

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL II
REVIEW STRUKTUR KONTROL**



Disusun Oleh :

Maulisa Elvita Sari / 2311102259

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

Struktur control dalam pemrograman, termasuk dalam Bahasa Go (Golang), adalah mekanisme yang digunakan untuk mengatur alur eksekusi program. Terdapat dua jenis utama struktur control, yaitu:

Struktur Kontrol Percabangan

Percabangan adalah bagian dari struktur kontrol yang memungkinkan pengambilan Keputusan dalam kode. Dalam Golang, percabangan umumnya dilakukan menggunakan pernyataan 'if', 'else if', 'else', dan 'switch'.

- Pernyataan if : digunakan untuk mengevaluasi kondisi Boolean. Jika kondisi tersebut true, maka blok kode di dalam pernyataan if akan dieksekusi.

Contoh:

```
if x > 10 {  
    fmt.Println("X lebih besar dari 10")  
} else {  
    fmt.Println("X tidak lebih besar dari 10")  
}
```

- Pernyataan Switch : memungkinkan pengecekan beberapa kondisi sekaligus. Jika satu kondisi terpenuhi, blok kode yang sesuai akan dieksekusi tanpa melanjutkan ke kondisi berikutnya, kecuali jika menggunakan 'fallthrough'.

Contoh:

```
switch x {  
case 1:  
    fmt.Println("Satu")  
case 2:  
    fmt.Println("Dua")  
default:  
    fmt.Println("Bukan satu atau dua")  
}
```

Struktur Kontrol Perulangan

Perulangan digunakan untuk mengeksekusi blok kode berulang kali selama kondisi tertentu terpenuhi. Dalam golang, perulangan umumnya dilakukan menggunakan pernyataan 'for'.

- Pernyataan for : merupakan satu-satunya bentuk perulangan di golang dan dapat digunakan dalam berbagai cara, termasuk sebagai perulangan tanpa batas atau dengan kondisi tertentu.

Contoh:

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

II. GUIDED

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 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.

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")
}
}

```

Screenshoot Output

```

Percobaan 3: merah kuning hijau ungu
Percobaan 4: merah kuning hijau ungu
Percobaan 5: merah kuning hijau ungu
BERHASIL : true
PS C:\Users\ASUS>
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Dokume
Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: ungu kuning hijau merah
BERHASIL : false
PS C:\Users\ASUS>

```

Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Algoritma:

- Inisialisasi daftar warna yang benar
- Membuat loop untuk 5 percobaan:
 - Minta input dari pengguna
 - Hapus spasi di awal/akhir dan pisahkan warna berdasarkan spasi.
 - Periksa setiap warna dalam urutan.
 - Jika salah, keluar dari loop

- Tampilkan hasil berdasarkan pengecekan.

Program tersebut merupakan program dalam bahasa GO dimana program di atas meminta pengguna untuk memasukkan urutan warna dalam 5 percobaan. Tujuannya adalah untuk memeriksa apakah urutan warna yang dimasukkan sesuai dengan urutan yang benar, yaitu "merah", "kuning", "hijau", dan "ungu". Jika program berhasil maka output yang muncul 'True' dan sebaliknya jika percobaan dalam memasukkan warna salah maka output yang muncul 'false'.

Sourcecode

```
package main

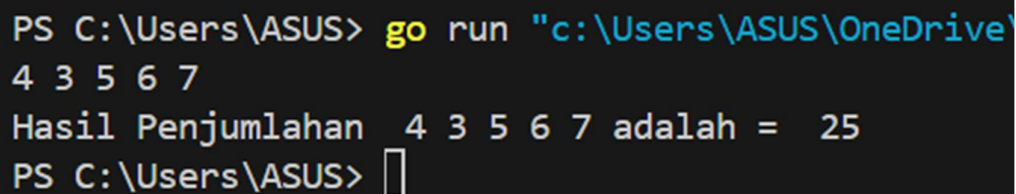
import "fmt"

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)
}
```

Screenshoot Output



```
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\
4 3 5 6 7
Hasil Penjumlahan 4 3 5 6 7 adalah = 25
PS C:\Users\ASUS> █
```

Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Algoritma:

- Inisialisasi daftar warna yang benar
 - Buat variabel 'a','b','c','d','e' untuk menyimpan lima angka bilangan bulat yang dimasukkan oleh pengguna.
- Membuat input
 - Gunakan 'fmt.Scanln' untuk membaca input dari pengguna. Input ini akan dibaca sebagai lima angka bilangan bulat.
- Melakukan Penjumlahan:
 - Hitung jumlah dari kelima angka tersebut dan simpan hasilnya dalam variabel 'hasil'.
- Menampilkan output:
 - Tampilkan hasil penjumlahan angka yang telah dimasukkan.

Program tersebut merupakan program dalam bahasa GO dimana program di atas meminta pengguna untuk menjumlahkan lima bilangan bulat yang dimasukkan oleh pengguna. Setelah melakukan penjumlahan, program akan menampilkan hasilnya.

Soal Strudi Case

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

NAM	NMK
$NAM > 80$	A
$72.5 < NAM \leq 80$	AB
$65 < NAM \leq 72.5$	B
$57.5 < NAM \leq 65$	BC
$50 < NAM \leq 57.5$	C
$40 < NAM \leq 50$	D
$NAM \leq 40$	E

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

Jawablah pertanyaan-pertanyaan berikut:

- Jika nam diberikan adalah 80.1, apa keluaran dari program tersebut? Apakah eksekusi program tersebut sesuai spesifikasi soal?
- Apa saja kesalahan dari program tersebut? Mengapa demikian? Jelaskan alur program seharusnya!
- Perbaiki program tersebut! Ujilah dengan masukan: 93.5; 70.6; dan 49.5. Seharusnya keluaran yang diperoleh adalah 'A', 'B', dan 'D'.

Sourcecode

```
package main

import "fmt"

func main() {
    var nam float32
    var nmk string

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

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

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

Screenshoot Output

```
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\
Masukkan nilai: 85
Nilai Indeks untuk nilai 85.00 adalah A
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\
Masukkan nilai: 66
Nilai Indeks untuk nilai 66.00 adalah C
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\
Masukkan nilai: 75
Nilai Indeks untuk nilai 75.00 adalah B
PS C:\Users\ASUS> █
```

Deskripsi Program

- a. Jika nam diberikan nilai 80.1 peogram tidak akan menghasilkan keluaran apa pun karena tidak ada instruksi untuk memproses nilai dan memberikan hasil (misalnya, menentukan grade).
- b. - memasukkan nilai
- Setelah menerima input nilai, program memeriksa nilai tersebut dan menentukan grade berdasarkan rentang nilai:
 - A: 80 – 100
 - B: 70 - 79.9
 - C: 60 – 69.9
 - D: 50 – 59.9
 - E: < 50- Tampilkan grade yang sesuai berdasarkan nilai yang dimasukkan
- c. Soal c:

```
package main

import "fmt"

func main() {
    var nam float32

    fmt.Print("Masukkan nilai: ")
    fmt.Scanln(&nam)
```

```
var grade string
if nam >= 80 {
    grade = "A"
} else if nam >= 70 {
    grade = "B"
} else if nam >= 60 {
    grade = "C"
} else if nam >= 50 {
    grade = "D"
} else {
    grade = "D"
}

fmt.Println("Grade:", grade)
}
```

```
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Doku
Masukkan nilai: 93.5
Grade: A
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Doku
Masukkan nilai: 70.6
Grade: B
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Doku
Masukkan nilai: 49.5
Grade: D
PS C:\Users\ASUS> █
```

III. UNGUIDED

1. 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-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.

Sourcecode

```
package main

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

func main() {
    var flowers []string
    reader := bufio.NewReader(os.Stdin)

    for {
        count := len(flowers) + 1
        fmt.Printf("Bunga %d: ", count)
        input, _ := reader.ReadString('\n')
        input = strings.TrimSpace(input)

        if input == "SELESAI" {
            break
        }

        flowers = append(flowers, input)
    }

    if len(flowers) > 0 {
        pita := strings.Join(flowers, " ")
        fmt.Printf("Pita: %s\n", pita)
        fmt.Printf("Bunga: %d\n", len(flowers))
    } else {
        fmt.Println("Pita: ")
        fmt.Println("Bunga: 0")
    }
}
```

Screenshoot Output

```
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Dokumen
Bunga 1: mawar
Bunga 2: melati
Bunga 3: tulip
Bunga 4: teratai
Bunga 5: kamboja
Bunga 6: anggrek
Bunga 7: SELESAI
Pita: mawar melati tulip teratai kamboja anggrek
Bunga: 6
PS C:\Users\ASUS> █
```

Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program di atas merupakan program menggunakan bahasa Go yang dimana digunakan untuk mengumpulkan nama-nama bunga dari pengguna. Pengguna diminta untuk memasukkan nama bunga satu per satu, dan proses akan berhenti ketika pengguna mengetik "SELESAI". Maka, program akan menampilkan daftar bunga yang dimasukkan beserta jumlahnya.

Algoritma:

- Inisialisasi
 - Buat slice kosong untuk menyimpan nama bunga.
 - Buat 'bufio.Reader' untuk membaca input
- Loop Input
 - Hitung urutan buga yang akan diminta (dimulai dari 1)
 - Tampilkan prompt untuk meminta nama bunga
 - Baca input dari pengguna dan hilangkan spasi di awal/akhir
 - Jika input adalah "SELESAI", keluar dari loop
 - Jika tidak, tambahkan nama bunga ke dalam slice.
- Tampilkan Hasil
 - Jika ada nama bunga yang dimasukkan, gabungkan menjadi string dan tampilkan
 - Tampilkan jumlah bunga

- Jika tidak ada bunga, tampilkan sesuai pesan dari pengguna

2. 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.

Sourcecode

```
package main

import (
    "fmt"
)

func main() {
    var kantong1, kantong2 float64

    for {
        fmt.Print("Masukan berat belanjaan di kedua kantong: ")
        _, err := fmt.Scanln(&kantong1, &kantong2)

        if err != nil {
            fmt.Println("Input tidak valid. Silakan masukkan dua angka.")
            continue
        }

        if kantong1 >= 9 || kantong2 >= 9 {
            break
        }
    }

    fmt.Println("Proses selesai.")
}
```

Screenshot Output

```
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Docume
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.
PS C:\Users\ASUS> █
```

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.

Sourcecode

```
package main

import (
    "fmt"
)

func main() {
    var kantong1, kantong2 float64

    for {
        fmt.Print("Masukan berat belanjaan di kedua kantong: ")
        _, err := fmt.Scanln(&kantong1, &kantong2)

        if err != nil {
            fmt.Println("Input tidak valid. Silakan masukkan dua angka.")
            continue
        }

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

        totalBerat := kantong1 + kantong2

        if totalBerat > 150 {
            fmt.Println("Proses selesai.")
            break
        }
    }
}
```



```

        selisih := kantong1 - kantong2
        if selisih < 0 {
            selisih = -selisih
        }

        if selisih >= 9 {
            fmt.Println("Sepeda motor pak Andi akan oleng: true")
        } else {
            fmt.Println("Sepeda motor pak Andi akan oleng: false")
        }
    }
}

```

Screenshoot Output

```

PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Dokumen Semester 3\P
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.
PS C:\Users\ASUS>

```

Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program di atas merupakan program menggunakan bahasa Go yang dimana digunakan untuk meminta pengguna memasukkan berat dari dua kantong belanjaan. Program ini akan terus meminta input hingga salah satu dari kantong memiliki berat 9 kg atau lebih. Program ini mengedepankan penggunaan pengulangan dan penanganan kesalahan dalam input, serta memberikan hasil yang jelas setelah memenuhi syarat tertentu.

3. 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.

Sourcecode

```
package main

import (
    "fmt"
)

func main() {
    var k float64

    fmt.Print("Nilai K: ")
    _, err := fmt.Scanln(&k)
    if err != nil {
        fmt.Println
        return
    }

    numerator := (4*k + 2) * (4*k + 2)
    denominator := (4*k + 1) * (4*k + 3)
    fK := numerator / denominator

    fmt.Printf("Nilai f(K): %.10f\n", fK)
}
```

Screenshoot Output



```
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents
Nilai K: 100
Nilai f(K): 1.0000061880
PS C:\Users\ASUS> █
```

$\sqrt{2}$ 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.

Sourcecode

```
package main

import (
    "fmt"
    "strings"
)

func approximateSqrt2(k int) float64 {
    result := 1.0
    for i := 0; i < k; i++ {
        result = (result + 2/result) / 2
    }
    return result
}

func main() {
    var k int

    fmt.Print("Nilai K = ")
    _, err := fmt.Scanln(&k)
    if err != nil {
        fmt.Println("Input tidak valid.")
        return
    }

    sqrt2 := approximateSqrt2(k)

    resultStr := fmt.Sprintf("%.10f", sqrt2)
    resultStr = replaceDotWithComma(resultStr)

    fmt.Printf("Nilai akar 2 %s\n", resultStr)
}

func replaceDotWithComma(s string) string {
    return strings.Replace(s, ".", ",", 1)
}
```

Screenshoot Output

```
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\I
Nilai K = 100
Nilai akar 2 1,4142135624
PS C:\Users\ASUS> █
```

Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program di atas merupakan program menggunakan bahasa Go yang dimana digunakan untuk menghitung nilai perkiraan dari akar kuadrat 2 menggunakan metode iterasi. Pengguna diminta untuk memasukkan nilai k, yang menentukan seberapa banyak iterasi yang akan dilakukan untuk memperbaiki etimasi hasil. Program ini juga mengganti pemisah desimal dari titik (.) menjadi koma (,) sesuai dengan format yang diinginkan. Jika pengguna memasukkan nilai k, program membaca input tersebut. Jika input tidak valid maka program akan mencetak pesan kesalahan dan keluar, tetapi jika input valid, maka program menghitung nilai perkiraan $\sqrt{2}$ dengan jumlah iterasi sesuai dengan nilai k yang diberikan. Hasil perhitungan kemudian diformat menjadi string dengan 10 angka desimal, setelah itu program mengganti titik dengan koma pada string hasil agar sesuai dengan format yang diinginkan.

4. 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.

Sourcecode

```
package main

import (
    "fmt"
)

func main() {
    var beratParsels int

    fmt.Print("Berat parcel (gram): ")
    _, err := fmt.Scanln(&beratParsels)
    if err != nil || beratParsels <= 0 {
        fmt.Println("Input tidak valid. Silakan masukkan angka positif.")
        return
    }

    totalKg := beratParsels / 1000
    sisaGram := beratParsels % 1000

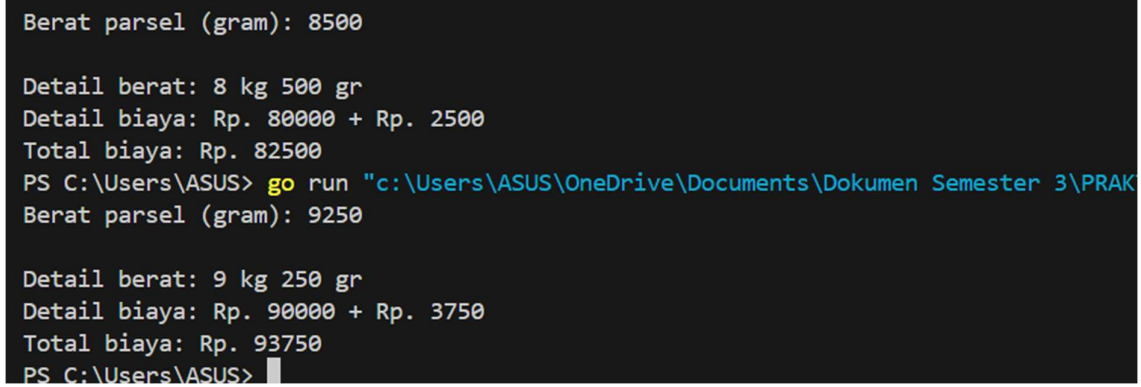
    biayaDasar := totalKg * 10000
    var biayaSisa int

    if totalKg > 10 {
        biayaSisa = 0
    } else if sisaGram >= 500 {
        biayaSisa = sisaGram * 5
    } else {
        biayaSisa = sisaGram * 15
    }

    fmt.Printf("\nDetail berat: %d kg %d gr\n", totalKg, sisaGram)
    fmt.Printf("Detail biaya: Rp. %d", biayaDasar)
    if biayaSisa > 0 {
        fmt.Printf(" + Rp. %d", biayaSisa)
    }
}
```

```
}  
    fmt.Printf("\nTotal biaya: Rp. %d\n", biayaDasar+biayaSisa)  
}
```

Screenshoot Output



```
Berat parcel (gram): 8500  
  
Detail berat: 8 kg 500 gr  
Detail biaya: Rp. 80000 + Rp. 2500  
Total biaya: Rp. 82500  
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Dokumen Semester 3\PRAK  
Berat parcel (gram): 9250  
  
Detail berat: 9 kg 250 gr  
Detail biaya: Rp. 90000 + Rp. 3750  
Total biaya: Rp. 93750  
PS C:\Users\ASUS> |
```

Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program di atas merupakan program menggunakan bahasa Go yang dimana digunakan untuk menghitung biaya pengiriman parcel yang dibawa oleh PT POS. Pengguna diminta untuk memasukkan berat parcel dalam satuan gram, dan program akan menghitung total berat dalam kilogram serta sisa berat dalam gram. Biaya pengiriman dihitung berdasarkan ketentuan tertentu, termasuk biaya per kg dan tambahan biaya untuk sisa berat. Setelah itu, output akan mencetak detail dari biaya dasar dan biaya tambahan jika ada, serta total biaya pengiriman.

Algoritma:

- Input: Meminta pengguna untuk memasukkan berat parcel dalam gram
- Perhitungan berat:
 - Hitung berat total dalam kg dengan membagi berat parcel dengan 1000.
 - Hitung sisa berat dalam gram dengan menggunakan operasi modulus.
- Perhitungan biaya:
 - Hitung biaya dasar sebagai 10.000 per kg.
 - Tentukan biaya tambahan berdasarkan sisa berat
- Output: menampilkan detail dari berat, biaya, dan total biaya pengiriman

5. 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!

Sourcecode

```
package main

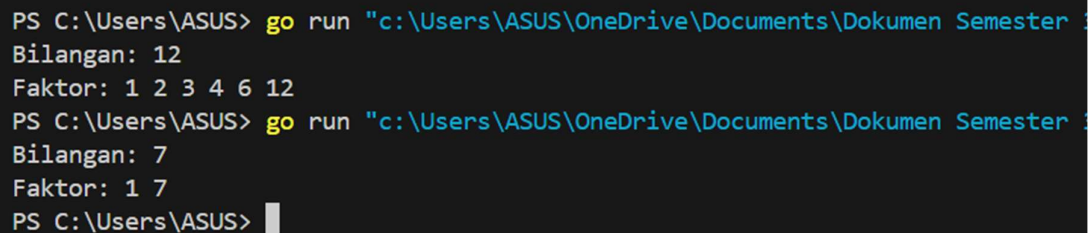
import (
    "fmt"
)

func main() {
    var b int

    fmt.Print("Bilangan: ")
    _, err := fmt.Scanln(&b)
    if err != nil || b <= 1 {
        fmt.Println("Input tidak valid. Silakan masukkan bilangan bulat lebih dari 1.")
        return
    }

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

Screenshoot Output



```
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Dokumen Semester 3\
Bilangan: 12
Faktor: 1 2 3 4 6 12
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Dokumen Semester 3\
Bilangan: 7
Faktor: 1 7
PS C:\Users\ASUS> 
```

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.

Sourcecode

```
package main

import (
    "fmt"
)

func main() {
    var b int

    fmt.Print("Bilangan: ")
    _, err := fmt.Scanln(&b)
    if err != nil || b <= 0 {
        fmt.Println("Input tidak valid. Silakan masukkan bilangan bulat lebih dari 0.")
        return
    }

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

    if factorCount == 2 {
        fmt.Println("Prima: true")
    } else {
        fmt.Println("Prima: false")
    }
}
```


Screenshoot Output

```
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Dokume
Bilangan: 12
Faktor: 1 2 3 4 6 12
Prima: false
PS C:\Users\ASUS> go run "c:\Users\ASUS\OneDrive\Documents\Dokume
Bilangan: 7
Faktor: 1 7
Prima: true
PS C:\Users\ASUS> █
```

Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program di atas merupakan program menggunakan bahasa Go yang dimana digunakan untuk mencari dan menampilkan semua faktor dari bilangan positif yang dimasukkan oleh pengguna, serta menentukan apakah bilangan tersebut merupakan bilangan prima. Dengan mencari faktor dari sebuah variabel, setelah itu program mencetak jumlah faktor yang ditemukan. Jika jumlah faktor tepat 2 (yaitu 1 dan bilangan itu sendiri), maka program menyimpulkan bahwa variabel tersebut adalah bilangan prima dan mencetak "prima: true", tetapi jika jumlah faktor lebih dari 2, maka program akan mencetak "Prima: false".

Algoritma:

- Meminta input bilangan bulat i.
- Jika input tidak valid atau $i \leq 0$, tampilkan pesan kesalahan.
- Inisialisasi 'factorCount' dengan 0.
- Lakukan loop dari 1 hingga i:
 - Jika $i \bmod x = 0$, cetak x dan tambahkan 'factorCount'
- Jika 'factorCount' sama dengan 2, cetak "prima: true", jika tidak maka cetak Prima: false".

DAFTAR PUSTAKA

[UNIKOM_MuhammadAlif_BAB 2.pdf](#)

[AhmadFaisal_A1C615001_Dasar-DasarBahasaPemrogramanGolang.pdf \(pilkommedia.org\)](#)

[Definisi Struktur Kontrol Percabangan dalam Pemrograman | kumparan.com](#)