

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



Oleh:

Geranada Saputra Priambudi

2311102008

IF-11-06

**PROGRAM STUDI S1 INFORMATIKA
FAKULTAS INFORMATIKA
UNIVERSITAS TELKOM PURWOKERTO
2024**

I. DASAR TEORI

Struktur Program Go

Dalam kerangka program yang ditulis dalam bahasa pemrograman Go, program utama selalu 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 baris teks yang diawali dengan garis miring ganda (//) s.d. akhir baris, atau.
- Beberapa baris teks yang dimulai dengan pasangan karakter ‘/*’ dan diakhiri dengan ‘*/’.

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 program go disimpan dalam file teks dengan ekstensi *.go, dengan nama bebas. Sebaiknya nama file adalah nama untuk program tersebut.
- Setiap satu program lengkap Go disimpan dalam satu folder tersendiri. Nama folder merupakan nama program tersebut. Karena itu secara prinsip, satu program Go dapat dipecah dalam beberapa file dengan ekstensi ".go selama disimpan dalam folder yang sama.

Kompilasi

Beberapa bahasa pemrograman dirancang untuk diimplementasikan sebagai interpreter dan lainnya sebagai kompilator. Interpreter akan membaca setiap baris instruksi dan kemudian langsung mengeksekusinya, dengan hanya sedikit pemeriksaan apakah penulisan keseluruhan program sudah benar atau belum. Kompilator akan memeriksa keseluruhan program sumber dan kemudian mengubahnya menjadi program eksekutabel, sehingga konsistensi penulisan (seperti penggunaan tipe data) sudah diperiksa sebelum eksekusi. Selain itu karena program dibuat menjadi eksekutabel lebih dahulu, proses optimasi dapat dilakukan sehingga program menjadi sangat efisien.

Tipe Data

Golang memiliki basic type data:

- string : untuk data text.
- bool: untuk tipe data boolean true dan false.
- numeric: int8, uint8(byte), int16, uint16, int32, uint32, int64, uint64, int, uint, uintptr, float32, float64, complex64, complex128

Struktur Kontrol Perulangan

Go hanya mempunyai kata kunci for untuk semua jenis perulangan yang kita pelajari dalam notasi algoritma. Dua bentuk yang kita gunakan di sini adalah struktur while-loop dan repeat-until.

Dalam konsep pemrograman terstruktur, setiap rancangan algoritma harus memenuhi syarat satu pintu masuk dan satu pintu keluar. Karena itu tidaklah diperkenankan untuk membuat program sumber yang mempunyai struktur loop yang mempunyai pintu keluar lebih dari satu, seperti:

- Satu pintu keluar dari kondisi for dan satu lagi dari instruksi if-break
 - Atau mempunyai instruksi If-break yang lebih dari satu.
- 1) Bentuk While-Loop
Bentuk while-loop memastikan setiap kali memasuki loop, ada kondisi yang harus terpenuhi (benar/true). Ini juga berarti saat keluar dari loop, maka nilai kondisi tersebut pasti salah/false!
 - 2) Bentuk Repeat-Until
Bentuk repeat-until di perulangan dilakukan terus menerus sampai kondisi keluar terpenuhi. Artinya selama kondisi belum terpenuhi (salah(false) maka perulangan akan terus dilakukan. Pada saat keluar dari loop maka nilai kondisi pasti benar/true!

Struktur Kontrol Percabangan

Untuk analisa kasus, bahasa Go mendukung dua bentuk percabangan, yaitu If-else dan switch-case.

- 1) Bentuk if-else
Ada 2 bentuk if-else : satu instruksi If-else-endif saja (hanya satu endif) dan bentuk if-else yang bersarang (dengan beberapa endif) dapat dibentuk dengan komposisi beberapa if-else-endif tersebut.
- 2) Bentuk Switch-Case

Dalam bahasa Go ada dua variasi bentuk switch-case. Bentuk yang biasa digunakan adalah ekspresi ditulis pada perintah switch dan nilai ditulis dalam setiap label case-nya. Bentuk yang kedua mempunyai switch tanpa ekspresi, tetapi setiap case boleh berisi ekspresi boolean. Tentunya bentuk yang kedua lebih bersifat umum, dan merupakan penyederhanaan bentuk (atau alias dari) susunan suatu if-elseif-...-else-endif

II. GUIDED

Latihan 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?

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:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\2a_1.go
Masukan input string: halo
Masukan input string: gery
Masukan input string: saputra
Output awal = halo gery saputra
Output akhir = gery saputra halo
```

Penjelasan:

Program di atas merupakan program dalam bahasa Go yang meminta pengguna untuk memasukkan tiga string secara berurutan. Ketiga string tersebut disimpan dalam variabel satu, dua, dan tiga. Setelah itu, program mencetak string sesuai urutan input sebagai "Output awal". Selanjutnya, program melakukan pertukaran urutan nilai dari ketiga variabel ini menggunakan sebuah variabel sementara (temp) untuk menyimpan

sementara nilai dari variabel pertama (satu). Dengan cara ini, nilai satu dipindahkan ke dua, dua ke tiga, dan tiga ke satu. Proses ini memastikan bahwa ketiga string bertukar posisi. Setelah proses pertukaran selesai, program mencetak "Output akhir" yang merupakan hasil dari perubahan urutan string.

Dalam contoh diatas, jika pengguna memasukkan "halo", "gery", dan "saputra", output awal akan menampilkan "halo gery saputra". Setelah pertukaran, output akhir akan menjadi "gery saputra halo", dimana string pertama digeser ke tempat kedua, yang kedua digeser ke tempat ketiga, dan yang ketiga kembali ke tempat pertama.

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).

Source Code:

```
package main

import "fmt"

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

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

func cekTahunKabisat(tahun int) bool {
    if tahun%400 == 0 {
        return true
    } else if tahun%4 == 0 {
        return true
    } else if tahun%100 == 0 {
        return false
    }
    return false
}
```

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\2a_2.go
Tahun: 2016
Kabisat: true

PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\2a_2.go
Tahun: 2018
Kabisat: false
```

Penjelasan:

Program di atas merupakan program dalam bahasa Go untuk menentukan apakah suatu tahun merupakan tahun kabisat atau bukan. Pengguna diminta untuk memasukkan tahun, dan kemudian program memanggil fungsi `cekTahunKabisat` yang mengimplementasikan aturan untuk memeriksa tahun kabisat. Di dalam fungsi ini, program memeriksa apakah tahun tersebut habis dibagi 400, 100, atau 4, karena aturan tahun kabisat didasarkan pada pembagian tersebut. Jika tahun habis dibagi 400, maka itu adalah tahun kabisat. Jika tahun habis dibagi 100 tetapi tidak habis dibagi 400, maka itu bukan tahun kabisat. Namun, jika tahun habis dibagi 4 tetapi tidak habis dibagi 100, maka itu adalah tahun kabisat. Program akan mengembalikan nilai boolean `true` jika kabisat, dan `false` jika tidak.

Contoh diatas menunjukkan bagaimana program bekerja. Misalnya, jika pengguna memasukkan tahun 2016, program akan mengembalikan `true`, karena 2016 habis dibagi 4 dan tidak habis dibagi 100, sehingga itu adalah tahun kabisat. Namun, jika pengguna memasukkan tahun 2018, hasilnya adalah `false`, karena 2018 tidak habis dibagi 4.

3. Buat program Bola yang menerima input jari-jari suatu bola (bilangan bulat). Tampilkan Volume dan Luas kulit bola.

Source Code:

```
package main

import (
    "fmt"
    "math"
)

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

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

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

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

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

```
}
```

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run "c:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06\2a_3.go"
Jejari: 5
Bola dengan jejari 5 memiliki volume 523.5987755833333 dan luas kulit 314.15926535
```

Penjelasan:

Program di atas merupakan program Go yang menghitung volume dan luas permukaan bola berdasarkan jejari yang dimasukkan oleh pengguna. Program tersebut menerima input berupa jejari dalam bentuk integer, lalu mengonversinya menjadi tipe float64. Setelah itu, program memanggil dua fungsi: `hitungVolumeBola` untuk menghitung volume bola dan `hitungLuasKulitBola` untuk menghitung luas permukaan bola. Keduanya menggunakan konstanta pi yang nilainya mendekati 3.1415926535 dan fungsi matematika dari package `math` untuk menghitung pangkat (exponent) dari jejari.

Sebagai contoh, jika pengguna memasukkan jejari 5, program akan menghitung volume bola menjadi 523.5987755833333 dan luas permukaannya sebesar 314.15926535.

III. UNGUIDED

Latihan 2A

4. Dibaca nilai temperatur dalam derajat Celsius. Nyatakan temperatur tersebut dalam Fahrenheit. Lanjutkan program di atas, sehingga temperatur dinyatakan juga dalam derajat Reamur dan Kelvin.

Source Code:

```
package main

import "fmt"

func main() {
    var celsius float64
    fmt.Print("Temperatur Celsius: ")
    fmt.Scanf("%f", &celsius)

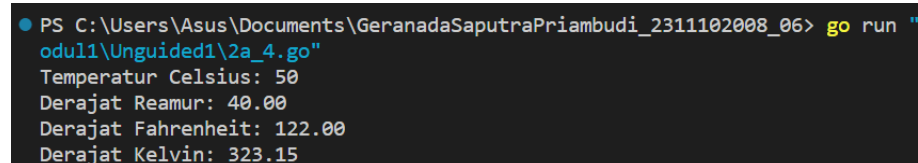
    fahrenheit := (celsius * 9 / 5) + 32

    reamur := celsius * 4 / 5

    kelvin := (fahrenheit + 459.67) * 5 / 9

    fmt.Printf("Derajat Reamur: %.2f\n", reamur)
    fmt.Printf("Derajat Fahrenheit: %.2f\n", fahrenheit)
    fmt.Printf("Derajat Kelvin: %.2f\n", kelvin)
}
```

Output:



```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run "
odul1\Unguided1\2a_4.go"
Temperatur Celsius: 50
Derajat Reamur: 40.00
Derajat Fahrenheit: 122.00
Derajat Kelvin: 323.15
```

Penjelasan:

Program di atas merupakan program Go yang digunakan untuk mengonversi suhu dari derajat Celsius ke tiga skala suhu lainnya, yaitu Reamur, Fahrenheit, dan Kelvin. Setelah meminta input suhu dalam Celsius dari pengguna, program menghitung nilai suhu dalam skala yang lain menggunakan rumus konversi yang sesuai.

Sebagai contoh, jika pengguna memasukkan suhu 50 derajat Celsius, program akan menghitung suhu dalam skala Reamur menjadi 40.00, dalam Fahrenheit menjadi 122.00, dan dalam Kelvin menjadi 323.15. Hasil konversi tersebut dicetak dengan format dua desimal menggunakan `fmt.Printf`, sehingga memberikan tampilan hasil yang lebih terstruktur dan mudah dibaca.

5. Tipe karakter sebenarnya hanya apa yang tampak dalam tampilan. Di dalamnya tersimpan dalam bentuk biner 8 bit (byte) atau 32 bit (rune) saja. Buat program ASCII yang akan membaca 5 buah data Integer dan mencetaknya dalam format karakter. Kemudian membaca 3 buah data karakter dan mencetak 3 buah karakter setelah karakter tersebut (menurut tabel ASCII)

Masukan terdiri dari dua baris. Baris pertama berisi 5 buah data integer. Data integer mempunyai nilai antara 32 s.d. 127. Baris kedua berisi 3 buah karakter yang berdampingan satu dengan yang lain (tanpa dipisahkan spasi). Keluaran juga terdiri dari dua baris. Baris pertama berisi 5 buah representasi karakter dari data yang diberikan, yang berdampingan satu dengan lain, tanpa dipisahkan spasi. Baris kedua berisi 3 buah karakter (juga tidak dipisahkan oleh spasi).

Source Code:

```
package main

import (
    "fmt"
)

func main() {
    var (
        integers    [5]int
        characters   [3]rune
    )

    fmt.Print("Masukkan 5 data integer (antara 32 s.d. 127): ")
    for i := 0; i < 5; i++ {
        fmt.Scan(&integers[i])
    }

    fmt.Print("Output karakter: ")
    for _, value := range integers {
        fmt.Printf("%c", value)
    }
    fmt.Println()

    fmt.Print("Masukkan 3 huruf: ")
    var input string
    fmt.Scan(&input)

    for i, char := range input {
        if i < 3 {
            characters[i] = char
        }
    }

    fmt.Print("Output huruf : ")
    for _, char := range characters {
        nextChar := char + 1
    }
}
```

```
        fmt.Printf("%c", nextChar)
    }
    fmt.Println()
}
```

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\Unguided2\2a_5.go
Masukkan 5 data integer (antara 32 s.d. 127): 66 97 103 117 115
Output karakter: Bagus
Masukkan 3 huruf: SNO
Output huruf : TOP
```

Penjelasan:

Program diatas untuk membaca dan mengonversi data integer menjadi karakter berdasarkan nilai ASCII, serta membaca karakter dari input dan mencetak karakter yang berikutnya. Pada bagian awal, program meminta pengguna untuk memasukkan 5 data integer dalam rentang 32 hingga 127. Setelah itu, program mencetak representasi karakter dari setiap integer yang dimasukkan. Dengan menggunakan fungsi `fmt.Scan`, program dengan mudah mengambil input dari pengguna dan mengonversi integer menjadi karakter dengan menggunakan format `%c`. Selanjutnya, program meminta pengguna untuk memasukkan 3 karakter dalam bentuk string, lalu program mengambil 3 karakter pertama dari string tersebut. Setelah mendapatkan karakter yang dimasukkan, program mencetak karakter yang merupakan hasil penambahan satu pada nilai ASCII dari masing-masing karakter tersebut. Hasil akhirnya adalah output yang menunjukkan karakter yang dihasilkan dari input integer dan karakter berikutnya dari input karakter.

Contoh penggunaan program menunjukkan bahwa ketika input karakter "SNO", program mencetak "TOP" sebagai karakter berikutnya, yang menunjukkan bahwa logika penghitungan ASCII telah diterapkan dengan benar.

Latihan 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.

Source Code:

```
package main

import (
    "fmt"
)

func main() {
    var warna1, warna2, warna3, warna4 string
    berhasil := true
    urutanBenar := [4]string{"merah", "kuning", "hijau", "ungu"}

    for percobaan := 1; percobaan <= 5; percobaan++ {
        fmt.Printf("Percobaan %d: ", percobaan)
        fmt.Scan(&warna1, &warna2, &warna3, &warna4)

        if warna1 != urutanBenar[0] || warna2 !=
urutanBenar[1] ||
            warna3 != urutanBenar[2] || warna4 !=
urutanBenar[3] {
            berhasil = false
        }
    }

    if berhasil {
        fmt.Println("BERHASIL: TRUE")
    } else {
        fmt.Println("BERHASIL: FALSE")
    }
}
```

Output:

```
● PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\Unguided3\2b_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
BERHASIL: TRUE
● PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\Unguided3\2b_1.go
Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: merah kuning merah ungu
Percobaan 5: merah kuning hijau ungu
BERHASIL: FALSE
```

Penjelasan:

Program Go diatas meminta pengguna untuk memasukkan empat warna dalam lima percobaan dan memeriksa apakah urutan warna yang dimasukkan sesuai dengan urutan yang benar, yaitu "merah", "kuning", "hijau", dan "ungu". Dalam setiap iterasi, pengguna diminta untuk memasukkan warna, dan program membandingkan input dengan urutan yang benar. Jika ada kesalahan dalam satu atau lebih percobaan, variabel boolean berhasil akan diatur menjadi false. Setelah semua percobaan selesai, program akan mencetak hasilnya. Jika semua input dalam urutan yang benar, maka program akan menampilkan "BERHASIL: TRUE"; sebaliknya, jika ada kesalahan pada salah satu percobaan, program akan mencetak "BERHASIL: FALSE".

2. Suatu pita (string) berisi kumpulan nama-nama bunga yang dipisahkan oleh spasi dan '-', contoh pita diilustrasikan seperti berikut ini.

Pita: mawar - melati - tulip-teratal - 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.

Source Code:

```
package main

import (
    "fmt"
)

func main() {
    var N int
    var pita string
    fmt.Print("N: ")
    fmt.Scanln(&N)
    for i := 1; i <= N; i++ {
        var bunga string
        fmt.Printf("Bunga %d: ", i)
        fmt.Scanln(&bunga)
        pita += bunga + " - "
    }
    fmt.Println("Pita:", pita)
}
```

Output:

```

PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\Unguided4\2b_2.go"
N: 3
Bunga 1: Kertas
Bunga 2: Mawar
Bunga 3: Tulip
Pita: Kertas - Mawar - Tulip -

PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\Unguided4\2b_2.go"
N: 0
Pita:

```

Penjelasan:

Program Go diatas untuk menerima input berupa nama-nama bunga dari pengguna dan menggabungkannya ke dalam sebuah string yang disebut "pita". Pertama, program meminta pengguna untuk memasukkan sebuah bilangan bulat positif, N, yang menunjukkan jumlah nama bunga yang akan dimasukkan. Setelah itu, program menjalankan loop sebanyak N kali, di mana setiap iterasi meminta pengguna untuk memasukkan nama bunga. Setiap nama bunga yang dimasukkan kemudian digabungkan ke dalam string pita menggunakan operator +=, dengan tambahan " - " sebagai pemisah antara nama-nama bunga. Jika pengguna memasukkan N dengan nilai 0, program tidak akan melakukan proses input, dan output dari pita akan kosong. Ini karena loop tidak akan dijalankan jika N adalah 0. Contoh output yang diberikan menunjukkan bahwa jika N=3, dan bunga yang dimasukkan adalah "Kertas", "Mawar", dan "Tulip", maka hasil akhir dari string pita adalah "Kertas - Mawar - Tulip - ". Namun, jika N adalah 0, program akan mencetak "Pita: " tanpa nama bunga di dalamnya, bahwa tidak ada input yang dilakukan.

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.

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.

Source Code:

```
package main
```

```

import (
    "fmt"
    "math"
)

func main() {
    var kantong1, kantong2, totalBerat float64
    const batasBerat = 150.0

    for {
        fmt.Print("Masukkan berat belanjaan di kedua kantong (kg): ")
        fmt.Scan(&kantong1, &kantong2)

        if kantong1 < 0 || kantong2 < 0 {
            fmt.Println("Proses selesai karena ada berat negatif.")
            break
        }

        totalBerat += kantong1 + kantong2

        if totalBerat > batasBerat {
            fmt.Printf("Total berat sudah melebihi batas %.2f kg.\nProses selesai.\n", batasBerat)
            break
        }

        if math.Abs(kantong1-kantong2) >= 9 {
            fmt.Println("Sepeda motor Pak Andi akan oleng: true")
        } else {
            fmt.Println("Sepeda motor Pak Andi akan oleng: false")
        }

        if kantong1 >= 9 || kantong2 >= 9 {
            fmt.Println("Proses selesai.")
            break
        }
    }
}

```

Output:

```

● PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_23111020
odul1\Unguided5\2b_3.go"
Masukkan berat belanjaan di kedua kantong (kg): 5 10
Sepeda motor Pak Andi akan oleng: false
Proses selesai.

```

```

PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul11\Unguided5\2b_3.go
Masukkan berat belanja di kedua kantong (kg): 55.6 70.2
Sepeda motor Pak Andi akan oleng: true
Proses selesai.

```

Penjelasan:

Program diatas untuk membantu Pak Andi dalam mengelola berat belanjaan yang dibawanya menggunakan sepeda motor. Program tersebut meminta pengguna untuk memasukkan berat dua kantong belanjaan dalam satuan kilogram. Setelah menerima input, program memeriksa apakah ada berat negatif, yang akan menghentikan proses jika terjadi. Selanjutnya, total berat kedua kantong ditambahkan, dan jika total tersebut melebihi 150 kg, program akan menampilkan pesan peringatan dan berhenti. Selain itu, program juga memeriksa selisih berat antara kedua kantong; jika selisihnya mencapai 9 kg atau lebih, program akan menunjukkan bahwa sepeda motor akan oleng. Proses akan berhenti jika salah satu kantong mencapai berat 9 kg atau lebih.

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.

$\sqrt{2}$ 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 di belakang koma.

Source Code:

```

package main

import (
    "fmt"
)

func hitungFungsiFK(K int) float64 {
    return float64((4*K+2)*(4*K+2)) /
float64((4*K+1)*(4*K+3))
}

```



```

}
func hitungAkar2(K int) float64 {
    akar2 := 1.0
    for k := 0; k <= K; k++ {
        akar2 *= float64((4*k+2)*(4*k+2)) /
float64((4*k+1)*(4*k+3))
    }
    return akar2
}

func main() {
    var pilihan int
    var K int

    fmt.Println("Pilih opsi:")
    fmt.Println("1. Hitung f(K)")
    fmt.Println("2. Hitung akar 2")
    fmt.Print("Masukkan pilihan (1/2): ")
    fmt.Scan(&pilihan)

    fmt.Print("Nilai K: ")
    fmt.Scan(&K)

    if pilihan == 1 {
        fK := hitungFungsiFK(K)
        fmt.Printf("Nilai f(K) = %.10f\n", fK)
    } else if pilihan == 2 {
        akar2 := hitungAkar2(K)
        fmt.Printf("Nilai akar 2 = %.10f\n", akar2)
    } else {
        fmt.Println("Pilihan tidak valid!")
    }
}

```

Output:

```

● PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\Unguided6\2b_4.go
Pilih opsi:
1. Hitung f(K)
2. Hitung akar 2
Masukkan pilihan (1/2): 1
Nilai K: 100
Nilai f(K) = 1.0000061880
● PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06> go run
odul1\Unguided6\2b_4.go
Pilih opsi:
1. Hitung f(K)
2. Hitung akar 2
Masukkan pilihan (1/2): 2
Nilai K: 100
Nilai akar 2 = 1.4133387072
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008_06>

```

Penjelasan:

Program diatas berfungsi untuk menghitung dua nilai berdasarkan input integer K dari pengguna: $f(K)$ dan hampiran akar 2. Pengguna diminta untuk memilih antara dua opsi; jika memilih opsi pertama, program akan menghitung nilai $f(K)$ menggunakan rumus yang didefinisikan dalam fungsi `hitungFungsiFK`, yang menghasilkan nilai mendekati 1. Program menghitungnya dengan rumus dan menampilkan hasil dengan ketelitian sepuluh angka di belakang koma. Jika pengguna memilih opsi kedua, program akan menghitung hampiran akar 2 dengan memanfaatkan rumus yang dijelaskan dalam fungsi `hitungAkar2`, yang menggunakan loop untuk menghitung hasil berdasarkan K. Hasilnya juga ditampilkan dengan format yang sama. Dengan demikian, program ini mendapatkan nilai $f(K)$ dan hampiran akar 2 berdasarkan nilai K yang diberikan oleh pengguna.