1.go"
 Masukkan kadar pH air: 9
 Air Tidak Layak Minum
- PS C:\coding didi\golang\Laprak 11> go run "c:\coding didi\golang\Laprak 11\soal1.go"
 Masukkan kadar pH air: 14
 Air Tidak Layak Minum
- PS C:\coding didi\golang\Laprak 11> go run "c:\coding didi\golang\Laprak 11\soal1.go"
 Masukkan kadar pH air: 16
 Nilai pH tidak valid. Nilai pH harus antara 0 dan 14.

The output pane shows the user input and the program's response for each run.

Deskripsi program

Program di atas dibuat untuk mengecek apakah kualitas air layak diminum berdasarkan nilai pH yang dimasukkan pengguna. Saat program dijalankan, pengguna diminta mengetik nilai pH air dalam bentuk angka desimal. Setelah nilai tersebut dibaca, program langsung memeriksa beberapa kemungkinan: jika angkanya berada di luar rentang 0 sampai 14, program menganggap input tidak valid karena pH tidak mungkin melebihi batas tersebut. Jika nilai pH berada dalam kisaran 6.5 hingga 8.6, air dinyatakan layak minum karena masih sesuai standar umum kualitas air. Selain itu, setiap nilai yang berada di luar rentang layak minum akan diberi status tidak layak. Alur pengecekannya berjalan dari validasi angka, kemudian pemeriksaan standar kelayakan, dan terakhir memberikan hasil yang sesuai dengan kondisi pH yang dimasukkan.

2. Tugas 2

Source code

```
package main

import "fmt"

func main() {
    var jenis string
    var durasi int

    fmt.Print("Masukkan jenis kendaraan (motor/mobil/truk) :")
    fmt.Scan(&jenis)
    fmt.Print("Masukkan durasi parkir (jam) : ")
    fmt.Scan(&durasi)

    if durasi < 1 {
        durasi = 1
    }

    var tarifPerJam int

    switch jenis {
    case "motor":
        tarifPerJam = 2000
    case "mobil":
        tarifPerJam = 5000
    case "truk":
        tarifPerJam = 8000
    default:
        fmt.Println("Jenis kendaraan tidak valid.")
        return
    }
}
```

```

        total := tarifPerJam * durasi
        fmt.Printf("Total biaya parkir: Rp %d\n", total)
    }

```

Screenshoot program

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

- EXPLORER:** Shows files in the project: modul1.go (4), modul2.go (1), modul3.go (1), soal1.go (1), and soal2.go (1).
- CODE EDITOR:** The file soal2.go is open, containing a Go program that uses a switch statement to calculate parking fees based on vehicle type (motor, mobil, truk) and duration (jam). It includes validation logic for invalid inputs.
- TERMINAL:** Shows the command `go run "c:\coding\didil\golang\Laprak 11\soal2.go"` being run three times, each time prompting for vehicle type and duration, and then printing the total parking fee.
- OUTPUT:** Displays the results of the terminal runs, showing fees of 6000, 5000, and 5000 respectively.
- STATUS BAR:** Shows the current line (Ln 22, Col 27), tab size (4), encoding (UTF-8), and other developer tools like Go Live and Windurf.

Deskripsi program

Program ini dibuat untuk menghitung biaya parkir berdasarkan jenis kendaraan dan lama waktu parkir yang diinput oleh pengguna. Di awal, program meminta pengguna memasukkan jenis kendaraan dan durasi parkir dalam jam. Jika durasi yang dimasukkan kurang dari satu jam, program otomatis mengubahnya menjadi satu jam sebagai batas minimum. Setelah itu, program menentukan tarif per jam sesuai jenis kendaraan lewat struktur *switch*: motor dikenai tarif 2000 per jam, mobil 5000, dan truk 8000. Jika jenis kendaraan tidak sesuai dengan pilihan yang tersedia, program langsung memberi pesan bahwa data tidak valid dan berhenti. Setelah tarif per jam ditentukan, program mengalikan tarif tersebut dengan durasi parkir untuk mendapatkan total biaya, lalu menampilkannya kepada pengguna. Dengan alur seperti ini, proses perhitungan berjalan sederhana dan jelas mulai dari input, validasi, penentuan tarif, hingga perhitungan total.

3. Tugas 3

Source code

```

package main

import "fmt"

```

```

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

    switch {
    case n%10 == 0:
        hasil := n / 10
        fmt.Println("Kategori: Bilangan Kelipatan 10")
        fmt.Printf("Hasil pembagian antara %d / 10 = %d\n", n, hasil)

    case n%5 == 0 && (n >= 10 || n <= -10):
        hasil := n * n
        fmt.Println("Kategori: Bilangan Kelipatan 5")
        fmt.Printf("Hasil kuadrat dari %d ^2 = %d\n", n, hasil)

    case n%2 != 0:
        hasil := n + (n + 1)
        fmt.Println("Kategori: Bilangan Ganjil")
        fmt.Printf("Hasil penjumlahan dengan bilangan berikutnya %d + %d = %d\n", n, n+1, hasil)

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

```

Screenshot program

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

- EXPLORER**: Shows files in the project: modul1.go, modul2.go, modul3.go, soal1.go, soal2.go, and soal3.go.
- CODE EDITOR**: Displays the source code of soal3.go. The code handles user input for a number and categorizes it based on its remainder when divided by 10, 5, or 2.
- TERMINAL**: Shows command-line output from running the program. It prompts for a number, then prints the category (e.g., "Bilangan Kelipatan 5") and the result of the calculation (e.g., "Hasil kuadrat dari 25 ^2 = 625").
- STATUS BAR**: Shows file path (C:\coding\didid\golang\Laprak 11), tab size (4), encoding (UTF-8), and line/col information (Ln 6, Col 14).

Deskripsi program

Program ini berfungsi untuk mengelompokkan sebuah bilangan ke dalam kategori tertentu—kelipatan 10, kelipatan 5, bilangan ganjil, atau bilangan genap—lalu melakukan operasi sesuai kategori tersebut. Setelah pengguna memasukkan nilai (n), program mengecek kondisinya menggunakan *switch* tanpa ekspresi. Jika angka tersebut habis dibagi 10, program menandainya sebagai kelipatan 10 dan menampilkan hasil pembagian dengan 10. Jika tidak, program memeriksa apakah angka tersebut kelipatan 5 dan memiliki nilai minimal dua digit; kategori ini akan menghasilkan perhitungan kuadrat dari bilangan tersebut. Bila dua kondisi sebelumnya tidak terpenuhi, program mengecek apakah angka tersebut ganjil; jika iya, program menambahkan angka itu dengan bilangan setelahnya. Dan jika semuanya tidak sesuai, angka tersebut dianggap bilangan genap dan program mengalikannya dengan angka berikutnya. setiap input selalu masuk ke salah satu kategori dan mendapatkan hasil perhitungan yang sesuai dengan yang diinputkan.