LAPORAN PRAKTIKUM
ALGORITMA PEMROGRAMAN II

MODUL II

“REVIEW STRUKTUR KONTROL”

Oleh:

ZAHRINA ANTIKA MALAHATI

2311102109

IF 11 02

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

A. Pengertian

Go (Golang) adalah bahasa pemrograman yang dirancang oleh Google. Dikenal karena kesederhanaan, efisiensi, dan dukungannya terhadap konkurensi (kemampuan menjalankan banyak tugas secara bersamaan), Go telah menjadi pilihan populer bagi banyak pengembang.

Setelah memahami apa itu golang, berikut beberapa fungsi dari bahasa pemrograman golang:

- Membantu membangun tim developer yang lebih *scalalable*
- Mengembangkan teknologi penyimpanan berbasis online dengan media penyimpanan yang besar
- Merancang aplikasi dengan basis web yang memiliki keamanan tinggi
- Membangun sebuah sistem yang memiliki kinerja tinggi dan lebih kompleks
- Mengembangkan kode server pada jaringan web server dan layanan mikro

Pada modul II ini kita telah belajar terkait struktur program Go, dalam kerangka program yang ditulis dalam bahasa pemrograman Go, program utama sealalu mempunyai dua komponen berikut :

- **package main** merupakan penanda bahwa file ini berisi program utama
- **func main()** berisi kode utama dari sebuah program Go.

Komentar, bukan bagian dari kode program, dan dapat ditulis di mana saja di dalam program :

- Satu teks baris yang diawali dengan garis miring ganda (‘//’) s.d. akhir baris, atau.
- Beberapa baris teks yang dimulai dengan pasangan karakter ‘/*’ dan diakhiri dengan ‘*/’.

```
// Setiap program utama dimulai dengan "package main"
package main

// Impor paket yang dibutuhkan, "fmt" berisi proses I/O standar
import “fmt”

// Kode program utama dalam "fungsi main"
func main() {
    …
}
```


Contoh sebuah program dalam bahasa pemrograman Go (nama file hello.go).

```
package main
import "fmt"
func main() {
    var greetings = "Selamat datang di dunia DAP"
    var a, b int

    fmt.Println(greetings)
    fmt.Scanln(&a, &b)
    fmt.Printf("%v + %v = %v\n", a, b, a+b)
}
```

B. Koding, Kompilasi, dan Eksekusi Go

- **Koding**

  Tidak berbeda dengan penulisan program sumber dalam bahasa lain, program Go harus dibuat menggunakan penyunting teks dan disimpan dalam format teks, bukan dalam format dokumen (doc, docx, atau lainnya). Setiap satu program Go disimpan dalam satu folder tersendiri. Nama folder merupakan nama program tersebut, karena itu secara prinsip, satu program Go dapat dipecah dalam beberapa file ekstensi*.go selama disimpan dalam folder yang sama.

- **Kompilasi**

  Kompilasi dalam bahasa Go adalah proses menerjemahkan kode sumber Go menjadi kode mesin yang dapat dijalankan komputer. Proses ini melibatkan beberapa tahap, seperti analisis leksikal, penguraian, analisis semantik, pengoptimalan, dan pembuatan kode. Go diimplementasikan sebagai kompilator.

- **Eksekusi Go**

  Eksekusi program Go menggunakan kompilasi ter-otomatisasi, sehingga kode Go akan dicompile menjadi biner yang siap dijalankan di mesin target. Semua proses terkait bahasa Go dilakukan melalui utilitas go. Beberapa opsi dengan utilitas go yaitu ada **go build**, **go build file.go**, **go fmt**, **go clean**.

II. GUIDED

❖ Guided 1

Soal :

Soal Latihan Modul 2A

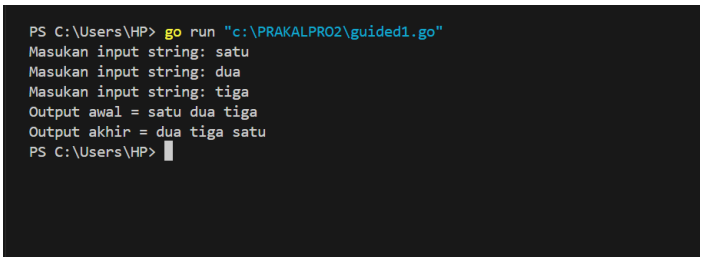
1. Telusuri program berikut dengan cara mengkompilasi dan mengeksekusi program. Silakan masukan data yang sesuai sebanyak yang diminta program. Perhatikan keluaran yang diperoleh. Coba terangkan apa sebenarnya yang dilakukan program tersebut?

```
1  package main
2  import "fmt"
3
4  func main() {
5      var (
6          satu, dua, tiga string
7          temp string
8      )
9      fmt.Print("Masukan input string: ")
10     fmt.Scanln(&satu)
11     fmt.Print("Masukan input string: ")
12     fmt.Scanln(&dua)
13     fmt.Print("Masukan input string: ")
14     fmt.Scanln(&tiga)
15     fmt.Println("Output awal = " + satu + " " + dua + " " + tiga)
16     temp = satu
17     satu = dua
18     dua = tiga
19     tiga = temp
20     fmt.Println("Output akhir = " + satu + " " + dua + " " + tiga)
21 }
```

Source code

```
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)
}
```

Output



Deskripsi

Program ini memberikan pemahaman dasar tentang bagaimana komputer dapat memanipulasi data dengan cara yang terstruktur. Konsep penukaran nilai yang sederhana ini merupakan salah satu fondasi penting dalam pemrograman.

Program ini dirancang untuk melakukan **penukaran nilai** dari tiga variabel string. Bayangkan Anda memiliki tiga kotak dengan label "satu", "dua", dan "tiga". Setiap kotak berisi sebuah kata. Program ini akan mengambil kata-kata dari ketiga kotak tersebut, lalu menukar posisi kata-kata di dalam kotak-kotak itu.

Langkah-langkah Kerja Program:

1. **Meminta Input:**
   Program meminta Anda untuk memasukkan tiga kata. Kata-kata ini akan disimpan di dalam kotak-kotak yang telah disebutkan tadi.
2. **Menampilkan Output Awal:**
   Setelah Anda memasukkan kata-kata, program akan menampilkan ketiga kata tersebut dalam urutan semula. Ini seperti menunjukkan kepada Anda isi dari ketiga kotak sebelum dilakukan penukaran.
3. **Menukar Nilai:**
   - Bagian yang paling menarik adalah ketika program melakukan penukaran nilai. Bayangkan Anda memiliki kotak tambahan (variabel temp) untuk menyimpan sementara sebuah kata.
   - Pertama, kata di kotak "satu" dipindahkan ke kotak tambahan.
   - Kemudian, kata di kotak "dua" dipindahkan ke kotak "satu".
   - Selanjutnya, kata di kotak "tiga" dipindahkan ke kotak "dua".
   - Terakhir, kata yang disimpan di kotak tambahan (yang awalnya dari kotak "satu") dipindahkan ke kotak "tiga".
4. **Menampilkan Output Akhir:**
   Setelah penukaran selesai, program akan menampilkan kembali ketiga kata tersebut. Kali ini, urutan kata-kata sudah berbeda karena telah terjadi penukaran.

❖ Guided 2

Soal

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**).

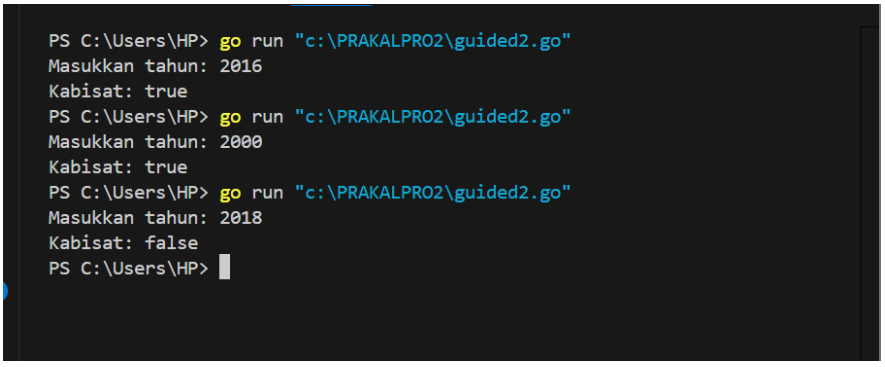
   (Contoh input/output, **Teks bergaris bawah** adalah input dari user):

1	Tahun: **2016** Kabisat: true
2	Tahun: **2000** Kabisat: true
3	Tahun: **2018** Kabisat: false

Source code

```
package main

import (
    "fmt"
)

func main() {
    var tahun int

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

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

    fmt.Printf("Kabisat: %t\n", kabisat)
}
```

Output



Deskripsi

Program ini adalah program dengan bahasa Go untuk apakah suatu tahun merupakan tahun kabisat atau bukan. Program ini menerima input berupa angka tahun (integer) dari pengguna, kemudian melakukan

perhitungan berdasarkan logika tahun kabisat. Cara kerja dari program ini yaitu,

1. Input Tahun: Program meminta pengguna untuk memasukkan tahun yang ingin diperiksa.
2. Perhitungan: Program melakukan dua pemeriksaan:
   - Habis Dibagi 400: Jika tahun tersebut habis dibagi 400, maka dipastikan tahun kabisat.
   - Habis Dibagi 4 dan Tidak Habis Dibagi 100: Jika tahun tersebut habis dibagi 4 tetapi tidak habis dibagi 100, maka juga merupakan tahun kabisat.
3. Output: Program akan mencetak hasil berupa "Kabisat: true" jika tahun tersebut adalah tahun kabisat, atau "Kabisat: false" jika bukan tahun kabisat.

❖ Guided 3

Soal

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 $luasbola = 4\pi r^2$ ($\pi \approx 3.1415926535$).

   (Contoh input/output, **Teks bergaris bawah** adalah input dari user):

   Jejari = **5**

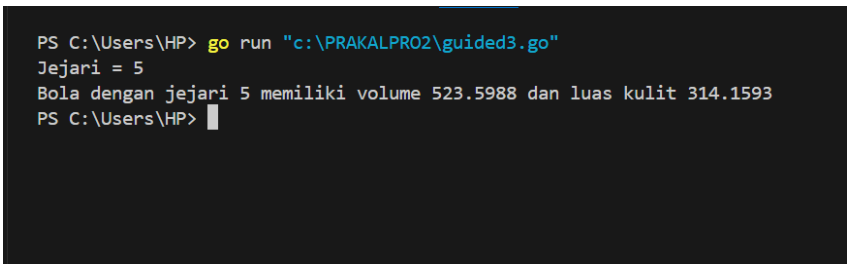
   Bola dengan jejari 5 memiliki volume 523.5988 dan luas kulit 314.1593

Source code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var r float64

    fmt.Print("Jejari = ")
    fmt.Scan(&r)

    volumeBola := (4.0 / 3.0) * math.Pi * math.Pow(r, 3)
    luasBola := 4 * math.Pi * math.Pow(r, 2)

    fmt.Printf("Bola dengan jejari %.0f memiliki volume %.4f
dan luas kulit %.4f\n", r, volumeBola, luasBola)
```

```
}
```

Output



Deskripsi

Program ini adalah program dengan bahasa Go yang dirancang Program ini dirancang untuk menghitung volume dan luas permukaan bola berdasarkan jari-jari yang diberikan oleh pengguna. Program ini memanfaatkan konsep-konsep dasar matematika seperti nilai pi (π) dan perpangkatan untuk menghitung kedua nilai tersebut.

III. UNGUIDED

❖ Unguided 1 (Latihan 2B no 1)

Soal

Soal Latihan Modul 2B

1. 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 berturutan 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):

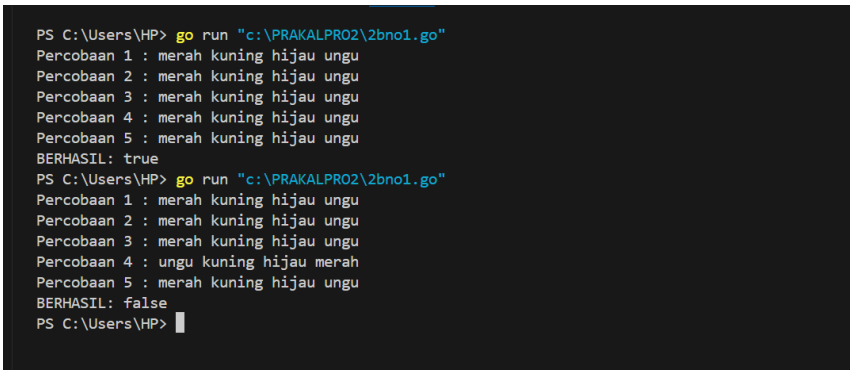
```
Percobaan 1: merah   kuning   hijau   ungu
Percobaan 2: merah   kuning   hijau   ungu
Percobaan 3: merah   kuning   hijau   ungu
Percobaan 4: merah   kuning   hijau   ungu
Percobaan 5: merah   kuning   hijau   ungu
BERHASIL: true

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"
    "strings"
)

// Fungsi untuk memeriksa apakah urutan warna benar
func cekPercobaan(input [5][4]string) bool {
    urutanBenar := [4]string{"merah", "kuning", "hijau",
"ungu"}

    for i := 0; i < 5; i++ {
        for j := 0; j < 4; j++ {
```

```
                    if   strings.ToLower(input[i][j])   !=
urutanBenar[j] {
                return false
            }
        }
    }
    return true
}

func main() {
    // Input percobaan
    var input [5][4]string

    // Masukkan data percobaan dari pengguna
    for i := 0; i < 5; i++ {
        fmt.Printf("Percobaan %d : ", i+1)
        for j := 0; j < 4; j++ {
            fmt.Scan(&input[i][j])
        }
    }

    // Memeriksa hasil percobaan
    if cekPercobaan(input) {
        fmt.Println("BERHASIL: true")
    } else {
        fmt.Println("BERHASIL: false")
    }
}

//Zahrina Antika Malahati_2311102109
```

Output



Deskripsi

Program ini adalah sebuah program dengan bahasa Go untuk memverifikasi susunan warna dalam percobaan kimia. Program tersebut akan menerima input berupa warna dari 4 gelas untuk 5 kali percobaan dan memeriksa apakah urutan warna tersebut sesuai dengan urutan yang benar yaitu: "merah", "kuning", "hijau", "ungu".

- Program ini menggunakan array 2 dimensi untuk menyimpan input dari 5 percobaan dengan masing-masing 4 warna.
- Fungsi cekPercobaan memeriksa apakah urutan warna untuk setiap percobaan sesuai dengan urutan yang benar, yaitu: "merah", "kuning", "hijau", "ungu".
- Input dimasukkan secara interaktif oleh pengguna melalui terminal.
- Setelah semua percobaan dimasukkan, program akan menampilkan hasil apakah urutan benar atau tidak dengan mencetak BERHASIL: true atau BERHASIL: false.

❖ Unguided 2 (Latihan 2B no 2)

Soal

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 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: **3** Bunga 1: **Kertas** Bunga 2: **Mawar** Bunga 3: **Tulip** Pita: Kertas – Mawar – Tulip –	N : **0** Pita :

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: **Kertas** Bunga 2: **Mawar** Bunga 3: **Tulip** Bunga 4: **SELESAI** Pita: Kertas – Mawar – Tulip – Bunga: 3	Bunga 1: **SELESAI** Pita : Bunga: 0

Source code

```
package main

import "fmt"

func main() {
    var pita string
    var bunga string
    var count int

    fmt.Print("Masukkan jumlah bunga: ")
    fmt.Scanln(&count)

    for i := 1; i <= count || bunga != "SELESAI"; i++ {
        fmt.Printf("Bunga %d: ", i)
        fmt.Scanln(&bunga)

        if bunga == "SELESAI" {
            break
        }

        pita += bunga + " - "
    }

    // Hapus tanda "-" terakhir jika ada
    if len(pita) > 0 {
        pita = pita[:len(pita)-3]
    }

    fmt.Println("Pita:", pita)
    fmt.Println("Bunga:", count-1)
}

//Zahrina Antika Malahati_2311102109
```

Output

```
PS C:\Users\HP> go run "c:\PRAKALPRO2\2bno2.go"
Masukkan jumlah bunga: 4
Bunga 1: Kertas
Bunga 2: Mawar
Bunga 3: Tulip
Bunga 4: SELESAI
Pita: Kertas - Mawar - Tulip
Bunga: 3
PS C:\Users\HP> go run "c:\PRAKALPRO2\2bno2.go"
Masukkan jumlah bunga: 1
Bunga 1: SELESAI
Pita:
Bunga: 0
PS C:\Users\HP>
```

Deskripsi

Program ini adalah program dengan bahasa Go untuk menerima input dari user yang berisi sejumlah bunga dan nama – nama bunga, kemudian program ini mengabungkan nama – nama bunga menjadi satu string yang dipisahkan oleh tanda hubung. Ketika user menginputkan kata “SELESAI” maka program akan menampilkan sejumlah bunga dan daftar bunga yang telah diinputkan.

1. **Deklarasi Variabel:**
    - pita: Menyimpan string yang berisi nama-nama bunga.
    - bunga: Menyimpan nama bunga yang diinputkan sementara.
    - count: Menghitung jumlah bunga yang valid.
2. **Input Jumlah Bunga:**

    Meminta pengguna memasukkan jumlah bunga yang ingin diinputkan.
3. **Looping Input Bunga:**
    - Looping akan berjalan hingga pengguna memasukkan "SELESAI" atau mencapai jumlah bunga yang ditentukan.
    - Meminta pengguna memasukkan nama bunga.
    - Jika input adalah "SELESAI", loop akan berhenti.
    - Menggabungkan nama bunga ke dalam pita.
4. **Menghapus Tanda Hubung Terakhir:**

    Menghapus tanda hubung "-" terakhir pada pita agar tampilan lebih rapi.
5. **Menampilkan Hasil:**
    - Mencetak pita yang berisi daftar bunga.
    - Mencetak jumlah bunga yang valid (dikurangi 1 karena kita juga menghitung input "SELESAI").

❖ Unguided 3 (Latihan 2B no 3)

Soal

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 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.
```

Source code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var berat1, berat2 float64
    var totalBerat float64
    var selisihBerat float64

    for {
        fmt.Print("Masukkan berat belanjaan di kedua
kantong: ")
        fmt.Scanf("%f %f", &berat1, &berat2)

        // Menghitung selisih berat antara kedua kantong
        selisihBerat = math.Abs(berat1 - berat2)
        totalBerat = berat1 + berat2

        // Mengecek apakah selisih berat lebih dari 9 kg
        if selisihBerat >= 9 {
            fmt.Println("Sepeda motor pak Andi akan oleng:
true")
        } else {
            fmt.Println("Sepeda motor pak Andi akan oleng:
false")
        }

        // Mengecek apakah berat total melebihi 150 kg atau
salah satu kantong beratnya negatif
        if totalBerat > 150 || berat1 < 0 || berat2 < 0 {
            fmt.Println("Proses selesai.")
            break
        }
    }
}

//Zahrina Antika Malahati_2311102109
```

Output

```
PS C:\Users\HP> go run "c:\PRAKALPRO2\2bno3.go"
Masukkan berat belanjaan di kedua kantong: 5 10
Sepeda motor pak Andi akan oleng: false
Masukkan berat belanjaan di kedua kantong: Sepeda motor pak Andi akan oleng: false
Masukkan berat belanjaan di kedua kantong: 55.6 70.2
Sepeda motor pak Andi akan oleng: true
Masukkan berat belanjaan di kedua kantong: Sepeda motor pak Andi akan oleng: true
Masukkan berat belanjaan di kedua kantong: 72.3 66.9
Sepeda motor pak Andi akan oleng: false
Masukkan berat belanjaan di kedua kantong: Sepeda motor pak Andi akan oleng: false
Masukkan berat belanjaan di kedua kantong: 59.5 98.7
Sepeda motor pak Andi akan oleng: true
Proses selesai.
PS C:\Users\HP>
```

Deskripsi

Program ini adalah program dengan bahasa Go yang digunakan untuk mengecek apakah sepeda motor Pak Andi akan oleng atau tidak berdasarkan perbandingan berat barang di dua kantong terpal yang dia bawa. Program akan melakukan hal-hal berikut:

1. **Input Berat Barang**: Program meminta input dua buah bilangan real positif yang menyatakan berat barang di dua kantong terpal (misalnya: berat1 dan berat2). Input ini dimasukkan secara berulang-ulang.

2. **Menghitung Selisih Berat**: Setiap kali input diberikan, program akan menghitung selisih antara berat barang di kedua kantong. Selisih ini digunakan untuk menentukan apakah sepeda motor akan oleng atau tidak.

    - Jika selisih berat antar kedua kantong **lebih besar atau sama dengan 9 kg**, maka sepeda motor dianggap **akan oleng**, dan program menampilkan "Sepeda motor pak Andi akan oleng: true".

    - Jika selisih berat **kurang dari 9 kg**, maka sepeda motor dianggap **tidak oleng**, dan program menampilkan "Sepeda motor pak Andi akan oleng: false".

3. **Menghitung Total Berat**: Program juga menghitung total berat barang di kedua kantong (totalBerat = berat1 + berat2).

4. **Kondisi Penghentian Program**: Program akan berhenti meminta input dan menghentikan proses jika salah satu dari kondisi berikut terjadi:

    - **Total berat barang** di kedua kantong melebihi **150 kg**.
    - **Salah satu kantong memiliki berat negatif** (berat yang tidak masuk akal).

Ketika salah satu dari kondisi di atas terpenuhi, program akan menampilkan pesan "Proses selesai." dan keluar dari loop.

❖ Unguided 4 (Latihan 2B no 4)

Soal

4. 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
```

$\sqrt{2}$ merupakan bilangan irasional. Meskipun demikian, nilai tersebut dapat dihampiri 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 di belakang koma.

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

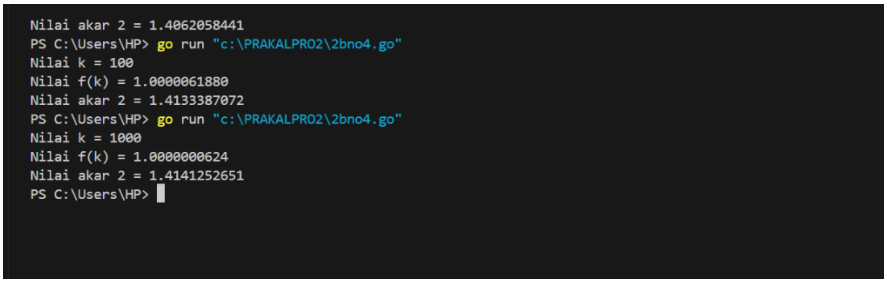
1	Nilai K = **10** Nilai akar 2 = 1.4062058441
2	Nilai K = **100** Nilai akar 2 = 1.4133387072
3	Nilai K = **1000** Nilai akar 2 = 1.4141252651

Source code

```
package main

import (
    "fmt"
    "math"
)
```

```go
// Fungsi untuk menghitung nilai f(k)
func hitungF(k int) float64 {
    pembilang := math.Pow(float64(4*k+2), 2)
    penyebut := float64((4*k + 1) * (4*k + 3))
    return pembilang / penyebut
}

func main() {
    var k int

    // Input nilai k
    fmt.Print("Nilai k = ")
    fmt.Scan(&k)

    // Hitung dan tampilkan nilai f(k)
    hasil := hitungF(k)
    fmt.Printf("Nilai f(k) = %.10f\n", hasil)

    // Hitung hampiran akar 2
    akarDua := 1.0
    for i := 0; i <= k; i++ {
        akarDua *= hitungF(i)
    }
    fmt.Printf("Nilai akar 2 = %.10f\n", akarDua)
}

//Zahrina Antika Malahati_2311102109
```

Output



Deskripsi

Program ini adalah program dengan bahasa Go yang menghitung nilai dari sebuah fungsi matematika f(k), dan menggunakan nilai tersebut untuk menghampiri nilai akar kuadrat dari 2.

1. **Input Nilai k**: Program meminta pengguna untuk memasukkan nilai kkk, sebuah bilangan bulat yang akan digunakan dalam perhitungan.
2. **Menghitung dan Menampilkan Nilai f(k)** Program memanggil fungsi hitungF(k) untuk menghitung nilai f(k) berdasarkan input pengguna.

Hasilnya kemudian ditampilkan dengan format 10 angka di belakang koma.

3. **Menghampiri Nilai Akar 2**: Program menghitung nilai hampiran akar kuadrat dari 2. Untuk ini, sebuah loop dari 000 hingga kkk dilakukan. Pada setiap iterasi, nilai hasil perkalian f(i) dengan hasil sebelumnya disimpan dalam variabel akarDua. Pada akhir loop, nilai hampiran akar 2 ditampilkan dengan format 10 angka di belakang koma.

Program ini bertujuan untuk menunjukkan bagaimana kita bisa menghampiri nilai akar kuadrat dari 2 dengan menggunakan hasil perhitungan dari fungsi f(k).

❖ Unguided 5 (Latihan 2C no 1)

Soal

Soal Latihan Modul 2C

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

   Dari berat parsel (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	Contoh #1 Berat parsel (gram): **8500** Detail berat: 8 kg + 500 gr Detail biaya: Rp. 80000 + Rp. 2500 Total biaya: Rp. 82500
2	Contoh #2 Berat parsel (gram): **9250** Detail berat: 9 kg + 250 gr Detail biaya: Rp. 90000 + Rp. 3750 Total biaya: Rp. 93750
3	Contoh #3 Berat parsel (gram): **11750** 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 beratGram int

    fmt.Print("Berat parsel (gram): ")
    fmt.Scanln(&beratGram)

    // Konversi ke kilogram dan gram
    beratKg := beratGram / 1000
    sisaGram := beratGram % 1000

    // Hitung biaya dasar
    biayaDasar := beratKg * 10000

    // Hitung biaya tambahan
    var biayaTambahan int
    if sisaGram >= 500 {
        biayaTambahan = sisaGram * 5
    } else {
        biayaTambahan = sisaGram * 15
    }

    // Pengecualian untuk sisa berat kurang dari 1kg jika
total berat lebih dari 10kg
    if beratKg > 10 && sisaGram < 1000 {
        biayaTambahan = 0
    }

    // Hitung total biaya
    totalBiaya := biayaDasar + biayaTambahan

    // Tampilkan hasil
     fmt.Printf("Detail berat: %d kg + %d gr\n", beratKg,
sisaGram)
       fmt.Printf("Detail  biaya:  Rp.  %d  +  Rp.  %d\n",
biayaDasar, biayaTambahan)
    fmt.Printf("Total biaya: Rp. %d\n", totalBiaya)
}

//Zahrina Antika Malahati_2311102109
```

Output

```
PS C:\Users\HP> go run "c:\PRAKALPRO2\2Cno1.go"
Berat parsel (gram): 8500
Detail berat: 8 kg + 500 gr
Detail biaya: Rp. 80000 + Rp. 2500
Total biaya: Rp. 82500
PS C:\Users\HP> go run "c:\PRAKALPRO2\2Cno1.go"
Berat parsel (gram): 9250
Detail berat: 9 kg + 250 gr
Detail biaya: Rp. 90000 + Rp. 3750
Total biaya: Rp. 93750
PS C:\Users\HP> go run "c:\PRAKALPRO2\2Cno1.go"
Berat parsel (gram): 11750
Detail berat: 11 kg + 750 gr
Detail biaya: Rp. 110000 + Rp. 0
Total biaya: Rp. 110000
PS C:\Users\HP>
```

Deskripsi

Program diatas adalah program dengan bahasa Go untuk menerima input dari user terkait berat parsel dalam gram, kemudian program akan melakukan perhitungan yaitu mengubah berat dari gram ke kilogram dan gram, menghitung biaya dasar berdasarkan berat dalam kilogram, menghitung biaya tambahan berdasarkan sisa berat dan ketentuan yang ada. Adapun penjelasan dari kode pada program tersebut,

- **Deklarasi variabel:** Mendeklarasikan variabel beratGram untuk menyimpan input berat dalam gram.
- **Input:** Meminta pengguna untuk memasukkan berat parsel dalam gram.
- **Konversi:** Mengubah berat dari gram ke kilogram dan gram menggunakan operasi pembagian dan modulus.
- **Hitung biaya dasar:** Menghitung biaya dasar berdasarkan berat dalam kilogram.
- **Hitung biaya tambahan:** Menghitung biaya tambahan berdasarkan sisa berat dan ketentuan yang ada.
- **Pengecualian:** Memeriksa kondisi khusus jika total berat lebih dari 10kg dan sisa berat kurang dari 1kg.
- **Hitung total biaya:** Menjumlahkan biaya dasar dan biaya tambahan.
- **Output:** Menampilkan hasil perhitungan dalam format yang sesuai dengan contoh.

❖ Unguided 6 (Latihan 2C no 2)

Soal

2. 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

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          nam = "A"
10     }
11     if nam > 72.5 {
12         nam = "AB"
13     }
14     if nam > 65 {
15         nam = "B"
16     }
17     if nam > 57.5 {
18         nam = "BC"
19     }
20     if nam > 50 {
21         nam = "C"
22     }
23     if nam > 40 {
24         nam = "D"
25     } else if nam <= 40 {
26         nam = "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?

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

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

Jawab :

a. Jika nilai nam diberikan 80.1, maka program akan mencetak "A". Hal ini sesuai dengan spesifikasi soal karena nilai 80.1 berada dalam rentang 80 hingga 100, yang sesuai dengan nilai huruf A.
b. Terdapat beberapa kesalahan logika dalam program tersebut:
   1. Kondisi if yang tumpang tindih: Kondisi-kondisi if yang digunakan saling tumpang tindih. Misalnya, jika nilai nam adalah 75, maka kondisi if nam >= 80 dan if nam >= 72.5 akan sama-sama bernilai benar. Hal ini menyebabkan program tidak dapat menentukan nilai huruf yang tepat.
   2. Kurangnya kondisi else if: Penggunaan if berulang kali tanpa menggunakan else if akan membuat program mengecek semua kondisi, meskipun kondisi sebelumnya sudah terpenuhi. Hal ini tidak efisien dan dapat menyebabkan kesalahan.

   Alur program yang seharusnya adalah:
   - Membaca nilai nam dari pengguna.
   - Memeriksa nilai nam secara berurutan mulai dari rentang nilai tertinggi hingga terendah.
   - Ketika menemukan rentang nilai yang sesuai, langsung mencetak nilai huruf dan menghentikan pengecekan.

c. Program awal memiliki kesalahan logika dalam penggunaan kondisi if. Dengan memperbaiki struktur kondisi menjadi menggunakan else if, program dapat berjalan dengan benar dan memberikan hasil yang sesuai dengan spesifikasi soal.

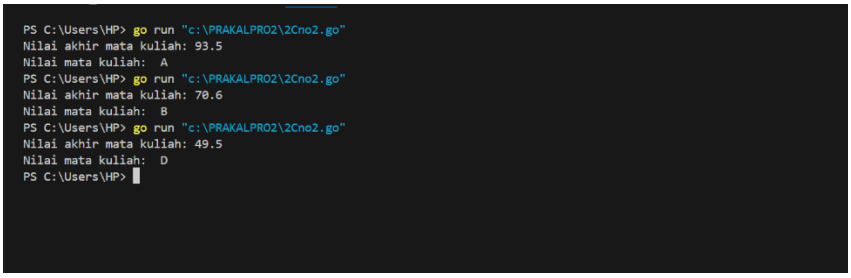
   Berikut adalah program yang telah diperbaiki :

   Source code

```
package main

import "fmt"

func main() {
    var nam float64

    fmt.Print("Nilai akhir mata kuliah: ")
    fmt.Scanln(&nam)

    var nmk string
    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)
        }


//Zahrina Antika Malahati_2311102109
```

Output



Perubahan yang dilakukan:

- Menggunakan else if untuk memeriksa kondisi secara berurutan.
- Menghentikan pengecekan setelah menemukan kondisi yang sesuai.

Jika kita menjalankan program yang telah diperbaiki dengan masukan 93.5, 70.6, dan 49.5, maka akan dihasilkan output yang sesuai, yaitu 'A', 'B', dan 'D'.

❖ Unguided 7 (Latihan 2C no 3)

Soal

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 di bawah ini (**teks bergaris bawah** adalah input/read):

Bilangan: **12** Faktor: 1 2 3 4 6 12	Bilangan: **7** 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):

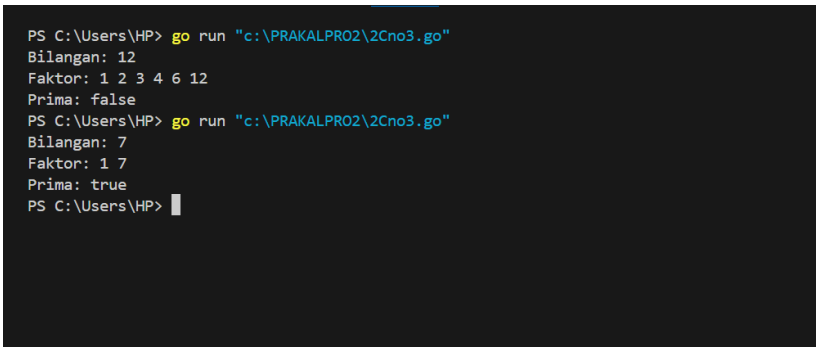
Bilangan: **12** Faktor: 1 2 3 4 6 12 Prima: false	Bilangan: **7** Faktor: 1 7 Prima: true

Source code

```
package main

import "fmt"

func main() {
    var bilangan int

    fmt.Print("Bilangan: ")
    fmt.Scan(&bilangan)

    fmt.Print("Faktor: ")
    var faktor []int
    for i := 1; i <= bilangan; i++ {
        if bilangan%i == 0 {
            faktor = append(faktor, i)
        }
    }
    for _, f := range faktor {
        fmt.Printf("%d ", f)
    }
    fmt.Println()

    if len(faktor) == 2 {
```

```
        fmt.Println("Prima: true")
    } else {
        fmt.Println("Prima: false")
    }
}


//Zahrina Antika Malahati_2311102109
```

Output



Deskripsi

Program ini adalah program dengan bahasa Go yang meminta user untuk memasukkan sebuah bilangan bulat. Kemudian, program akan mencari semua faktor dari bilangan tersebut dan mencetaknya. Setelah itu, program akan menentukan apakah bilangan tersebut adalah bilangan prima atau bukan. Bilangan prima adalah bilangan yang hanya memiliki dua faktor, yaitu 1 dan bilangan itu sendiri. Adapun penjelasan dari kode diatas yaitu,

1. **Import:** Kita mengimpor paket fmt untuk input/output.
2. **Deklarasi Variabel:** Kita mendeklarasikan variabel bilangan untuk menyimpan input pengguna.
3. **Input:** Kita meminta pengguna untuk memasukkan bilangan dan menyimpannya ke dalam variabel bilangan.
4. **Mencari Faktor:**
    - Kita membuat slice faktor untuk menyimpan faktor-faktor yang ditemukan.
    - Kita melakukan perulangan dari 1 hingga bilangan.
    - Jika bilangan habis dibagi i, maka i adalah faktor dan kita tambahkan ke slice faktor.

5. **Menampilkan Faktor:**
    - Kita melakukan perulangan untuk setiap elemen dalam slice faktor dan mencetaknya.
6. **Menentukan Bilangan Prima:**
    - Jika panjang slice faktor adalah 2 (hanya 1 dan bilangan itu sendiri), maka bilangan tersebut adalah prima.
    - Kita mencetak Prima: true jika bilangan prima, Prima: false jika bukan