

**LAPORAN PRAKTIKUM  
ALGORITMA PEMROGRAMAN 2  
MODUL 2  
REVIEW STRUKTUR KONTROL**



**Oleh:**

**TSAQIF KANZ AHMAD  
2311102075  
IF-11-02**

**S1 TEKNIK INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO  
2024**

## I. DASAR TEORI

### A. Struktur Kontrol

Struktur kontrol adalah komponen penting dalam setiap Bahasa pemrograman yang mengatur alur eksekusi program. Golang mendukung berbagai struktur kontrol seperti perulangan (loop), pengkondisian (conditional statement) dan pengalihan (switch).

### B. Pengkondisian

Pada tipe data **if** dan **else** Pada Golang mendukung penggunaan pernyataan if-else yang mirip dengan bahasa pemrograman lain, tetapi tidak memerlukan tanda kurung disekitar kondisi.

- **If statement** digunakan untuk mengeksekusi blok kode tertentu terpenuhi.

```
if condition {  
    // kode yang dieksekusi jika kondisi true  
}
```

- **If-else statement** Menyediakan alternatif jika kondisi tidak terpenuhi

```
if condition {  
    // kode jika true  
} else {  
    // kode jika false  
}
```

- **else if statement** digunakan untuk mengevaluasi beberapa kondisi secara berurutan.

```
if condition1 {  
    // kode jika condition1 true  
} else if condition2 {  
    // kode jika condition2 true  
} else {  
    // kode jika semua kondisi false  
}
```

### C. Perulangan (Loops)

- **for loop** satu-satunya struktur perulangan Go yang dapat digunakan dalam berbagai cara mirip dengan perulangan for, while, dan do-while dalam bahasa lain.

```
// Bentuk dasar  
for i := 0; i < 10; i++ {  
    // kode yang dieksekusi  
}  
  
// Perulangan tanpa batas  
for {  
    // kode yang dieksekusi  
}
```

```
// Perulangan dengan kondisi
for condition {
    // kode yang dieksekusi
}
```

Tanpa ekspresi inisialisasi dan iterasi, `for` berfungsi sebagai loop tak terbatas. Golang tidak memiliki perulangan `while` secara eksplisit, tetapi loop `for` tanpa kondisi bisa digunakan sebagai pengganti `while`.

#### D. Pengalihan (Switch)

Golang memiliki implementasi **switch** yang sederhana tetapi sangat fleksibel. Salah satu keunikan **switch** dalam Go adalah tidak perlu menggunakan pernyataan **break** untuk menghentikan eksekusi, karena Go otomatis menghentikan eksekusi setelah satu case dijalankan (kecuali ada pernyataan **fallthrough**).

```
switch day {
case "Monday":
    fmt.Println("Hari Senin")
case "Tuesday":
    fmt.Println("Hari Selasa")
default:
    fmt.Println("Hari lain")
}
```

Golang juga mendukung evaluasi ekspresi yang lebih kompleks pada setiap case, sehingga `switch` dalam Golang bisa digunakan dalam berbagai situasi yang lebih fleksibel.

#### D. Kontrol Pengulangan

- **break**: Menghentikan eksekusi loop.
- **continue**: Melewatkan eksekusi iterasi saat ini dan melanjutkan ke iterasi berikutnya.
- **goto**: Memungkinkan lompat ke label yang dideklarasikan di dalam kode. Penggunaan `goto` dalam Go sebaiknya dibatasi karena dapat menyebabkan kode yang sulit dibaca.

#### E. Defer, Panic dan Recover

Golang juga memiliki mekanisme unik seperti `defer`, `panic`, dan `recover` yang terkait dengan pengelolaan aliran kontrol, khususnya dalam penanganan error:

- **defer**: Menjadwalkan fungsi untuk dijalankan setelah fungsi yang sedang berjalan selesai, berguna dalam pengelolaan sumber daya seperti file dan koneksi.
- **panic**: Digunakan untuk menghentikan eksekusi program dan mencetak pesan error.

- **recover:** Digunakan untuk menangani situasi panic dan mencegah program berhenti secara tidak terkendali.

#### **F. Concurrency dengan Goroutines dan Channels**

Golang memiliki fitur concurrency yang kuat dengan penggunaan goroutines dan channels untuk mengelola paralelisme:

- Goroutines: Fungsi ringan yang dijalankan secara asynchronous. Eksekusi goroutine dimulai dengan kata kunci go.
- Channels: Alat komunikasi antara goroutine untuk mengirim dan menerima data secara synchronous.

#### **G. Konsistensi dalam Penanganan Error**

Golang memiliki pendekatan eksplisit dalam menangani error. Semua fungsi diharapkan untuk mengembalikan error sebagai nilai kedua jika ada kesalahan, memungkinkan kontrol aliran program yang lebih jelas.

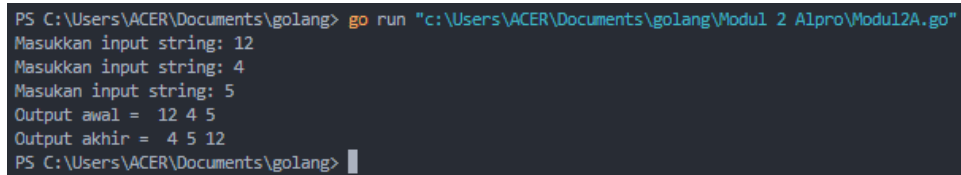
## I. GUIDED

- 1) Telusuri program berikut dengan cara mengkompilasi dan mengeksekusi program. Silahkan masukan data yang sesuai sebanyak yang diminta program. Perhatikan keluaran yang diperoleh. Coba terangkan apa sebenarnya yang dilakukan program tersebut.

```
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)
}
```

### SCREENSHOT OUTPUT :



```
PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\Modul2A.go"
Masukkan input string: 12
Masukkan input string: 4
Masukkan input string: 5
Output awal = 12 4 5
Output akhir = 4 5 12
PS C:\Users\ACER\Documents\golang> █
```

### DESKRIPSI PROGRAM :

*Pada program diatas dirancang untuk melakukan pertukaran nilai antara tiga variabel string yaitu 1, 2 dan 3. Program meminta pengguna memasukan tiga variabel, kemudian menyimpan input 3 variabel yang berbeda tersebut dari pengguna, program akan menukar nilai dari ketiga variabel tersebut menggunakan variabel sementara (temp) dan program akan menampilkan output nilai ketiga variabel tersebut setelah ditukar.*

- 2). Tahun kabisat adalah tahun yang habis dibagi 400 atau habis dibagi 4 tetapi tidak habis dibagi 100. **Buatlah sebuah program yang menerima input sebuah bilangan bulat.** Dan memeriksa apakah bilangan tersebut merupakan tahun kabisat (**true**) atau bukan (**false**)

1	Tahun: <u>2016</u> Kabisat: true
2	Tahun: <u>2000</u> Kabisat: true
3	Tahun: <u>2018</u> Kabisat: false

### SOURCE CODE

```
package main

import "fmt"

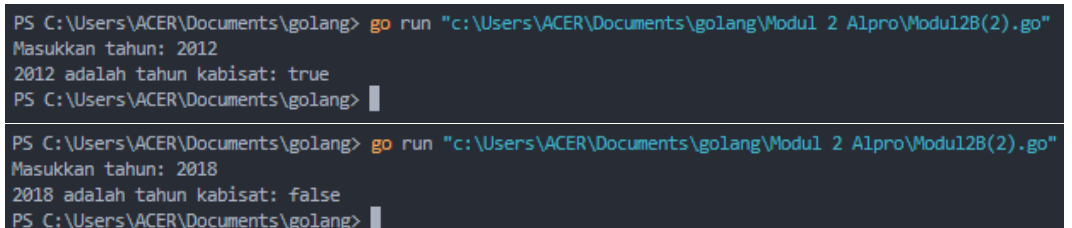
func main() {
    var tahun int

    fmt.Print("Masukkan tahun: ")
    fmt.Scanln(&tahun)

    kabisat := (tahun%400 == 0) || (tahun%4 == 0 &&
    tahun%100 != 0)

    fmt.Printf("%d adalah tahun kabisat: %t\n", tahun,
    kabisat)
}
```

### SCREENSHOT OUTPUT



```
PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\Modul2B(2).go"
Masukkan tahun: 2012
2012 adalah tahun kabisat: true
PS C:\Users\ACER\Documents\golang>

PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\Modul2B(2).go"
Masukkan tahun: 2018
2018 adalah tahun kabisat: false
PS C:\Users\ACER\Documents\golang>
```

### DESKRIPSI PROGRAM

*Program diatas adalah program yang menggunakan aturan tahun kabisat untuk menentukan apakah tahun tersebut merupakan tahun kabisat atau tidak. Program ini meminta input tahun dari pengguna, dan program memeriksa apakah tahun tersebut kabisat dengan menggunakan logika berdasarkan pembagian, dan menampilkan hasilnya dalam bentuk teks di layar.*

3). **Buat** program **bola** yang menerima input jari-jari suatu bola (Bilangan bulat). Tampilkan Volume dan Luas kulit bola.  $Volumebola = \frac{4}{3}\pi r^3$  dan  $luasbola = 4\pi r^2$

Jejari = 5

Bola dengan jejari 5 memiliki volume 523.5988 dan luas kulit 314.1593

#### SOURCE CODE

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var radius float64

    fmt.Print("Masukkan jari-jari bola: ")
    fmt.Scanln(&radius)

    volume := (4.0 / 3.0) * math.Pi * math.Pow(radius,
3)
    area := 4 * math.Pi * math.Pow(radius, 2)

    fmt.Printf("Bola dengan jari-jari %.2f memiliki
volume %.4f dan luas permukaan %.4f\n", radius, volume,
area)
}
```

#### SCREENSHOT OUTPUT

```
PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2A\Modul2A(3).go"
Masukkan jari-jari bola: 7
Bola dengan jari-jari 7.00 memiliki volume 1436.7550 dan luas permukaan 615.7522
PS C:\Users\ACER\Documents\golang> |
```

#### DESKRIPSI PROGRAM

Pada Program tersebut program meminta pengguna memasukan jari-jari bola dalam bilangan bulat. Setelah pengguna memasukan jari-jari, program akan menghitung volume dan luas permukaan bola menggunakan rumus Volume bola  $= \frac{4}{3}\pi r^3$  dan rumus Luas permukaan bola (A)  $= 4\pi r^2$ . Program ini menggunakan paket math untuk mengakses nilai  $\pi$ .

## II. UNGUIDED

### MODUL 2C

1). 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,-pergram saja. Tetapi jika kurang dari 500 gram, maka tambahan biaya akan dibebankan sebesar Rp. 15, - pergram. Sisa berat (yang kurang dari 1kg) digratiskan biaya apabila total berat ternyata lebih dari 10kg. Perhatikan contoh sesi interaksi program dibawah ini.

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

### SOURCE CODE

```
package main

import (
    "fmt"
)

func main() {
    var beratParsel int
    var beratKg, sisaBerat int
    var biayaKg, biayaSisa, totalBiaya int

    fmt.Print("Masukkan berat parcel (gram): ")
    fmt.Scanln(&beratParsel)
```



```

beratKg = beratParsel / 1000
sisBerat = beratParsel % 1000

biayaKg = beratKg * 10000

if beratKg > 10 {
    biayaSisa = 0
} else if sisBerat >= 500 {
    biayaSisa = sisBerat * 5
} else {
    biayaSisa = sisBerat * 15
}

totalBiaya = biayaKg + biayaSisa

fmt.Println("Detail berat:", beratKg, "kg +", sisBerat,
"gr")
fmt.Println("Detail biaya: Rp.", biayaKg, "+ Rp.",
biayaSisa)
fmt.Println("Total biaya: Rp.", totalBiaya)
}

```

## SCREENSHOT OUTPUT

```

PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2C\Unguided1c.go"
Masukkan berat parsel (gram): 12543
Detail berat: 12 kg + 543 gr
Detail biaya: Rp. 120000 + Rp. 0
Total biaya: Rp. 120000
PS C:\Users\ACER\Documents\golang>

PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2C\Unguided1c.go"
Masukkan berat parsel (gram): 2800
Detail berat: 2 kg + 800 gr
Detail biaya: Rp. 20000 + Rp. 4000
Total biaya: Rp. 24000
PS C:\Users\ACER\Documents\golang>

```

## DESKRIPSI PROGRAM

*Pada program tersebut adalah program untuk menghitung biaya pengiriman berdasarkan berat parsel dengan tarif tertentu untuk sisa gram tergantung jumlah total berat. Aturan ini memastikan bahwa biaya tambahan diperhitungkan secara akurat, dan parsel dengan berat lebih dari 10 kg mendapatkan pembebasan biaya untuk sisa gram.*

- 2). Diberikan sebuah nilai akhir mata kuliah (NAM) [0..100] dan standar penilaian 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

Program berikut menerima input sebuah bilangan riil yang menyatakan NAM. Program menghitung NMK dan menampilkannya.

```
1 package main
2 import "fmt"
3 func main() {
4     var nam float64
5     var nmk string
6     fmt.Print("Nilai akhir mata kuliah: ")
7     fmt.Scanln(&nam)
8     if nam > 80 {
9         nmk = "A"
10    }
11    if nam > 72.5 {
12        nmk = "AB"
13    }
14    if nam > 65 {
15        nmk = "B"
16    }
17    if nam > 57.5 {
18        nmk = "BC"
19    }
20    if nam > 50 {
21        nmk = "C"
22    }
23    if nam > 40 {
24        nmk = "D"
25    } else if nam <= 40 {
26        nmk = "E"
27    }
28    fmt.Println("Nilai mata kuliah: ", nmk)
29 }
```

Jawablah pertanyaan-pertanyaan berikut :

- a. Jika **nam** diberikan adalah 80.1, apa keluaran dari program tersebut?  
Apakah eksekusi program tersebut sesuai spesifikasi soal?

**Jawab :** output dari program tersebut adalah A dan eksekusi program tersebut sesuai spesifikasi soal karena nilai 80,1 termasuk dalam rentang nilai A.

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

**Jawab : Kesalahan pada program tersebut :**

- **Sintaksis If yang tidak benar :** Penggunaan if tanpa kurung kurawal untuk blok kode yang terdiri dari lebih dari satu baris adalah salah.
- **Logika perbandingan yang tidak lengkap:** Beberapa kondisi perbandingan tidak mencakup semua kemungkinan nilai NAM.
- **Variabel nak tidak terdefinisi:** Variabel ini digunakan untuk menyimpan nilai NMK, namun tidak pernah dideklarasikan.

**Alur program yang seharusnya :**

- Meminta input nilai akhir mata kuliah.
- Membandingkan nilai input dengan rentang nilai yang telah ditentukan.
- Menentukan nilai huruf berdasarkan rentang nilai yang sesuai.
- Menampilkan nilai huruf.

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

**Jawab :**

**SOURCE CODE**

```
package main

import "fmt"

func main() {
    var nam float64
    var nmk string
    fmt.Print("Nilai akhir mata kuliah: ")
    fmt.Scanln(&nam)
    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"
    }
    fmt.Println("Nilai mata kuliah: ", nmk)
}
```

## SCREENSHOT OUTPUT

```
Nilai akhir mata kuliah: 93.5
Nilai mata kuliah: A
PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2C\Unguided2C(2).go"
Nilai akhir mata kuliah: 70.6
Nilai mata kuliah: B
PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2C\Unguided2C(2).go"
Nilai akhir mata kuliah: 49.5
Nilai mata kuliah: D
PS C:\Users\ACER\Documents\golang> 
```

## DESKRIPSI PROGRAM

*Program perbaikan tersebut terdapat deklarasi nmk sebagai string untuk menyimpan nilai huruf. Penggunaan else if digunakan memeriksa rentang nilai secara berurutan. Setiap blok if dan else menggunakan kurung kurawal untuk membungkus kode yang akan dieksekusi jika kondisi terpenuhi. Kondisi perbandingan disesuaikan agar mencakup semua kemungkinan nilai NAM.*

3). 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 dibawah ini :

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 dibawah ini :

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

## SOURCE CODE

```
package main

import "fmt"

func main() {
    var b int
    var prisma bool
    fmt.Print("Bilangan: ")
    fmt.Scan(&b)
    fmt.Print("Faktor: ")
    for i := 1; i <= b; i++ {
        if b%i == 0 {
            fmt.Print(i, " ")
        }
    }
    fmt.Println()
    if b < 2 {
        prisma = false
    } else {
```

```

        prisma = true
    }
    for i := 2; i*i <= b; i++ {
        if b%i == 0 {
            prisma = false
        }
    }
    if prisma == true {
        fmt.Println("Prima: true")
    } else {
        fmt.Println("Prima: false")
    }
}

```

## SCREENSHOT OUTPUT

```

PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2C\Unguided2C(3).go"
Bilangan: 14
Faktor: 1 2 7 14
Prima: false
PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2C\Unguided2C(3).go"
Bilangan: 11
Faktor: 1 11
Prima: true
PS C:\Users\ACER\Documents\golang> 

```

## DESKRIPSI PROGRAM

*Program tersebut adalah program yang berfungsi menampilkan faktor-faktor dari bilangan yang dimasukan oleh pengguna dan menentukan apakah bilangan tersebut merupakan bilangan prima atau bukan. Program meminta pengguna untuk memasukan sebuah bilangan bulat. Program akan menghitung dan menampilkan semua faktor dari bilangan tersebut. Jika bilangan kurang dari 2, maka bilangan tersebut bukan bilangan prima. Jika bilangan 2 atau lebih, program menganggap bilangan prima kecuali ditemukan pembagi selain 1 dan bilangan itu sendiri. dan Program akan menampilkan apakah bilangan tersebut adalah prima (true) atau bukan (false).*

## MODUL 2B

1). Siswa kelas IPA di salah satu sekolah menengah atas di Indonesia sedang mengadakan praktikum kimia. Disetiap percobaan akan menggunakan 4 tabung reaksi, yang mana susunan warna cairan disetiap 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 berurutan adalah 'merah', 'hijau', dan 'ungu' selama 5 kali percobaan berulang.

**Buatlah Program yang menerima input berupa warna dari 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 dibawah ini :**

```
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

Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: ungu kuning hijau merah
Percobaan 5: merah kuning hijau ungu
BERHASIL: false
```

## SOURCE CODE

```
package main

import "fmt"

func main() {
    var a, b, c, d string
    berhasil := true
    var i int // Inisialisasi variabel i

    for i < 5 { // Loop selama i kurang dari 6
        fmt.Printf("percobaan %d: ", i+1)
        fmt.Scanln(&a, &b, &c, &d)

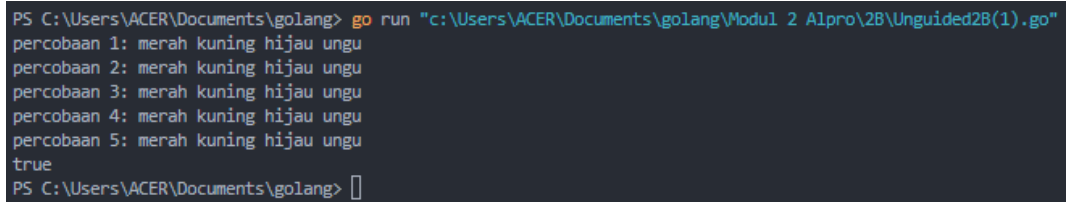
        // Memeriksa apakah semua variabel tidak
        // sesuai dengan kondisi yang diinginkan
        if a != "merah" && b != "kuning" && c !=
            "hijau" && d != "ungu" {
```

```
        berhasil = false
    }

    i++ // Increment counter
}

// Menampilkan hasil
fmt.Println(berhasil)
}
```

## SCREENSHOT OUTPUT



```
PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2B\Unguided2B(1).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
true
PS C:\Users\ACER\Documents\golang> 
```

## DESKRIPSI PROGRAM

*Program tersebut meminta pengguna memasukkan empat input (warna) sebanyak 5 kali. Pada setiap percobaan, program memeriksa apakah keempat input tersebut **bukan**: "merah", "kuning", "hijau", dan "ungu". Jika ada satu kali percobaan di mana semua input tidak sesuai, maka variabel berhasil diubah menjadi false. Setelah 5 kali percobaan, program mencetak hasil true jika semua input sesuai, dan false jika ada yang tidak sesuai.*



2). 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 kedalam pita.**

*(Petunjuk: gunakan operasi penggabungan string dengan operator "+")*

Tampilkan isi pita setelah proses input selesai.

Perhatikan contoh sesi interaksi program seperti dibawah ini :

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 mengetikan 'SELESAI'. Kemudian tampilkan isi pita beserta banyaknya bunga yang ada di dalam pita. Perhatikan contoh sesi interaksi program seperti dibawah ini :

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	

## SOURCE CODE

```
package main

import (
    "fmt"
    "strings"
)

func main() {
    var n int
    var bunga string
    var pita []string

    // Meminta input jumlah bunga N
```

```

fmt.Print("N: ")
fmt.Scan(&n)

// Proses input bunga sebanyak N kali
for i := 1; i <= n; i++ {
    fmt.Printf("Bunga %d: ", i)
    fmt.Scan(&bunga)
    // Menyimpan nama bunga ke dalam slice pita
    pita = append(pita, bunga)
}

// Menampilkan pita setelah input selesai
fmt.Println("Pita:", strings.Join(pita, " - "))

// Modifikasi: Berhenti jika input adalah
"SELESAI"
pita = nil // Reset pita untuk sesi berikutnya
for {
    fmt.Printf("Bunga %d: ", len(pita)+n+1)
    fmt.Scan(&bunga)

    // Jika user mengetik "SELESAI", proses
    berhenti
    if strings.ToUpper(bunga) == "SELESAI" {
        break
    }

    // Menyimpan nama bunga ke dalam slice pita
    pita = append(pita, bunga)
}

// Menampilkan pita terakhir dan jumlah bunga
fmt.Println("Pita:", strings.Join(pita, " - "))
fmt.Println("Total Bunga:", len(pita)+n)
}

```

## SCREENSHOT OUTPUT

```

PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2B\Unguided2B(2).go"
N: 3
Bunga 1: Mawar
Bunga 2: Anggrek
Bunga 3: teratai
Pita: Mawar - Anggrek - teratai
Bunga 4: selesai
Pita:
Total Bunga: 3
PS C:\Users\ACER\Documents\golang>

```

## **DESKRIPSI PROGRAM**

*Program tersebut adalah program untuk menerima input  $N$  dan meminta input nama bunga sebanyak  $N$  kali.*

*Program meminta pengguna memasukkan bilangan bulat positif  $N$ , yang mewakili jumlah bunga yang akan diinputkan.*

*Jika  $N$  kurang dari atau sama dengan 0, program akan menampilkan pesan kesalahan dan berhenti. Saat*

*Program memasuki loop untuk meminta pengguna memasukkan nama bunga sebanyak  $N$  kali.*

*Jika pengguna memasukkan "SELESAI", loop akan berhenti. Nama-nama bunga ini disimpan dalam string yang digabungkan menjadi satu yang dipisahkan oleh " – ".*

*Program mencetak string hasil penggabungan nama-nama bunga (pita) dan*

*Program juga mencetak jumlah bunga yang dimasukkan sebelum "SELESAI" atau mencapai  $N$ .*

3). 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 dibawah ini:

```
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 dibawah ini :

```
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.
```

## SOURCE CODE

```
package main

import "fmt"

func main() {
    var a, b, val float32
    var isTrue bool

    for {
        fmt.Print("masukkan berat di kedua kantong: ")
        fmt.Scanln(&a, &b)

        val = a + b
```

```

        if val > 150 || a < 0 || b < 0 {
            break
        }

        if a < 9.0 || b < 9.0 {
            isTrue = true
        } else {
            isTrue = false
        }

        fmt.Println("Sepeda motor pak Andi akan oleng:",
isTrue)
    }

    fmt.Println("program selesai")
}

```

## SCREENSHOT OUTPUT

```

PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2B\Unguided2B(3).go"
Masukan berat belanjaan di kedua kantong: 5.8 7.0
Sepeda motor pak Andi oleng: false
Masukan berat belanjaan di kedua kantong: 55.6 70.2
Sepeda motor pak Andi oleng: true
Masukan berat belanjaan di kedua kantong: 

```

## DESKRIPSI PROGRAM

Pada program tersebut bertujuan untuk membantu Pak Andi dalam menyeimbangkan berat belanjaan di kedua kantong terpal yang dibawa disepeda motornya. Program ini menerima input dua bilangan real positif yang menyatakan berat masing-masing kantong terpal. Program akan terus meminta input hingga salah satu kantong terpal berisi 9 kg atau lebih, atau total berat kedua kantong melebihi 150 kg, atau salah satu kantong beratnya negatif. Pada modifikasi program, program akan menampilkan true jika selisih berat ke dua kantong lebih dari atau sama dengan 9 kg, dan false jika tidak. Program akan berhenti memproses jika total berat kedua kantong melebihi 150 kg atau salah satu kantong beratnya negatif.

4). Diberikan sebuah persamaan sebagai berikut.

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

Buatlah sebuah program yang menerima input sebuah bilangan sebagai **K**, kemudian menghitung dan menampilkan nilai  $f(K)$  sesuai persamaan diatas. Perhatikan contoh sesi interaksi program seperti dibawah ini :

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

$\sqrt{2}$  merupakan bilangan irasional. Meskipun demikian, nilai tersebut dapat dihamperi dengan rumus berikut :

$$\sqrt{2} = \prod_{k=0}^{\infty} \frac{(4k + 2)^2}{(4k + 1)(4k + 3)}$$

Modifikasi program sebelumnya yang menerima input integer **K** dan menghitung  $\sqrt{2}$  untuk **K** tersebut. Hampiran  $\sqrt{2}$  dituliskan dalam ketelitian 10 angka dibelakang koma. Perhatikan contoh sesi interaksi program seperti dibawah ini :

1	Nilai K = <u>10</u> Nilai akar 2 = 1.4062058441
2	Nilai K = <u>100</u> Nilai akar 2 = 1.4133387072
3	Nilai K = <u>1000</u> Nilai akar 2 = 1.4141252651

#### SOURCE CODE

```
package main

import (
    "fmt"
    "math"
)

// Fungsi untuk menghitung f(K)
```

```

func f(K int) float64 {
    numerator := math.Pow(float64(4*K + 2), 2)
    denominator := float64((4*K + 1) * (4*K + 3))
    return numerator / denominator
}

// Fungsi untuk menghampiri sqrt(2) menggunakan rumus
yang diberikan
func approximateSqrt2(K int) float64 {
    product := 1.0
    for k := 0; k <= K; k++ {
        product *= f(k)
    }
    return product
}

func main() {
    var K int

    // Bagian pertama: Menghitung f(K)
    fmt.Print("Masukkan nilai K: ")
    fmt.Scan(&K)
    fmt.Printf("Nilai f(K) = %.10f\n", f(K))

    // Bagian kedua: Menghitung hampiran sqrt(2)
    fmt.Print("Masukkan nilai K untuk menghitung akar 2: ")
    fmt.Scan(&K)
    fmt.Printf("Nilai akar 2 = %.10f\n",
approximateSqrt2(K))
}

```

## SCREENSHOT OUTPUT

```

PS C:\Users\ACER\Documents\golang> go run "c:\Users\ACER\Documents\golang\Modul 2 Alpro\2B\Unguided2B(4).go"
Masukkan nilai K: 12
Nilai f(K) = 1.0004001601
Masukkan nilai K untuk menghitung akar 2: 1200
Nilai akar 2 = 1.4141399687
PS C:\Users\ACER\Documents\golang>

```

## DESKRIPSI PROGRAM

Program tersebut merupakan program menghitung nilai  $f(K)$  dan menghitung nilai akar 2. Pada perhitungan nilai  $f(K)$  terdapat fungsi  $f(K)$  yang menerima input integer  $K$  dan menghitung nilai  $f(K)$  sesuai dengan persamaan yang diberikan  $f(K) = (4K + 2)^2 / ((4K + 1)(4K + 3))$ . Program meminta pengguna untuk memasukan nilai  $K$  dan kemudian menampilkan hasil perhitungan  $f(K)$  dengan presisi 10 angka di belakang koma. Pada perhitungan akar 2 Fungsi

*approximateSqrt2(K)* menghitung nilai hampiran 2 menggunakan rumus produk yang diberikan  $\sqrt{2} = \prod (4k + 2)^2 / ((4k + 1)(4k + 3))$ . Program meminta pengguna untuk memasukkan nilai  $K$  dan kemudian menampilkan hasil perhitungan hampiran 2 dengan presisi 10 angka di belakang koma. Program ini menggunakan perulangan untuk menghitung fungsi dari  $f(k)$  hingga nilai  $K$  yang diberikan, dan menampilkan hasil output sesuai dengan contoh gambar yang diberikan.