

**LAPORAN PRAKTIKUM
ALGORITMA PEMROGRAMAN 2**

**MODUL II
REVIEW STRUKTUR KONTROL**



Disusun Oleh :

Rizkulloh Alpriansah

2311102142

IF-11-08

Dosen Pengampu :

Arif Amrulloh, S.Kom., M.Kom

**PROGRAM STUDI S1 INFORMATIKA
FAKULTAS INFORMATIKA
UNIVERSITAS TELKOM PURWOKERTO**

2024

I. DASAR TEORI

1. Setiap file program dalam Golang harus didefinisikan dalam sebuah package. Dalam setiap proyek Go, harus ada setidaknya satu file yang menggunakan package bernama main. File dengan package main ini adalah file yang akan dieksekusi pertama kali saat program dijalankan. Keyword import digunakan untuk menambahkan package lain ke dalam program, sehingga kita dapat memanfaatkan fungsionalitas yang disediakan oleh package tersebut. Salah satu package bawaan Go adalah fmt, yang menawarkan berbagai fungsi untuk kebutuhan input-output (I/O) terkait teks.

Untuk menampilkan teks ke layar (seperti di terminal atau CMD), kita dapat menggunakan fungsi `fmt.Println()`. Mengingat bahwa fungsi `fmt.Println()` merupakan bagian dari package `fmt`, package tersebut harus diimpor terlebih dahulu agar bisa digunakan. Fungsi `fmt.Println()` dapat menerima sejumlah parameter yang tidak terbatas, dan setiap parameter yang diterima akan dipisahkan dengan spasi saat ditampilkan di layar.

2. Struktur kontrol adalah bagian dari bahasa pemrograman yang digunakan untuk menentukan alur eksekusi program. Dalam Go, struktur kontrol dapat dibagi menjadi beberapa kategori utama:

1. Percabangan (Branching):

Percabangan digunakan untuk mengambil keputusan dalam program berdasarkan kondisi tertentu. Go memiliki beberapa bentuk percabangan, termasuk:

if statement:

Digunakan untuk mengeksekusi blok kode jika kondisi yang diberikan bernilai true. Contoh:

```
if x > 0 {  
    fmt.Println("x adalah positif")  
} else {  
    fmt.Println("x adalah negatif atau nol")  
}
```

- switch statement:

Menyediakan cara yang lebih bersih untuk memilih di antara beberapa kemungkinan berdasarkan nilai tertentu. Contoh:

```
switch day {  
    case "Senin":  
        fmt.Println("Hari kerja")  
    case "Sabtu", "Minggu":  
        fmt.Println("Akhir pekan")  
    default:  
        fmt.Println("Hari biasa")  
}
```

2. Perulangan (Looping):

Perulangan digunakan untuk mengeksekusi blok kode berulang kali selama kondisi tertentu terpenuhi. Dalam Go, terdapat dua jenis perulangan:

- for loop:

Merupakan satu-satunya pernyataan perulangan dalam Go. `for` dapat digunakan dalam beberapa cara:

- Dengan inisialisasi, kondisi, dan pernyataan akhir.
- Hanya dengan kondisi.
- Sebagai perulangan tak terbatas.

Contoh:

```
for i := 0; i < 5; i++ {  
    fmt.Println(i)  
}
```

- range loop:

Digunakan untuk mengiterasi elemen dari array, slice, string, map, atau channel.

Contoh:

```
for index, value := range slice {  
    fmt.Println(index, value)
```

```
}
```

3. Pengendalian Alur (Control Flow):

Struktur kontrol ini digunakan untuk mengubah alur eksekusi program secara langsung. Terdapat beberapa cara untuk mengendalikan alur dalam Go:

- break statement:

Menghentikan perulangan atau switch case dan melanjutkan eksekusi setelah blok tersebut.

```
for i := 0; i < 10; i++ {  
    if i == 5 {  
        break  
    }  
    fmt.Println(i)  
}
```

- continue statement:

Melewatkan iterasi saat ini dan melanjutkan ke iterasi berikutnya dalam perulangan.

```
for i := 0; i < 10; i++ {  
    if i%2 == 0 {  
        continue  
    }  
    fmt.Println(i)  
}
```

- goto statement:

Memindahkan alur eksekusi program ke label yang ditentukan. Namun, penggunaan `goto` tidak disarankan karena dapat membuat kode sulit dibaca.

```
goto label
```

```
label:
```

```
    fmt.Println("Ini adalah label")
```

Kesimpulan

Struktur kontrol dalam Go memudahkan pengembang untuk mengatur alur eksekusi program. Dengan memahami dan menggunakan berbagai bentuk percabangan dan perulangan, programmer dapat membuat program yang lebih dinamis dan responsif terhadap kondisi dan input yang berbeda. Penggunaan struktur kontrol yang tepat juga dapat meningkatkan keterbacaan dan maintainability dari kode yang ditulis.

II. GUIDED

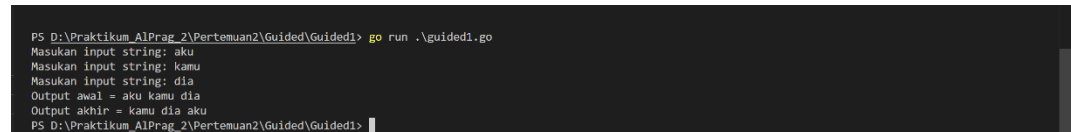
Guided 1

```
package main

import "fmt"

func main() {
    var (
        satu, dua, tiga string
        temp      string
    )
    fmt.Print("Masukan input string: ")
    fmt.Scanln(&satu)
    fmt.Print("Masukan input string: ")
    fmt.Scanln(&dua)
    fmt.Print("Masukan input string: ")
    fmt.Scanln(&tiga)
    fmt.Println("Output awal = " + satu + " " + dua + " " + tiga)
    temp = satu
    satu = dua
    dua = tiga
    tiga = temp
    fmt.Println("Output akhir = " + satu + " " + dua + " " + tiga)
}
```

Screenshots Output



```
PS D:\Praktikum AI\Prag 2\Pertemuan2\Guided\Guided1> go run .\guided1.go
Masukan input string: aku
Masukan input string: kamu
Masukan input string: dia
Output awal = aku kamu dia
Output akhir = kamu dia aku
PS D:\Praktikum AI\Prag 2\Pertemuan2\Guided\Guided1>
```

Deskripsi:

Program di atas meminta tiga input string dari pengguna, lalu mencetak output awal berupa ketiga string tersebut dalam urutan yang diinputkan. Setelah itu, program menukar urutannya dengan cara memindahkan nilai dari string pertama ke string kedua, string kedua ke string ketiga, dan string ketiga ke string pertama. Akhirnya, program mencetak output akhir dengan urutan string yang sudah diubah.

Guided 2

```
package main

import "fmt"

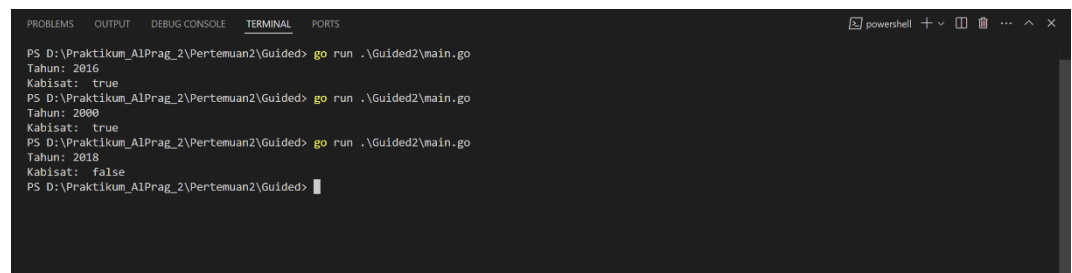
func main() {
    var tahun int
    fmt.Print("Tahun: ")
    fmt.Scanln(&tahun)

    kabisat := cekKabisat(tahun)
    fmt.Println("Kabisat: ", kabisat)
}

func cekKabisat(tahun int) bool {
    if tahun%400 == 0 {
        return true
    } else if tahun%4 == 0 {
        return true
    }
}
```

```
} else if tahun%100 == 0 {  
    return false  
}  
return false  
}
```

Screenshots Output



```
PS D:\Praktikum_AIPrag_2\Pertemuan2\Guided> go run .\Guided2\main.go  
Tahun: 2016  
Kabisat: true  
PS D:\Praktikum_AIPrag_2\Pertemuan2\Guided> go run .\Guided2\main.go  
Tahun: 2000  
Kabisat: true  
PS D:\Praktikum_AIPrag_2\Pertemuan2\Guided> go run .\Guided2\main.go  
Tahun: 2018  
Kabisat: false  
PS D:\Praktikum_AIPrag_2\Pertemuan2\Guided>
```

Deskripsi:

Program di atas digunakan untuk mengecek apakah suatu tahun adalah tahun kabisat atau bukan. Alurnya:

1. Pengguna diminta untuk memasukkan nilai “tahun”.
2. Program memanggil fungsi “cekKabisat(tahun)” yang akan mengevaluasi apakah tahun tersebut memenuhi syarat sebagai tahun kabisat.
3. Di dalam fungsi “cekKabisat”, tahun kabisat didefinisikan dengan kondisi:
 - Tahun habis dibagi 400 (``tahun%400 == 0``), maka tahun tersebut kabisat.
 - Jika tidak, tetapi habis dibagi 4 (``tahun%4 == 0``), maka juga kabisat.
 - Namun, jika habis dibagi 100 (``tahun%100 == 0``), maka bukan tahun kabisat.
 - Selain kondisi di atas, tahun dianggap bukan tahun kabisat.
4. Hasil dari fungsi “cekKabisat” ditampilkan di konsol, menunjukkan apakah `tahun` tersebut kabisat (``true``) atau tidak (``false``).

Kondisi yang harus diperbaiki: Urutan pengecekan ``tahun%100 == 0`` harus didahulukan dari ``tahun%4 == 0``.

Jika tidak, tahun yang habis dibagi 100 dan 4 akan salah terdeteksi sebagai kabisat.

Guided 3

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var jejari int
    fmt.Print("Jejari: ")
    fmt.Scanf("%d", &jejari)

    volume := hitungVolumeBola(float64(jejari))
    luas := hitungLuasKulitBola(float64(jejari))

    fmt.Println("Bola dengan jejari", jejari, "memiliki volume", volume,
        "dan luas kulit", luas)
}

func hitungVolumeBola(jejari float64) float64 {
    const pi = 3.1415926535
    return (4.0 / 3.0) * pi * math.Pow(jejari, 3)
}

func hitungLuasKulitBola(jejari float64) float64 {
    const pi = 3.1415926535
    return 4 * pi * math.Pow(jejari, 2)
}
```

Screenshots Output

```
PS D:\Praktikum_AlPrag_2\Pertemuan2\Guided> go run .\Guided3\main.go
Jejari: 5
Bola dengan jejari 5 memiliki volume 523.5987755833333 dan luas kulit 314.15926535
PS D:\Praktikum_AlPrag_2\Pertemuan2\Guided> █
```

Deskripsi:

Program ini meminta pengguna memasukkan jejari bola, lalu menghitung volume dan luas permukaan bola. Volume dihitung dengan rumus $\frac{4}{3} \times \pi \times \text{jejari}^3$, dan luas permukaan dengan rumus $4 \times \pi \times \text{jejari}^2$, menggunakan konstanta ``pi`` dan fungsi ``math.Pow`` untuk operasi pangkat. Hasilnya berupa volume dan luas kulit bola yang ditampilkan berdasarkan jejari yang diinputkan.

Guided 4

```
package main

import (
    "fmt"
)

func main() {
    // Input suhu dalam Celsius

    var celsius float64

    fmt.Print("Masukkan suhu dalam derajat Celsius: ")

    fmt.Scan(&celsius)

    // Konversi ke Fahrenheit
```

```

fahrenheit := (celsius * 9 / 5) + 32

// Konversi ke Reamur

reamur := celsius * 4 / 5

// Konversi ke Kelvin

kelvin := celsius + 273.15

// Output hasil konversi

fmt.Printf("Temperatur Celsius: %.2f\n", celsius)

fmt.Printf("Derajat Reamur: %.2f\n", reamur)

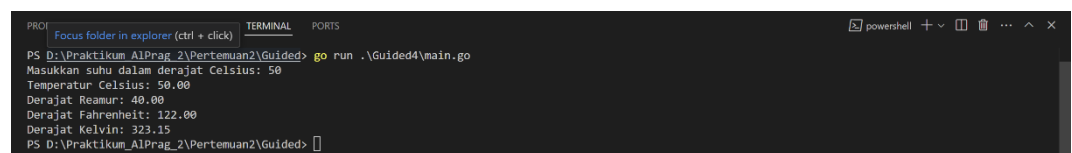
fmt.Printf("Derajat Fahrenheit: %.2f\n", fahrenheit)

fmt.Printf("Derajat Kelvin: %.2f\n", kelvin)

}

```

Screenshots Output



```

PS D:\Praktikum_AI\Prag_2\Pertemuan2\Guided> go run .\Guided4\main.go
Masukkan suhu dalam derajat Celsius: 50
Temperatur Celsius: 50.00
Derajat Reamur: 40.00
Derajat Fahrenheit: 122.00
Derajat Kelvin: 323.15
PS D:\Praktikum_AI\Prag_2\Pertemuan2\Guided>

```

Deskripsi:

Program ini mengubah suhu dari Celsius ke Fahrenheit, Reamur, dan Kelvin. Setelah pengguna memasukkan suhu dalam Celsius, program menghitung Fahrenheit dengan rumus $(F = \frac{9}{5} \times C + 32)$, Reamur dengan $(R = \frac{4}{5} \times C)$, dan Kelvin dengan $(K = C + 273.15)$, lalu menampilkan hasil konversi dalam format desimal dua angka.

Guided 5

```
//By Rizkulloh NIM 2311102142
package main

import (
    "fmt"
)

func main() {
    // Inisialisasi array untuk menampung 5 integer dan 3 karakter
    var nums [5]int
    var chars [3]rune

    // Meminta input untuk 5 data integer
    fmt.Println("Masukkan 5 angka integer (32-127):")
    for i := 0; i < 5; i++ {
        fmt.Scanf("%d", &nums[i])
    }

    // Meminta input untuk 3 karakter
    fmt.Println("Masukkan 3 karakter:")
    for i := 0; i < 3; i++ {
        fmt.Scanf("%c", &chars[i])
        // Mengabaikan newline dari input sebelumnya
        if chars[i] == '\n' {
            i--
        }
    }
}
```

```

// Output pertama: 5 integer yang dikonversi menjadi karakter ASCII
fmt.Println("Keluaran:")
for i := 0; i < 5; i++ {
    fmt.Printf("%c", nums[i])
}
fmt.Println()

// Output kedua: 3 karakter yang diinputkan
for i := 0; i < 3; i++ {
    fmt.Printf("%c", chars[i])
}
fmt.Println()
}

```

Screenshots Output

```

PS D:\Praktikum_AI\Prag_2\Pertemuan2\Guided> go run .\Guided5\main.go
Masukkan 5 angka integer (32-127):
66 97 103 117 115
Masukkan 3 karakter:
SNO
Keluaran:
Bagus
SNO
PS D:\Praktikum_AI\Prag_2\Pertemuan2\Guided>

```

Deskripsi:

Program ini meminta pengguna menginput 5 angka integer (32-127) yang kemudian dikonversi menjadi karakter ASCII, serta 3 karakter tambahan. Setelah semua input diterima, program menampilkan dua output: angka-angka yang diinput ditampilkan sebagai karakter ASCII, dan karakter-karakter yang diinput ditampilkan secara langsung. Program juga mengabaikan newline agar tidak mempengaruhi input karakter.

III. UNGUIDED

Unguided 1

```
//By Rizkulloh NIM 2311102142
package main

import (
    "bufio"
    "fmt"
    "os"
    "strings"
)

func main() {
    // Urutan warna yang benar
    correctOrder_142 := []string{"merah", "kuning", "hijau",
    "ungu"}

    // Membaca input untuk 5 percobaan
    reader_142 := bufio.NewReader(os.Stdin)
    success_142 := true

    for i_142 := 1; i_142 <= 5; i_142++ {
        fmt.Printf("Percobaan %d: ", i_142)

        // Membaca input dari pengguna
        input_142, _ := reader_142.ReadString('\n')
        input_142 = strings.TrimSpace(input_142)

        // Memisahkan input berdasarkan spasi
        colors_142 := strings.Split(input_142, " ")

        // Mengecek apakah urutan warna sesuai
        for j_142 := 0; j_142 < 4; j_142++ {
            if colors_142[j_142] != correctOrder_142[j_142]
        {
                success_142 = false
                break
            }
        }

        // Jika ada percobaan yang tidak sesuai, keluar dari
loop
        if !success_142 {
```

```

        break
    }
}

// Menampilkan hasil
if success_142 {
    fmt.Println("BERHASIL : true")
} else {
    fmt.Println("BERHASIL : false")
}
}

```

Screenshots Output

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS D:\Praktikum_AIPrag_2\Pertemuan2> go run .\Unguided\Unguided1\main.go
Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: merah kuning hijau ungu
Percobaan 5: merah kuning hijau ungu
BERHASIL : true
PS D:\Praktikum_AIPrag_2\Pertemuan2> go run .\Unguided\Unguided1\main.go
Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: ungu kuning hijau merah
BERHASIL : false
PS D:\Praktikum_AIPrag_2\Pertemuan2>

```

Deskripsi:

Program ini mengecek apakah urutan warna yang dimasukkan pengguna sesuai dengan urutan yang benar, yaitu "merah", "kuning", "hijau", dan "ungu". Pengguna diminta untuk memasukkan urutan warna sebanyak 5 kali, yang kemudian dibaca dan dipecah menjadi array `colors_142`. Program membandingkan setiap warna yang dimasukkan dengan urutan yang benar dan mengubah variabel `success_142` menjadi `false` jika ada warna yang tidak sesuai, sehingga proses percobaan berhenti. Pada akhir, jika semua percobaan berhasil, program menampilkan "BERHASIL : true"; jika ada kesalahan, ditampilkan "BERHASIL : false". Semua variabel dalam program diberi akhiran `_142` sesuai permintaan.

Unguided 2

```
//By Rizkulloh NIM 2311102142
package main

import (
    "bufio"
    "fmt"
    "os"
    "strconv"
    "strings"
)

func main() {
    // Membuat reader untuk membaca input dari pengguna
    reader_142 := bufio.NewReader(os.Stdin)

    // Meminta input jumlah bunga yang akan dimasukkan
    // (bilangan bulat positif N)
    fmt.Print("N: ")
    var N_142 int
    for {
        // Baca input dari pengguna
        input_142, err := reader_142.ReadString('\n')
        if err != nil {
            fmt.Printf("Error membaca input: %v\n", err)
            return
        }

        // Konversi input ke integer
        N_142, err =
        strconv.Atoi(strings.TrimSpace(input_142))
        if err != nil || N_142 <= 0 {
            fmt.Println("Harap masukkan bilangan bulat
        positif.")
        } else {
            break
        }
    }

    // Inisialisasi variabel pita (string) untuk menyimpan
    nama bunga
    var pita_142 string
    var count_142 int // Menyimpan jumlah bunga yang
    dimasukkan
```



```

        // Loop untuk menerima input nama bunga sebanyak N kali
        for i_142 := 1; i_142 <= N_142; i_142++ {
            fmt.Printf("Bunga %d: ", i_142) // Menambahkan
instruksi

            // Membaca input dari pengguna
            input_142, err := reader_142.ReadString('\n')
            if err != nil {
                fmt.Printf("Error: %v\n", err)
                return // Keluar dari program jika ada kesalahan
            }

            // Menghapus spasi dan karakter newline dari input
            input_142 = strings.TrimSpace(input_142)

            // Cek jika pengguna mengetik "SELESAI"
            if strings.ToUpper(input_142) == "SELESAI" {
                break // Menghentikan input jika "SELESAI"
dimasukkan
            }

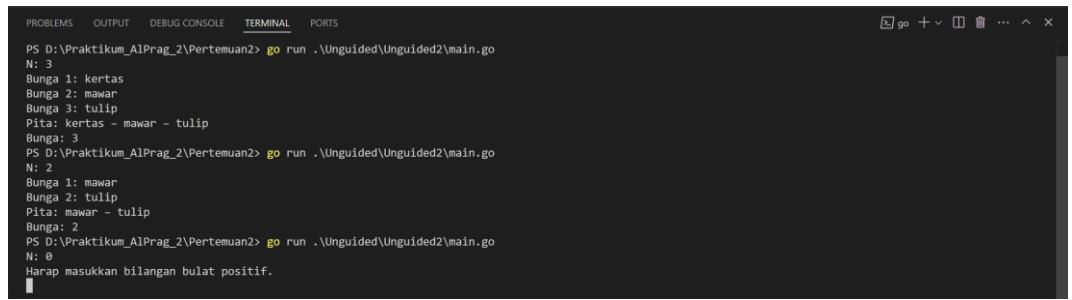
            // Menggabungkan nama bunga dengan pita menggunakan
" - " sebagai pemisah
            if pita_142 == "" {
                pita_142 = input_142 // Jika pita masih kosong,
langsung masukkan nama bunga
            } else {
                pita_142 = pita_142 + " - " + input_142 // Jika
sudah ada isinya, tambahkan dengan pemisah " - "
            }

            count_142++ // Menambah jumlah bunga yang dimasukkan
        }

        // Menampilkan isi pita dan jumlah bunga setelah semua
input dimasukkan
        fmt.Println("Pita:", pita_142)
        fmt.Printf("Bunga: %d\n", count_142)
    }
}

```

Screenshots Output

A screenshot of a terminal window showing the execution of a Go program. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The output shows three runs of the program. Each run starts with a prompt 'PS D:\Praktikum_AI\Prag_2\Pertemuan2>' followed by 'go run .\Unguided\Unguided2\main.go'. The first run shows 'N: 3', 'Bunga 1: kertas', 'Bunga 2: mawar', 'Bunga 3: tulip', and 'Pita: kertas - mawar - tulip'. The second run shows 'N: 2', 'Bunga 1: mawar', 'Bunga 2: tulip', 'Pita: mawar - tulip'. The third run shows 'N: 0' and the message 'Harap masukkan bilangan bulat positif.'.

```
PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided2\main.go
N: 3
Bunga 1: kertas
Bunga 2: mawar
Bunga 3: tulip
Pita: kertas - mawar - tulip
PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided2\main.go
N: 2
Bunga 1: mawar
Bunga 2: tulip
Pita: mawar - tulip
PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided2\main.go
N: 0
Harap masukkan bilangan bulat positif.
```

Deskripsi:

Program ini meminta pengguna untuk memasukkan jumlah bunga yang ingin diinput (bilangan bulat positif) dan kemudian mengumpulkan nama-nama bunga sebanyak jumlah yang ditentukan. Setelah pengguna memasukkan jumlah bunga, program membaca nama bunga satu per satu, dan jika pengguna mengetik "SELESAI", input akan dihentikan. Nama bunga yang dimasukkan akan digabungkan dalam satu string bernama `pita 142`, dengan pemisah " – " antara setiap nama bunga. Program juga menghitung jumlah bunga yang dimasukkan dengan variabel `count 142`. Akhirnya, program menampilkan isi dari pita bunga dan jumlah total bunga yang dimasukkan.

Unguided 3

```
//By Rizkulloh NIM 2311102142
package main

import (
    "fmt"
)

func main() {
    var beratKantongKiri_142, beratKantongKanan_142 float64
    for {
        // Menerima input berat kedua kantong dalam satu baris
        fmt.Print("Masukan berat belanjaan di kedua kantong (kg): ")
        fmt.Scan(&beratKantongKiri_142, &beratKantongKanan_142)
```

```

        // Cek jika salah satu kantong mencapai atau lebih
dari 9 kg
        if beratKantongKiri_142 >= 9 ||
beratKantongKanan_142 >= 9 {
            fmt.Println("Proses selesai.")
            break
        }

        // Hitung selisih berat antara kantong kiri dan
kanan
        selisihBerat_142 := beratKantongKiri_142 -
beratKantongKanan_142
        if selisihBerat_142 < 0 {
            selisihBerat_142 = -selisihBerat_142
        }

        // Cek apakah selisih berat melebihi 9 kg
        if selisihBerat_142 > 9 {
            fmt.Println("Selisih berat antara kantong kiri
dan kanan melebihi 9 kg.")
        }
    }
}

```

Screenshots Output

```

Mode                LastWriteTime         Length Name
-----
d-----          10/6/2024   5:56 PM             Guided
d-----          10/6/2024   4:50 PM             Unguided

PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided3\setelah_modif\main.go
Masukan berat belanja di kedua kantong (kg): 5.5 1.1
Masukan berat belanja di kedua kantong (kg): 7.1 8.5
Masukan berat belanja di kedua kantong (kg): 2 6
Masukan berat belanja di kedua kantong (kg): 9 8.5
Proses selesai.
PS D:\Praktikum_AI\Prag_2\Pertemuan2>

```

Deskripsi:

Program ini meminta pengguna untuk memasukkan berat dari dua kantong belanja. Variabel beratKantongKiri_142 dan beratKantongKanan_142 digunakan untuk menyimpan berat masing-masing kantong. Program akan memeriksa apakah salah satu dari kantong tersebut memiliki berat 9 kg atau lebih. Jika iya, proses akan dihentikan dan program mencetak "Proses selesai." Jika tidak, program akan menghitung selisih berat antara kedua kantong dan menyimpannya di variabel selisihBerat_142. Jika selisih berat

ini lebih dari 9 kg, program akan menampilkan pesan yang menyatakan bahwa selisih berat antara kantong kiri dan kanan melebihi 9 kg.

Setelah di modif

```
package main

import (
    "fmt"
)

func main() {
    var beratKantongKiri_142, beratKantongKanan_142 float64
    for {
        // Menerima input berat kedua kantong dalam satu baris
        fmt.Print("Masukan berat belanjaan di kedua kantong (kg): ")

        fmt.Scan(&beratKantongKiri_142,
            &beratKantongKanan_142)

        // Cek jika salah satu kantong memiliki berat negatif
        if beratKantongKiri_142 < 0 || beratKantongKanan_142 < 0 {
            fmt.Println("Proses selesai.")
            break
        }

        // Cek jika total berat kedua kantong melebihi 150 kg
        totalBerat_142 := beratKantongKiri_142 +
            beratKantongKanan_142
        if totalBerat_142 > 150 {
            fmt.Println("Proses selesai.")
        }
    }
}
```

```

        break
    }

    // Hitung selisih berat antara kantong kiri dan kanan
    selisihBerat_142 := beratKantongKiri_142 -
beratKantongKanan_142
    if selisihBerat_142 < 0 {
        selisihBerat_142 = -selisihBerat_142
    }

    // Menampilkan hasil apakah sepeda motor akan oleng
atau tidak
    if selisihBerat_142 >= 9 {
        fmt.Println("Sepeda motor Pak Andi akan oleng:
true")
    } else {
        fmt.Println("Sepeda motor Pak Andi akan oleng:
false")
    }
}
}

```

Output

```

PS D:\Praktikum_AIPrag_2\Pertemuan2> go run .\Unguided\Unguided3\setelah_modif\main.go
Masukan berat belanjaan di kedua kantong (kg): 5.5 1.1
Masukan berat belanjaan di kedua kantong (kg): 7.1 8.5
Masukan berat belanjaan di kedua kantong (kg): 2 6
Masukan berat belanjaan di kedua kantong (kg): 9 8.5
Proses selesai.
PS D:\Praktikum_AIPrag_2\Pertemuan2> go run .\Unguided\Unguided3\setelah_modif\main.go
Masukan berat belanjaan di kedua kantong (kg): 5 10
Sepeda motor Pak Andi akan oleng: false
Masukan berat belanjaan di kedua kantong (kg): 55.6 70.2
Sepeda motor Pak Andi akan oleng: true
Masukan berat belanjaan di kedua kantong (kg): 72.3 66.9
Sepeda motor Pak Andi akan oleng: false
Masukan berat belanjaan di kedua kantong (kg): 59.5 98.7
Proses selesai.
PS D:\Praktikum_AIPrag_2\Pertemuan2> █

```

Deskripsi

Program ini berfungsi untuk memantau berat dua kantong belanjaan. Pengguna diminta untuk memasukkan berat dari kantong kiri (beratKantongKiri 142) dan kanan (beratKantongKanan 142). Program

akan terus meminta input selama kedua berat tidak negatif dan total berat dari kedua kantong tidak melebihi 150 kg. Jika salah satu kantong memiliki berat negatif atau totalnya lebih dari 150 kg, program akan mencetak "Proses selesai." dan keluar dari loop. Selanjutnya, program menghitung selisih berat antara kantong kiri dan kanan, disimpan dalam variabel selisihBerat 142. Jika selisih berat lebih dari atau sama dengan 9 kg, program akan menampilkan bahwa sepeda motor Pak Andi akan oleng; jika tidak, program akan menampilkan bahwa sepeda motor tidak akan oleng.

Unguided 4

Setelah DiModif

Source code

```
package main

import (
    "fmt"
)

// Fungsi untuk menghitung nilai hampiran akar 2 berdasarkan iterasi K
func hitungAkar2(iterasi_142 int) float64 {
    hasilPerkalian_142 := 1.0 // Mulai dengan nilai produk 1

    // Iterasi dari 0 hingga iterasi yang dimasukkan
    for indeks_142 := 0; indeks_142 < iterasi_142; indeks_142++ { //
Menggunakan < bukan <=
```

```

        pembilang_142 := (4*float64(indeks_142) + 2) *
(4*float64(indeks_142) + 2) // (4k + 2)^2

        penyebut_142 := (4*float64(indeks_142) + 1) *
(4*float64(indeks_142) + 3) // (4k + 1)(4k + 3)

        hasilPerkalian_142 *= pembilang_142 / penyebut_142
    }

    return hasilPerkalian_142 // Mengembalikan nilai hasil perkalian
}

func main() {

    var jumlahIterasi_142 int

    // Membaca input nilai jumlah iterasi

    fmt.Print("Masukkan jumlah iterasi K: ")

    fmt.Scan(&jumlahIterasi_142)

    // Memastikan jumlah iterasi tidak negatif

    if jumlahIterasi_142 < 0 {

        fmt.Println("Jumlah iterasi K tidak boleh negatif.")

        return

    }
}

```

```

// Menghitung nilai hampiran sqrt(2)

hasilAkar2_142 := hitungAkar2(jumlahIterasi_142)

// Menampilkan hasil dengan 10 angka di belakang koma

fmt.Printf("Nilai hampiran akar 2 = %.10f\n", hasilAkar2_142)

}

```

Output

```

PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided4\Modif\main.go
Masukkan jumlah iterasi K: 10
Nilai hampiran akar 2 = 1.4054086752
PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided4\Modif\main.go
Masukkan jumlah iterasi K: 100
Nilai hampiran akar 2 = 1.4133299615
PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided4\Modif\main.go
Masukkan jumlah iterasi K: 1000
Nilai hampiran akar 2 = 1.4141251768
PS D:\Praktikum_AI\Prag_2\Pertemuan2>

```

Deskripsi

Program ini menghitung nilai hampiran akar dua ($\sqrt{2}$) menggunakan metode iterasi berdasarkan rumus tertentu. Fungsi `hitungAkar2` menerima parameter `iterasi_142`, yang menentukan berapa kali perhitungan dilakukan. Di dalam fungsi ini, `hasilPerkalian_142` diinisialisasi dengan nilai 1.0, dan perulangan dilakukan dari 0 hingga jumlah iterasi yang ditentukan. Dalam setiap iterasi, pembilang dihitung dengan rumus $(4k+2)^2(4k+2)^2(4k+2)^2$ dan penyebut dengan rumus $(4k+1)(4k+3)(4k+1)(4k+3)(4k+1)(4k+3)$, lalu hasil pembagian pembilang dengan penyebut dikalikan dengan `hasilPerkalian_142`. Setelah semua iterasi selesai, fungsi mengembalikan nilai akhir dari `hasilPerkalian_142`. Di dalam fungsi `main`, program meminta pengguna untuk memasukkan jumlah iterasi, memeriksa apakah nilai tersebut negatif, dan jika valid, akan memanggil fungsi untuk menghitung dan menampilkan **hasil hampiran akar dua hingga 10 angka di belakang koma**.

Unguided 5

```
//By Rikulloh NIM 2311102142
package main

import (
    "fmt"
)

func hitungBiayaKirim(berat_142 int) int {
    // Menghitung berat dalam kilogram dan gram
    kg_142 := berat_142 / 1000
    gram_142 := berat_142 % 1000

    // Biaya pengiriman per kilogram
    biayaPerKg_142 := 10000
    biayaTotal_142 := kg_142 * biayaPerKg_142

    // Biaya tambahan untuk sisa gram
    biayaTambahan_142 := 0
    if kg_142 >= 10 {
        biayaTambahan_142 = 0 // Gratis biaya tambahan jika
berat lebih dari 10 kg
    } else {
        if gram_142 >= 500 {
            biayaTambahan_142 = gram_142 * 5 // Rp. 5 per
gram jika sisa >= 500 gram
        } else {
            biayaTambahan_142 = gram_142 * 15 // Rp. 15 per
gram jika sisa < 500 gram
        }
    }

    // Total biaya
    return biayaTotal_142 + biayaTambahan_142
}

func main() {
    var berat_142 int

    // Meminta input berat dari pengguna
    fmt.Print("Berat parsel (gram): ")
    fmt.Scan(&berat_142)

    // Menghitung berat dalam kg dan gram
```

```

kg_142 := berat_142 / 1000
gram_142 := berat_142 % 1000

// Menghitung total biaya pengiriman
biayaPerKg_142 := 10000 * kg_142
biayaTambahan_142 := 0

// Kondisi untuk biaya tambahan berdasarkan sisa berat
gram
if kg_142 >= 10 {
    biayaTambahan_142 = 0 // Gratis biaya tambahan jika
lebih dari 10 kg
} else {
    if gram_142 >= 500 {
        biayaTambahan_142 = gram_142 * 5 // Rp. 5 per
gram untuk sisa >= 500 gram
    } else {
        biayaTambahan_142 = gram_142 * 15 // Rp. 15 per
gram untuk sisa < 500 gram
    }
}

// Menghitung total biaya
totalBiaya_142 := biayaPerKg_142 + biayaTambahan_142

// Menampilkan hasil
fmt.Printf("Detail berat: %d kg + %d gr\n", kg_142,
gram_142)
fmt.Printf("Detail biaya: Rp. %d + Rp. %d\n",
biayaPerKg_142, biayaTambahan_142)
fmt.Printf("Total biaya: Rp. %d\n", totalBiaya_142)
}

```

Screenshots Output

```

PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided5\main.go
Berat parcel (gram): 8500
Detail berat: 8 kg + 500 gr
Detail biaya: Rp. 80000 + Rp. 2500
Total biaya: Rp. 82500
PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided5\main.go
Berat parcel (gram): 9250
Detail berat: 9 kg + 250 gr
Detail biaya: Rp. 90000 + Rp. 3750
Total biaya: Rp. 93750
PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided5\main.go
Berat parcel (gram): 11750
Detail berat: 11 kg + 750 gr
Detail biaya: Rp. 110000 + Rp. 0
Total biaya: Rp. 110000
PS D:\Praktikum_AI\Prag_2\Pertemuan2>

```

Deskripsi:

Program ini menghitung biaya pengiriman parcel berdasarkan beratnya yang diinput dalam gram. Fungsi hitungBiayaKirim menerima parameter berat_142 untuk menghitung berat dalam kilogram (kg_142) dan gram (gram_142). Biaya pengiriman per kilogram ditetapkan Rp. 10.000, dan biaya tambahan dihitung berdasarkan sisa berat gram: jika berat mencapai 10 kg, biaya tambahan gratis; jika sisa lebih dari atau sama dengan 500 gram, biaya tambahan Rp. 5 per gram; jika kurang dari 500 gram, Rp. 15 per gram. Di dalam fungsi main, pengguna diminta untuk memasukkan berat parcel, dan total biaya ditampilkan bersama dengan rincian berat dan biaya.

Unguided 6

```
//By Rizkulloh NIM 2311102142
package main

import "fmt"

func main() {
    var nam float64
    var nmk string

    // Meminta input nilai
    fmt.Print("Nilai akhir mata kuliah: ")
    fmt.Scan(&nam)

    // Logika penentuan nilai huruf berdasarkan nilai
    numerik
    if nam > 80 {
        nmk = "A"
    } else if nam > 72.5 {
        nmk = "AB"
    } else if nam > 65 {
        nmk = "B"
    } else if nam > 57.5 {
        nmk = "BC"
    } else if nam > 50 {
        nmk = "C"
    } else if nam > 40 {
        nmk = "D"
    }
```

```

    } else {
        nmK = "E"
    }

    // Menampilkan hasil
    fmt.Printf("Nilai mata kuliah: %s\n", nmK)
}

```

Screenshots Output

```

PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided6\main.go
Nilai akhir mata kuliah: 88.1
Nilai mata kuliah: A
PS D:\Praktikum_AI\Prag_2\Pertemuan2>

```

- a.
- b. **Apa saja kesalahan dari program tersebut? Mengapa demikian? Jelaskan alur program seharusnya!**

Kesalahan dalam Program: Variabel nam = "B" seharusnya ditulis sebagai nmK = "B"

Dalam kode ini, semua kondisi menggunakan if tanpa else if untuk rentang nilai yang saling tumpang tindih. Dengan cara ini, jika nilai nam lebih dari 80, maka nilai nmK akan ditetapkan ke "A", tetapi program akan tetap melanjutkan pemeriksaan kondisi lain. Hal ini tidak efisien dan dapat menghasilkan output yang tidak diinginkan. Pada pernyataan fmt.Printf, format string tidak digunakan. Seharusnya ada format placeholder untuk variabel nmK.

Dalam kondisi penentuan nilai E, menggunakan else if setelah kondisi if biasa. Sebaiknya gunakan else tanpa syarat sebelumnya.

Alur Program seharusnya:

Meminta input

Menentukan nilai huruf:

Jika nam lebih dari 80, maka nmK menjadi "A".

Jika nam lebih dari 72.5 dan kurang dari atau sama dengan 80, maka nmK menjadi "AB".

Jika nam lebih dari 65 dan kurang dari atau sama dengan 72.5, maka nmK menjadi "B".

Jika nam lebih dari 57.5 dan kurang dari atau sama dengan 65, maka nmK menjadi "BC".

Jika nam lebih dari 50 dan kurang dari atau sama dengan 57.5, maka nmK menjadi "C".

Jika nam lebih dari 40 dan kurang dari atau sama dengan 50, maka nmK menjadi "D".

Jika nam kurang dari atau sama dengan 40, maka nmK menjadi "E".

- c.

IV. package main

V.

```
VI.      import "fmt"
VII.
VIII.    func main() {
IX.        var nam_142 float64
X.        var nmk_142 string
XI.
XII.        // Meminta input nilai
XIII.       fmt.Print("Nilai akhir mata kuliah: ")
XIV.       fmt.Scan(&nam_142)
XV.
XVI.       // Logika penentuan nilai huruf berdasarkan nilai numerik
XVII.      if nam_142 > 80 {
XVIII.          nmk_142 = "A"
XIX.      } else if nam_142 > 72.5 {
XX.          nmk_142 = "AB"
XXI.      } else if nam_142 > 65 {
XXII.          nmk_142 = "B"
XXIII.      } else if nam_142 > 57.5 {
XXIV.          nmk_142 = "BC"
XXV.      } else if nam_142 > 50 {
XXVI.          nmk_142 = "C"
XXVII.      } else if nam_142 > 40 {
XXVIII.          nmk_142 = "D"
XXIX.      } else {
XXX.          nmk_142 = "E"
XXXI.      }
XXXII.
XXXIII.     // Menampilkan hasil
XXXIV.     fmt.Printf("Nilai Indeks untuk nilai akhir mata kuliah:
XXXV.     %s\n", nmk_142)
XXXVI. }
```

Unguided 7

Setelah Pembaruan

Source Code

```
package main

import (
    "fmt"
)

func main() {
    var number_142 int

    // Input bilangan bulat
    fmt.Print("Masukkan bilangan: ")
    fmt.Scan(&number_142)

    // Mencari dan menampilkan semua faktor dari bilangan
    fmt.Printf("Faktor: ")
    factors_142 := findFactors(number_142)

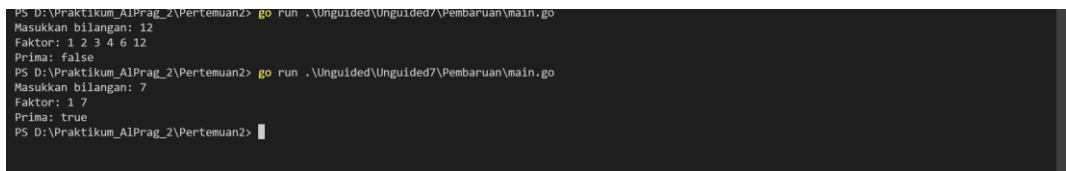
    // Menampilkan faktor-faktor
    for _, factor_142 := range factors_142 {
        fmt.Printf("%d ", factor_142)
    }
    fmt.Println()

    // Mengecek apakah bilangan tersebut prima
    if len(factors_142) == 2 {
        fmt.Println("Prima: true")
    } else {
        fmt.Println("Prima: false")
    }
}

// Fungsi untuk mencari faktor dari bilangan
func findFactors(n_142 int) []int {
```

```
var factors_142 []int
for i_142 := 1; i_142 <= n_142; i_142++ {
    if n_142%i_142 == 0 {
        factors_142 = append(factors_142, i_142)
    }
}
return factors_142
}
```

Output



```
PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided7\Pembaruan\main.go
Masukkan bilangan: 12
Faktor: 1 2 3 4 6 12
Prima: false
PS D:\Praktikum_AI\Prag_2\Pertemuan2> go run .\Unguided\Unguided7\Pembaruan\main.go
Masukkan bilangan: 7
Faktor: 1 7
Prima: true
PS D:\Praktikum_AI\Prag_2\Pertemuan2> |
```

Deskripsi

Program ini meminta pengguna untuk memasukkan sebuah bilangan bulat dan kemudian mencari serta menampilkan semua faktor dari bilangan tersebut. Setelah menerima input, program menggunakan fungsi `findFactors` untuk mendapatkan faktor-faktor dari bilangan yang dimasukkan dan menyimpannya dalam slice `factors_142`. Setelah itu, program mencetak semua faktor yang ditemukan. Selanjutnya, program memeriksa apakah jumlah faktor tersebut sama dengan 2, yang menunjukkan bahwa bilangan tersebut adalah bilangan prima (hanya memiliki faktor 1 dan dirinya sendiri), dan mencetak hasilnya.