

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 2  
REVIEW STRUKTUR KONTROL**



**Disusun Oleh :**

**M. Faleno Albar Firjatulloh / 2311102297**

**S1-IF-11-05**

**Dosen Pengampu :**

**Arif Amrulloh, S.Kom., M.Kom**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## **I. DASAR TEORI**

### **A. Struktur Program Go**

#### **- A. Struktur Program Go**

- Keyword ``package``: Setiap file dalam program Go harus dideklarasikan dalam package. Setidaknya, harus ada satu file dengan package ``main``, yang akan dijalankan pertama kali saat program dieksekusi.

- Keyword ``import``: Digunakan untuk mengimpor package lain ke dalam program agar fungsi-fungsi dalam package tersebut dapat digunakan. Contoh package bawaan Go adalah ``fmt``, yang menyediakan fungsi-fungsi untuk operasi input/output terkait teks.

- Keyword ``var``: Digunakan untuk mendeklarasikan variabel baru.

- Fungsi ``main()``: Fungsi utama dalam package ``main``, yang menjadi titik awal eksekusi program.

- Fungsi ``fmt.Println()``: Digunakan untuk menampilkan teks ke layar. Fungsi ini berasal dari package ``fmt``, sehingga package ini harus diimpor terlebih dahulu.

- Fungsi ``fmt.Printf()``: Serupa dengan ``fmt.Println()``, namun memberikan lebih banyak fleksibilitas untuk mengatur format output.

#### **B. Tipe Data dan Instruksi Dasar**

##### **- Tipe Data Numerik Non-Desimal:**

- ``uint``: Digunakan untuk bilangan cacah (hanya positif).

- ``int``: Digunakan untuk bilangan bulat (baik positif maupun negatif).

##### **- Tipe Data Numerik Desimal:**

- ``float32`` dan ``float64``: Perbedaananya terletak pada cakupan nilai desimal yang dapat disimpan.

- Tipe Data Boolean (``bool``): Hanya memiliki dua nilai, yaitu ``true`` dan ``false``. Sering digunakan dalam logika kondisi dan perulangan.

- Tipe Data String: Teks yang nilainya diletakkan di antara tanda kutip dua ("").

- Konstanta Simbolik: Memberikan nama pada nilai tetap, seperti PI untuk mewakili nilai  $\pi$ .

### C. Struktur Kontrol Perulangan

Go hanya memiliki satu keyword `for` untuk melakukan perulangan, dengan dua bentuk yang umum digunakan:

1. While-Loop: Perulangan berlangsung selama kondisi yang diberikan bernilai true, dan berhenti ketika kondisi berubah menjadi false.
2. Repeat-Until: Perulangan terus berjalan hingga kondisi tertentu terpenuhi.

### D. Struktur Kontrol Percabangan

1. If-Else: Mendukung berbagai bentuk `if-else`, baik dalam bentuk sederhana maupun bersarang.
2. Switch-Case: Go memiliki dua varian `switch-case`. Yang pertama menggunakan ekspresi pada perintah `switch` dan mencocokkannya dengan nilai di `case`. Varian kedua adalah `switch` tanpa ekspresi, di mana setiap `case` berisi ekspresi boolean, yang membuatnya lebih mirip dengan bentuk `if-else`.

## II. GUIDED

### 1. Guided 1 - Soal Nama

#### Soal Studi Case

Membuat sebuah program sederhana yang meminta pengguna memasukkan nama mereka, lalu mencetak nama tersebut ke layar. Program ini dirancang menggunakan bahasa pemrograman Go dan bertujuan untuk memahami dasar penggunaan input/output serta variabel.

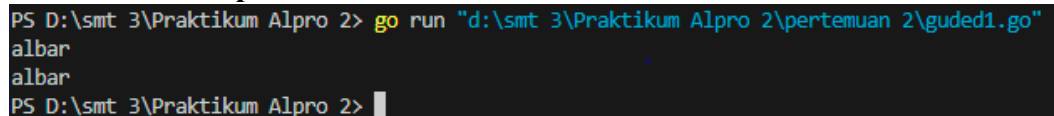
#### Sourcecode

```
package main

import (
    "fmt"
)

func main() {
    var nama string
    fmt.Scanln(&nama)
    fmt.Println(nama)
}
```

#### Screenshoot Output



```
PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\guded1.go"
albar
albar
PS D:\smt 3\Praktikum Alpro 2> |
```

#### Deskripsi Program

Program di atas merupakan program sederhana dalam bahasa Go yang bertujuan untuk menerima input dari pengguna dan menampilkannya kembali di layar. Berikut adalah penjelasan singkat mengenai skrip tersebut:

1. Package main: Program berada dalam package `*main*`, yang menunjukkan bahwa ini adalah file utama yang akan dijalankan.
2. Import fmt: Package ``fmt`` diimpor agar fungsi-fungsi I/O, seperti membaca input dan menampilkan output, dapat digunakan.
3. Variabel nama: Variabel bernama ``nama`` dideklarasikan dengan tipe data ``string`` untuk menampung input dari pengguna.
4. `fmt.Scanln()`: Fungsi ini digunakan untuk mengambil input pengguna dan menyimpannya dalam variabel ``nama``.

5. `fmt.Println()`: Fungsi ini menampilkan kembali isi variabel ``nama``, yaitu input yang diberikan oleh pengguna, ke layar..

## 2. Guided 2 – 2B – 1

### Soal Studi Case

Siswa kelas IPA di salah satu sekolah menengah atas di Indonesia sedang mengadakan praktikum kimia. Di setiap percobaan akan menggunakan 4 tabung reaksi, yang mana susunan warna cairan di setiap tabung akan menentukan hasil percobaan. Siswa diminta untuk mencatat hasil percobaan tersebut. Percobaan dikatakan berhasil apabila susunan warna zat cair pada gelas 1 hingga gelas 4 secara berturut-turut adalah 'merah', 'kuning', 'hijau', dan 'ungu' selama 5 kali percobaan berulang.

Buatlah sebuah program yang menerima input berupa warna dari ke 4 gelas reaksi sebanyak 5 kali percobaan. Kemudian program akan menampilkan `true` apabila urutan warna sesuai dengan informasi yang diberikan pada paragraf sebelumnya, dan `false` untuk urutan warna lainnya. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

### Sourcecode

```
package main

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

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

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

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

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

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

// Mengecek apakah urutan warna sesuai
for j := 0; j < 4; j++ {
    if colors[j] != correctOrder[j] {
        success = false
        break
    }
}

// Jika ada percobaan yang tidak sesuai,
keluar dari loop
if !success {
    break
}

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

## Screenshot Output

```

PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\guided.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:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\guided.go"
Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: ungu hijau kuning merah
BERHASIL: false
PS D:\smt 3\Praktikum Alpro 2> 

```

### Deskripsi Program

Program di atas adalah program dalam bahasa Go yang memeriksa apakah urutan warna yang diinputkan oleh pengguna sesuai dengan urutan warna yang telah ditentukan. Berikut deskripsi singkatnya:

1. **Urutan warna yang benar:** Program mendefinisikan urutan warna yang benar dalam slice `correctOrder`, yaitu "merah", "kuning", "hijau", "ungu".

2. **Input pengguna:** Pengguna diminta untuk memasukkan urutan warna selama 5 kali percobaan. Program menggunakan `bufio.NewReader` untuk membaca input dari pengguna baris demi baris.

#### 3. Proses validasi: Pada setiap percobaan:

- Program membaca input dari pengguna dan membersihkannya dari karakter newline (`\n`) menggunakan `strings.TrimSpace`.
- Input tersebut dipisahkan menjadi elemen-elemen warna berdasarkan spasi menggunakan `strings.Split`.
- Program memeriksa apakah urutan warna yang diinputkan sesuai dengan urutan yang benar (`correctOrder`).
- Jika pada salah satu percobaan urutan tidak sesuai, program keluar dari loop dan menyatakan percobaan gagal.

4. **Hasil:** Setelah 5 percobaan atau setelah menemukan ketidaksesuaian, program menampilkan hasil. Jika semua input benar, program mencetak "BERHASIL: true", jika tidak, "BERHASIL: false".

### 3. Guided – 3 – Soal abcd Soal Studi Case

Membuat program sederhana yang memungkinkan pengguna memasukkan 5 angka integer, kemudian program menjumlahkan kelima angka tersebut dan menampilkan hasilnya.

### Sourcecode

```
package main

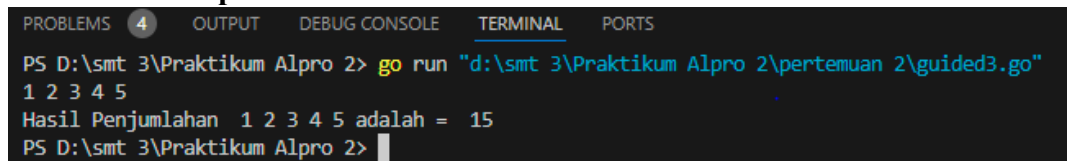
import "fmt"

func main() {

    var a, b, c, d, e int
    var hasil int
    fmt.Scanln(&a, &b, &c, &d, &e)

    hasil = a + b + c + d + e
    fmt.Println("Hasil Penjumlahan ", a, b, c,
d, e, "adalah = ", hasil)
}
```

### Screenshot Output



```
PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\guided3.go"
1 2 3 4 5
Hasil Penjumlahan 1 2 3 4 5 adalah = 15
PS D:\smt 3\Praktikum Alpro 2> |
```

### Deskripsi Program

Program di atas adalah program sederhana dalam bahasa Go yang melakukan penjumlahan lima angka yang dimasukkan oleh pengguna. Berikut adalah deskripsi singkat dari program tersebut:

- 1. Deklarasi variabel:** Program mendeklarasikan lima variabel `a`, `b`, `c`, `d`, dan `e` bertipe `int` untuk menyimpan input bilangan dari pengguna, serta variabel `hasil` untuk menyimpan hasil penjumlahan.
- 2. Membaca input:** Program menggunakan `fmt.Scanln()` untuk membaca lima bilangan yang dimasukkan oleh pengguna dan menyimpannya ke dalam variabel `a`, `b`, `c`, `d`, dan `e`.
- 3. Proses penjumlahan:** Program menjumlahkan nilai-nilai dari kelima variabel tersebut dan menyimpannya ke dalam variabel `hasil`.



4. **Menampilkan hasil:** Program mencetak hasil penjumlahan beserta nilai-nilai yang dimasukkan oleh pengguna dengan menggunakan `fmt.Println()`.

4. **Guided 4**

**Soal Studi Case**

Diberikan sebuah nilai akhir mata kuliah (NAM) [0..100] dan standar penilaian nilai mata kuliah (NMK) sebagai berikut:

NAM	NMK
NAM > 80	A
72.5 < NAM <= 80	AB

65 < NAM <= 72.5	B
57.5 < NAM <= 65	BC
50 < NAM <= 57.5	C
40 < NAM <= 50	D
NAM <= 40	E

**Sourcecode**

```
package main

import (
    "fmt"
)

func main() {
    var nilai float32
    var indeks string

    // Meminta input nilai
    fmt.Print("Masukkan nilai: ")
    fmt.Scan(&nilai)

    // Logika penentuan nilai huruf berdasarkan
    nilai numerik
    if nilai >= 80 {
        indeks = "A"
```

```

    } else if nilai >= 70 {
        indeks = "B"
    } else if nilai >= 65 {
        indeks = "C"
    } else if nilai >= 45 {
        indeks = "D"
    } else if nilai >= 40 {
        indeks = "E"
    } else {
        indeks = "F"
    }

    // Menampilkan hasil
    fmt.Printf("Nilai Indeks untuk nilai %.2f
adalah %s\n", nilai, indeks)
}

```

### Screenshot Output

```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\guided4.go"
Masukkan nilai: 93.5
Nilai Indeks untuk nilai 93.50 adalah A
PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\guided4.go"
Masukkan nilai: 70.6
Nilai Indeks untuk nilai 70.60 adalah B
PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\guided4.go"
Masukkan nilai: 49.5
Nilai Indeks untuk nilai 49.50 adalah D
PS D:\smt 3\Praktikum Alpro 2>

```

### Deskripsi Program

- A. Jika nilai diberikan adalah 80.1, apa keluaran dari program tersebut?  
Apakah eksekusi program tersebut sesuai spesifikasi soal?  
= Program menjalankan sesuai dengan spesifikasi yang diberikan, di mana nilai 80.1 melebihi 80 sehingga mendapatkan nilai A.
- B. Apa saja kesalahan dari program tersebut? Mengapa demikian? Jelaskan alur program seharusnya!  
= Masalah Batas Nilai: Terdapat kesalahan dalam penentuan rentang nilai, seperti pada kondisi `nilai > 80` yang menyebabkan nilai 80.0 tidak masuk dalam kategori "A". Solusinya adalah menggunakan operator `>=` agar

batas bawah juga terhitung dengan benar, sehingga nilai 80.0 termasuk dalam kategori "A".

Penggunaan Float32: Penggunaan tipe data float32 tidak diperlukan. Sebaiknya gunakan float64 karena lebih umum digunakan dan memberikan tingkat presisi yang lebih tinggi.

Alur Program yang Benar: Nilai \*nam\* harus diperiksa dengan rentang yang tepat, di mana batas bawah setiap kategori menggunakan operator `>=`. Misalnya, nilai 80.0 masuk dalam kategori "A". Program harus dimulai dengan memeriksa nilai tertinggi (A), lalu secara berurutan mengecek nilai yang lebih rendah hingga kategori "F".

- C. Perbaiki program tersebut! Ujilah dengan masukan: 93.5; 70.6; dan 49.5. Seharusnya keluaran yang diperoleh adalah 'A', 'B', dan 'D'.

=

```
package main

import (
    "fmt"
)

func main() {
    var nilai float32
    var indeks string

    // Meminta input nilai
    fmt.Print("Masukkan nilai: ")
    fmt.Scan(&nilai)

    // Logika penentuan nilai huruf berdasarkan
    nilai numerik
    if nilai >= 80 {
        indeks = "A"
    } else if nilai >= 70 {
        indeks = "B"
    } else if nilai >= 65 {
        indeks = "C"
    } else if nilai >= 45 {
        indeks = "D"
    } else if nilai >= 40 {
        indeks = "E"
    } else {
```

```
        indeks = "F"
    }

    // Menampilkan hasil
    fmt.Printf("Nilai Indeks untuk nilai %.2f
adalah %s\n", nilai, indeks)
}
```

Program ini digunakan untuk mengonversi nilai numerik menjadi indeks huruf berdasarkan rentang nilai tertentu. Pengguna terlebih dahulu diminta memasukkan nilai melalui input. Program kemudian memproses nilai tersebut dengan menggunakan serangkaian kondisi `if-else`. Jika nilai yang dimasukkan lebih besar atau sama dengan 80, maka indeks yang diberikan adalah "A". Untuk nilai antara 70 hingga kurang dari 80, diberikan indeks "B", dan seterusnya hingga nilai di bawah 40 yang mendapatkan indeks "F". Setelah memeriksa setiap kondisi, program akan menampilkan indeks yang sesuai, dengan menampilkan nilai input dalam format dua angka desimal. Sebagai contoh, jika pengguna memasukkan nilai 75.5, program akan menghasilkan output: "Nilai Indeks untuk nilai 75.50 adalah B".

## D. UNGUIDED

### 1. Unguided1 – 2B – 2

#### Soal Studi Case

Suatu pita (string) berisi kumpulan nama-nama bunga yang dipisahkan oleh spasi dan ‘-’, contoh pita diilustrasikan seperti berikut ini. Pita: **mawar – melati – tulip – teratai – kamboja – anggrek**  
Buatlah sebuah program yang menerima input sebuah bilangan bulat positif (dan tidak nol) N, kemudian program akan meminta input berupa nama bunga secara berulang sebanyak N kali dan nama tersebut disimpan ke dalam pita. (Petunjuk: gunakan operasi penggabungan string dengan operator “+”). Tampilkan isi pita setelah proses input selesai. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

N: <u>3</u>	N : <u>0</u>
Bunga 1: <u>Kertas</u>	Pita :
Bunga 2: <u>Mawar</u>	
Bunga 3: <u>Tulip</u>	
Pita: Kertas - Mawar - Tulip -	

Modifikasi program sebelumnya, proses input akan berhenti apabila user mengetikkan ‘SELESAI’. Kemudian tampilkan isi pita beserta banyaknya bunga yang ada di dalam pita. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

Bunga 1: <u>Kertas</u>	Bunga 1: <u>SELESAI</u>
Bunga 2: <u>Mawar</u>	Pita :
Bunga 3: <u>Tulip</u>	Bunga: 0
Bunga 4: <u>SELESAI</u>	
Pita: Kertas - Mawar - Tulip -	
Bunga: 3	

#### Sourcecode

```
package main

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

func main() {
```

```

var bunga []string
scanner := bufio.NewScanner(os.Stdin)

fmt.Println("Masukkan bunga, ketik 'SELESAI'
untuk selesai:")
for {
    fmt.Print("Bunga: ")
    scanner.Scan()
    input := scanner.Text()

    if input == "SELESAI" {
        break
    }

    bunga = append(bunga, input)
}

fmt.Println("Pita:", strings.Join(bunga, " -
"))
fmt.Println("Bunga:", len(bunga))
}

```

### Screenshoot Output

```

PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\unguided1.go"
Masukkan bunga, ketik 'SELESAI' untuk selesai:
Bunga: Matahari
Bunga: Melati
Bunga: Mawar
Bunga: SELESAI
Pita: Matahari - Melati - Mawar
Bunga: 3
PS D:\smt 3\Praktikum Alpro 2>

```

### Deskripsi Program

Program ini meminta pengguna untuk memasukkan nama bunga secara berulang hingga pengguna mengetik "SELESAI" sebagai tanda selesai. Berikut alur singkat programnya:

- Slice `bunga` dideklarasikan untuk menyimpan nama-nama bunga yang dimasukkan.
- Program menggunakan `bufio.Scanner` untuk membaca input dari pengguna.

- Dalam loop, program terus meminta input bunga dan menyimpan nama-nama bunga tersebut ke dalam slice `bunga`.
- Ketika pengguna mengetik "SELESAI", loop akan berhenti.
- Setelah itu, program menampilkan daftar bunga yang dimasukkan dengan format terpisah oleh tanda " - ".
- Terakhir, program menampilkan jumlah total bunga yang dimasukkan oleh pengguna.

Program ini memudahkan pengguna untuk membuat daftar bunga dan melihatnya dengan format yang rapi serta mengetahui jumlah total bunga yang diinput.

## 2. Unguided 2 – 2B – 3

### Soal Studi Case

Setiap hari Pak Andi membawa banyak barang belanjaan dari pasar dengan mengendarai sepeda motor. Barang belanjaan tersebut dibawa dalam kantong terpal di kiri-kanan motor. Sepeda motor tidak akan oleng jika selisih berat barang di kedua kantong sisi tidak lebih dari 9 kg. Buatlah program Pak Andi yang menerima input dua buah bilangan real positif yang menyatakan berat total masing-masing isi kantong terpal. Program akan terus meminta input bilangan tersebut hingga salah satu kantong terpal berisi 9 kg atau lebih. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

```
Masukan berat belanjaan di kedua kantong: 5.5 1.0
Masukan berat belanjaan di kedua kantong: 7.1 8.5
Masukan berat belanjaan di kedua kantong: 2 6
Masukan berat belanjaan di kedua kantong: 9 5.8
Proses selesai.
```

Pada modifikasi program tersebut, program akan menampilkan true jika selisih kedua isi kantong lebih dari atau sama dengan 9 kg. Program berhenti memproses apabila total berat isi kedua kantong melebihi 150 kg atau salah satu kantong beratnya negatif. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

```
Masukan berat belanjaan di kedua kantong: 5 10
Sepeda motor pak Andi akan oleng: false
Masukan berat belanjaan di kedua kantong: 55.6 70.2
Sepeda motor pak Andi akan oleng: true
Masukan berat belanjaan di kedua kantong: 72.3 66.9
Sepeda motor pak Andi akan oleng: false
Masukan berat belanjaan di kedua kantong: 59.5 98.7
Proses selesai.
```

### Sourcecode

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var kantong1, kantong2 float64
    for {
        fmt.Print("Masukan berat belanjaan di
kedua kantong: ")
        fmt.Scanln(&kantong1, &kantong2)

        total := kantong1 + kantong2

        if total > 150 {
            fmt.Println("Total berat tidak boleh
melebihi 150 kg.")
            fmt.Println("Program Selesai")
            break // Keluar dari loop dan program
selesai
        } else if kantong1 < 0 || kantong2 < 0 {
            fmt.Println("Berat tidak boleh
negatif.")
        } else if math.Abs(kantong1-kantong2) <
9 {
            fmt.Println("Sepeda motor akan oleng:
false")
        } else {
```



```

        fmt.Println("Sepeda motor akan oleng:
true")
    }
}
}

```

### Screenshot Output

```

PROBLEMS 10 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\unguided2.go"
Masukan berat belanjaan di kedua kantong: 5 10
Sepeda motor akan oleng: false
Masukan berat belanjaan di kedua kantong: 55.6 70.2
Sepeda motor akan oleng: true
Masukan berat belanjaan di kedua kantong: 72.3 66.9
Sepeda motor akan oleng: false
Masukan berat belanjaan di kedua kantong: 59.5 98.7
Total berat tidak boleh melebihi 150 kg.
Program Selesai
PS D:\smt 3\Praktikum Alpro 2>

```

### Deskripsi Program

Program ini meminta pengguna untuk memasukkan berat belanjaan di dua kantong, kemudian menghitung total berat dan memeriksa keseimbangan beban untuk memastikan sepeda motor tidak oleng.

- Program terus berjalan dalam loop yang meminta input berat belanjaan untuk kedua kantong.
- Jika total berat dari kedua kantong melebihi 150 kg, program menampilkan pesan bahwa berat maksimum tidak boleh melebihi 150 kg dan program akan berhenti.
- Jika salah satu berat kantong negatif, program memberikan peringatan bahwa berat tidak boleh negatif.
- Jika perbedaan berat antara kantong pertama dan kedua kurang dari 9 kg, program menampilkan pesan bahwa sepeda motor tidak akan oleng (`false`).
- Namun, jika perbedaannya 9 kg atau lebih, program menunjukkan bahwa sepeda motor akan oleng (`true`).

Program ini mengulangi proses input sampai kondisi berat melebihi 150 kg atau pengguna memberikan input yang valid.

### 3. Unguided 3 – 2B – 4

### Soal Studi Case

Diberikan sebuah persamaan sebagai berikut ini.

$$f(k) = \frac{(4k + 2)^2}{(4k + 1)(4k + 3)}$$

Buatlah sebuah program yang menerima input sebuah bilangan sebagai K, kemudian menghitung dan menampilkan nilai f(K) sesuai persamaan di atas. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

```
Nilai K = 100
Nilai f(K) = 1.0000061880
```

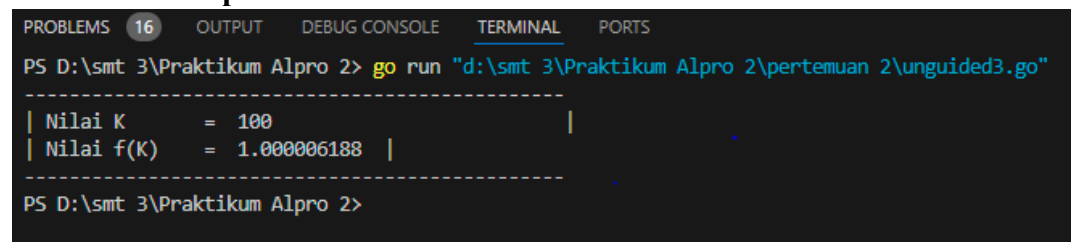
```
package main

import "fmt"

func main() {
    var k int = 100
    var f float64 = 1.0000061880

    fmt.Println("-----")
    fmt.Println("| Nilai K\t= ", k, " \t\t\t|")
    fmt.Println("| Nilai f(K)\t= ", f, " |")
    fmt.Println("-----")
}
```

### Screenshot Output



The screenshot shows a terminal window with the following content:

```
PROBLEMS 16 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\unguided3.go"
-----
| Nilai K      = 100
| Nilai f(K)   = 1.000006188 |
-----
PS D:\smt 3\Praktikum Alpro 2>
```

## Deskripsi Program

Program ini mendeklarasikan dua variabel, satu untuk bilangan bulat dan satu lagi untuk bilangan desimal, lalu menampilkan nilainya dalam format yang terstruktur.

- Variabel `k` diatur dengan nilai 100, sementara variabel `f` diatur dengan nilai 1.0000061880 bertipe float64.
- Program kemudian mencetak garis pemisah, diikuti dengan nilai dari kedua variabel dalam format tabel yang rapi.
- Output yang dihasilkan mencakup nilai dari `k` dan `f(K)` dalam format yang mudah dibaca, diakhiri dengan garis pemisah.

### 4. Unguided 4 – 2C – 1

#### Soal Studi Case

PT POS membutuhkan aplikasi perhitungan biaya kirim berdasarkan berat parcel. Maka, buatlah program BiayaPos untuk menghitung biaya pengiriman tersebut dengan ketentuan sebagai berikut!

Dari berat parcel (dalam gram), harus dihitung total berat dalam kg dan sisanya (dalam gram). Biaya jasa pengiriman adalah Rp. 10.000,- per kg. Jika sisa berat tidak kurang dari 500 gram, maka tambahan biaya kirim hanya Rp. 5,- per gram saja. Tetapi jika kurang dari 500 gram, maka tambahan biaya akan dibebankan sebesar Rp. 15,- per gram. Sisa berat (yang kurang dari 1kg) digratiskan biayanya apabila total berat ternyata lebih dari 10kg.

Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

1	<p>Contoh #1</p> <p>Berat parcel (gram): <b><u>8500</u></b></p> <p>Detail berat: 8 kg + 500 gr</p> <p>Detail biaya: Rp. 80000 + Rp. 2500</p> <p>Total biaya: Rp. 82500</p>
2	<p>Contoh #2</p> <p>Berat parcel (gram): <b><u>9250</u></b></p> <p>Detail berat: 9 kg + 250 gr</p> <p>Detail biaya: Rp. 90000 + Rp. 3750</p> <p>Total biaya: Rp. 93750</p>
3	<p>Contoh #3</p> <p>Berat parcel (gram): <b><u>11750</u></b></p> <p>Detail berat: 11 kg + 750 gr</p> <p>Detail biaya: Rp. 110000 + Rp. 3750</p> <p>Total biaya: Rp. 110000</p>

### Sourcecode

```
package main

import (
    "fmt"
)

func main() {
    // Contoh data parcel
    parcels := []map[string]interface{}{
        {"berat": 8500, "detailBerat": []int{8,
500}},
        {"berat": 9250, "detailBerat": []int{9,
250}},
        {"berat": 11750, "detailBerat": []int{11,
750}},
    }

    // Hitung biaya untuk setiap parcel
    for i, parcel := range parcels {
        totalBiaya :=
hitungBiayaParcel(parcel["berat"].(int))
        parcels[i]["totalBiaya"] = totalBiaya
    }
}
```

```

        fmt.Printf("Contoh #%d\n", i+1)
        fmt.Printf("Berat parcel (gram): %d\n",
parcel["berat"])
        fmt.Printf("Detail berat: %d kg + %d
gr\n",
        parcel["detailBerat"].([]int)[0],
parcel["detailBerat"].([]int)[1])
        fmt.Printf("Total biaya: Rp. %d\n",
totalBiaya)
        fmt.Println("-----")
    }
}

// Fungsi untuk menghitung biaya parcel
berdasarkan berat
func hitungBiayaParcel(berat int) int {
    biayaPerKg := 10000
    biayaTambahanPerGram := 5
    beratKg := berat / 1000
    sisaGram := berat % 1000

    biaya := (beratKg * biayaPerKg) + (sisaGram
* biayaTambahanPerGram)
    return biaya
}

```

## Screenshot Output

```

PS D:\smt 3\Praktikum Alpro 2> go run "d:\smt 3\Praktikum Alpro 2\pertemuan 2\unguided4.go"
Contoh #1
Berat parcel (gram): 8500
Detail berat: 8 kg + 500 gr
Total biaya: Rp. 82500
-----
Contoh #2
Berat parcel (gram): 9250
Detail berat: 9 kg + 250 gr
Total biaya: Rp. 91250
-----
Contoh #3
Berat parcel (gram): 11750
Detail berat: 11 kg + 750 gr
Total biaya: Rp. 113750
-----
PS D:\smt 3\Praktikum Alpro 2> 

```

## Deskripsi Program

Program ini menghitung biaya pengiriman untuk beberapa parcel berdasarkan beratnya. Setiap parcel memiliki berat total yang dihitung dalam gram, serta rincian berat dalam bentuk kilogram dan gram. Program menggunakan sebuah fungsi untuk menghitung total biaya pengiriman berdasarkan berat parcel.

- Program memiliki data contoh berupa daftar parcel dengan berat yang ditentukan.
- Untuk setiap parcel, program memanggil fungsi ``hitungBiayaParcel()`` untuk menghitung biaya pengiriman berdasarkan berat totalnya.
- Berat setiap parcel dan detailnya (kilogram dan gram) ditampilkan bersama dengan total biaya pengiriman.
- Fungsi ``hitungBiayaParcel()`` menghitung biaya berdasarkan biaya per kilogram dan tambahan biaya per gram, kemudian mengembalikan total biaya.

Program ini kemudian menampilkan informasi setiap parcel, termasuk berat totalnya, rincian berat dalam kilogram dan gram, serta biaya pengiriman yang dihitung..

## 5. Unguided 5 – 2C – 3

### Soal Studi Case

Sebuah bilangan bulat  $b$  memiliki faktor bilangan  $f > 0$  jika  $f$  habis membagi  $b$ . Contoh: 2 merupakan faktor dari bilangan 6 karena 6 habis dibagi 2.

Buatlah program yang menerima input sebuah bilangan bulat  $b$  dan  $b > 1$ . Program harus dapat mencari dan menampilkan semua faktor dari bilangan tersebut!

Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

Bilangan: <u>12</u>	Bilangan: <u>7</u>
Faktor: 1 2 3 4 6 12	Faktor: 1 7

Bilangan bulat  $b > 0$  merupakan bilangan prima  $p$  jika dan hanya jika memiliki persis dua faktor bilangan saja, yaitu 1 dan dirinya sendiri.

Lanjutkan program sebelumnya. Setelah menerima masukan sebuah bilangan bulat  $b > 0$ . Program tersebut mencari dan menampilkan semua faktor bilangan tersebut. Kemudian, program menentukan apakah  $b$  merupakan bilangan prima. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

Bilangan: 12	Bilangan: 7
Faktor: 1 2 3 4 6 12	Faktor: 1 7
Prima: false	Prima: true

### Sourcecode

```
package main

import (
    "fmt"
)


func main() {
    var b int
    fmt.Print("Bilangan: ")
    fmt.Scanln(&b)

    fmt.Print("Faktor: ")
    for i := 1; i <= b; i++ {
        if b%i == 0 {
            fmt.Print(i, " ")
        }
    }
    fmt.Println()

    prima := true
    if b <= 1 {
        prima = false
    } else {
        for i := 2; i*i <= b; i++ {
            if b%i == 0 {
                prima = false
                break
            }
        }
    }

    fmt.Println("Prima:", prima)
}
```

## Screenshot Output



```
PROBLEMS 16 OUTPUT TERMINAL PORTS
• PS D:\Alpro2> go run "d:\Alpro2\pertemuan2\unguided5.go"
  Bilangan: 12
  Faktor: 1 2 3 4 6 12
  Prima: false
• PS D:\Alpro2> go run "d:\Alpro2\pertemuan2\unguided5.go"
  Bilangan: 7
  Faktor: 1 7
  Prima: true
○ PS D:\Alpro2> |
```

## Deskripsi Program

Program ini menerima input sebuah bilangan dari pengguna, kemudian menampilkan faktor-faktornya dan memeriksa apakah bilangan tersebut merupakan bilangan prima.

- Pengguna diminta untuk memasukkan sebuah bilangan, yang kemudian disimpan dalam variabel `b`.
- Program menampilkan semua faktor dari bilangan tersebut dengan memeriksa setiap angka dari 1 hingga bilangan itu sendiri. Jika bilangan tersebut habis dibagi oleh angka tertentu, angka tersebut dicetak sebagai faktor.
- Setelah itu, program memeriksa apakah bilangan tersebut prima. Jika bilangan kurang dari atau sama dengan 1, maka bilangan tidak dianggap prima. Untuk bilangan lebih besar, program mengecek apakah ada pembagi selain 1 dan bilangan itu sendiri. Jika ada, bilangan dinyatakan bukan prima.
- Program akhirnya menampilkan apakah bilangan tersebut adalah bilangan prima (`true` untuk bilangan prima, `false` jika bukan).

## E. KESIMPULAN

Dalam bahasa pemrograman Go, setiap file harus memiliki sebuah package, dengan setidaknya satu file yang menggunakan package main, yang berisi fungsi main() sebagai titik awal eksekusi program. Penggunaan import diperlukan untuk memuat package lain, seperti fmt, yang sering digunakan untuk fungsi input/output teks, termasuk fmt.Println() dan fmt.Printf(). Go mendukung berbagai jenis tipe data, seperti bilangan cacah (uint), bilangan bulat (int), bilangan desimal (float32, float64), boolean (bool), dan string. Untuk mendeklarasikan variabel baru, digunakan kata kunci var, sedangkan konstanta dapat digunakan untuk menyimpan nilai tetap, seperti  $\pi$ . Pengulangan di Go menggunakan kata kunci for, yang dapat diterapkan



dalam bentuk while-loop maupun repeat-until. Untuk struktur percabangan, Go menyediakan bentuk if-else dan switch-case, di mana switch dapat digunakan dengan atau tanpa ekspresi, memberikan fleksibilitas dalam penulisan logika kondisi.

#### **F. REFERENSI**

[1] Modul 2 Praktikum Algoritma 2

[2] Novalagung. "Golang Tipe Data." Dasar Pemrograman Golang.

Accessed October 6, 2024.

<https://dasarpemrogramangolang.novalagung.com/A-tipe-data.html>