

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL II  
REVIEW STRUKTUR KONTROL**



**Disusun Oleh :**

**Wafiq Nur Azizah / 2311102270**

**S1IF-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 kontrol dari Go berkaitan dengan bahasa C namun berbeda dalam hal-hal tertentu. Tidak ada pengulangan *do* atau *while*, hanya *for*, *switch* yang lebih fleksibel *if* dan *switch* bisa menggunakan perintah inisialisasi seperti halnya pada *for*, perintah *break* dan *continue* memiliki label identifikasi yang opsional, dan ada beberapa kontrol struktur baru termasuk *switch* pada tipe dan komunikasi *multiplexer*, *select*. Sintaksnya juga sedikit berbeda, tidak ada tanda kurung dan bagian badan dari kontrol harus selalu dibatasi oleh kurung kurawal.

### A. IF ( Percabangan )

Dalam Go *if* yang sederhana itu bentuknya seperti ini:

```
if x > 0 {  
    return y  
}
```

Wajibnya kurung kurawal mendorong penulisan perintah *if* menjadi beberapa baris. Gaya penulisan seperti ini sangat bagus, khususnya bila bagian badan kondisi memiliki perintah kontrol seperti *return* atau *break*. Secara *if* dan *switch* dapat melakukan perintah inisialisasi, maka sangat umum melihatnya digunakan untuk mendeklarasikan lokal variabel. Sebagai contoh, tidak perlu menghabiskan waktu mensejajarkan bagian komentar pada *field* dari sebuah *struct*.

```
if err := file.Chmod(0664); err != nil {  
    log.Print(err)  
    return err  
}
```

### B. For ( Perulangan )

Pengulangan *for* pada Go mirip tapi tidak sama dengan C. Ia menggabungkan *for* dan *while* dan tidak ada *for-while*. Ada tiga bentuk pengulangan *for*, hanya satu yang menggunakan titik koma.

```
// Seperti "for" pada C  
for inisialisasi; kondisi; selanjutnya { }  
  
// Seperti "while" pada C  
for kondisi { }  
  
// Seperti "for(;;)" pada C  
for { }
```

Deklarasi singkat membuatnya mudah mendeklarasikan variabel *index* di dalam pengulangan.

```
sum := 0
for i := 0; i < 10; i++ {
    sum += i
}
```

Jika melakukan pengulangan pada *array*, *slice*, *string*, atau *map*, atau membaca dari sebuah channel, sebuah klausa *range* dapat digunakan pada pengulangan.

```
for key, value := range oldMap {
    newMap[key] = value
}
```

Jika hanya membutuhkan item pertama dalam *range* (*key* dari *map* atau indeks dari *array/slice/string*), hapus variabel kembalian kedua,

```
for key := range m {
    if key.expired() {
        delete(m, key)
    }
}
```

Jika hanya membutuhkan item kedua (nilainya), gunakan pengidentifikasi kosong, sebuah garis bawah, untuk mengindahkan yang pertama:

```
sum := 0
for _, value := range array {
    sum += value
}
```

### C. Switch ( Menu )

Kontrol switch pada Go lebih generik daripada C. Ekspresi *switch* pada Go tidak harus konstan maupun *integer*, bagian kondisi *case* dievaluasi dari atas ke bawah sampai ditemukan kondisi yang sesuai, dan jika ekspresi *switch* tidak memiliki ekspresi, ia akan memeriksa kondisi *case* yang bernilai *true*. Oleh karena itu, memungkinkan dan idiomatic untuk menulis kondisi *if-else-if-else* menggunakan *switch*.

```
func unhex(c byte) byte {
    switch {
    case '0' <= c && c <= '9':
```

```

        return c - '0'
    case 'a' <= c && c <= 'f':
        return c - 'a' + 10
    case 'A' <= c && c <= 'F':
        return c - 'A' + 10
    }
    return 0
}

```

Tidak seperti C, *case* pada Go tidak otomatis jatuh ke bawah, namun kondisi *case* bisa lebih dari satu yang dipisahkan dengan koma,

```

func shouldEscape(c byte) bool {
    switch c {
    case ' ', '?', '&', '=', '#', '+', '%':
        return true
    }
    return false
}

```

Perintah *break* pada Go bisa digunakan untuk mengakhiri blok *switch*. Terkadang, perlu juga untuk keluar dari pengulangan, namun bukan dari *switch*, dan dalam Go hal ini bisa dilakukan dengan memberi label pada pengulangan dan "keluar" dari label tersebut. Contoh berikut memperlihatkan penggunaan kedua *break* tersebut.

```

Loop:
    for n := 0; n < len(src); n += size {
        switch {
        case src[n] < sizeOne:
            if validateOnly {
                break // keluar dari
switch, tapi tetap dalam pengulangan `for`
            }
            size = 1
            update(src[n])

        case src[n] < sizeTwo:
            if n+1 >= len(src) {
                err = errShortInput
                break Loop // keluar
dari `switch` dan pengulangan `for`
            }
            if validateOnly {
                break // keluar dari
switch, tapi tetap dalam pengulangan `for`
            }
            size = 2
            update(src[n] +
src[n+1]<&lt;shift)
        }
    }
}

```

Tentu saja, perintah *continue* juga dapat menggunakan label tapi hanya berlaku pada pengulangan. Untuk mengakhiri bagian ini, berikut fungsi yang membandingkan dua *slice byte* menggunakan dua perintah *switch*

```
// Compare mengembalikan sebuah integer hasil
pembandingan dari dua slice byte
// secara leksikografi.
// Hasilnya adalah 0 jika a == b, -1 jika a < b, dan +1
jika a > b .
func Compare(a, b []byte) int {
    for i := 0; i < len(a) && i < len(b); i++ {
        switch {
            case a[i] > b[i]:
                return 1
            case a[i] < b[i]:
                return -1
        }
    }
    switch {
        case len(a) > len(b):
            return 1
        case len(a) < len(b):
            return -1
    }
    return 0
}
```

## II. GUIDED

### 1. Soal Study Case

#### Source Code

```
package main

import
    "fmt"

func main() {

    nama := "Wafiq Nur Azizah"
    fmt.Print(nama)
}
```

#### Output

```
PS D:\Pratikum Alpro 2\Laparak 2\Unguided 3> go run "d:\Pratikum Alpro 2\Laparak 2\Unguided 3\main.go"
Wafiq Nur Azizah
PS D:\Pratikum Alpro 2\Laparak 2\Unguided 3>
```

#### Deskripsi

Program di atas adalah program sederhana dalam bahasa Go yang tugasnya mencetak nama "Wafiq Nur Azizah" ke layar. Di dalam program, ada variabel nama yang menyimpan teks nama tersebut, lalu `fmt.Print(nama)` digunakan untuk menampilkan isinya.

### 2. Soal Study 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.

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

func main() {
    correctOrder := []string{"merah", "kuning", "hijau",
    "ungu"}

    reader := bufio.NewReader(os.Stdin)
    success := true

    for i := 1; i <= 5; i++ {
        fmt.Print("Percobaan ", i, " : ")
        input, _ := reader.ReadString('\n')
        input = strings.TrimSpace(input)

        colors := strings.Split(input, " ")

        for j := 0; j < 4; j++ {
            if colors[j] != correctOrder[j] {
                success = false
                break
            }
        }
        if !success {
            break
        }
    }

    if success {
```

```

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

```

## Screenshoot Output

```

PS D:\Semester 3\P Alpro\Laparak 1\Unguided 2B.1> go run "d:\Semester 3\P Alpro\Laparak 1\Unguided 2B.1\main.go"
Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: merah kuning hijau ungu
Percobaan 5: merah kuning hijau ungu
Berhasil : True
PS D:\Semester 3\P Alpro\Laparak 1\Unguided 2B.1> go run "d:\Semester 3\P Alpro\Laparak 1\Unguided 2B.1\main.go"
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 D:\Semester 3\P Alpro\Laparak 1\Unguided 2B.1>

```

## Deskripsi Program

Program diatas digunakan untuk mencocokkan inputan pengguna dengan data pada array, program ini memiliki algoritma :

- Mendeklarasi sebuah *array* digunakan untuk sebuah data base dengan nama ***CorrectOrder***
- Lalu menambahkan ***bufio.NewReader*** yang digunakan untuk membaca inputan *user* secara panjang
- ***strings.Split*** digunakan untuk membaca inputan dengan sebuah spasi

Cara kerja program ini, user akan menginputkan sebuah warna kemudian program akan mencocokkan apakah warna yang diinput user sesuai dengan warna yang ada. Jika berhasil dan sama, maka program akan mengirim output true, begitu juga sebaliknya

## 3. Soal Study Case

### Source Code

```

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

```

1 2 3 4 5
Hasil Penjumlahan 1 2 3 4 5 adalah 15
PS D:\Semester 3\P Alpro\Laparak 1\Unguided 2C.3>

```

### Deskripsi Program

Program diatas melakukan penjumlahan dengan lima inputan *user*, Program diatas memiliki beberapa variable dengan tipe data yang sama, yaitu a,b,c,d,e, dan hasil. Program diatas memiliki fungsi hasil = a+b+c+d+e. Cara kerja program diatas adalah *user* akan menginputkan lima buah nilai secara random, kemudian program akan menghitung semua inputan lalu dikembalikan nilainya.

## 4. Soal Study Case

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

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

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

### Source Code

```

package main

import "fmt"

func main() {

```

```

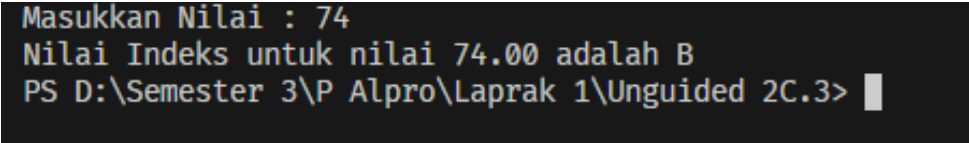
var nam float32
var nmk string
fmt.Print("Masukkan Nilai : ")
fmt.Scan(&nam)

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"
}

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

```

### Screenshoot Output



```

Masukkan Nilai : 74
Nilai Indeks untuk nilai 74.00 adalah B
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3>

```

### Deskripsi Program

Program ini digunakan untuk memilah sebuah nilai akhir dengan kategori indeks. Program ini memiliki algoritma sederhana yaitu :

- Dengan menggunakan 2 variable (nim float32, dan nmk string)
- Menggunakan if else, program ini memiliki banyak percabangan yang digunakan untuk memisahkan sebuah inputan nilai akhir yang akan dikasih indeks nilai

Cara kerja program ini adalah *user* akan menginputkan sebuah nilai akhir kemudian program akan menjalankan nilai akhir tersebut di kategori mana, program akan mencocokkan inputan *user* sesuai dengan kategori yang ada.

### III. UNGUIDED

#### 1. Soal Study 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

- a. 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.
- b. Modifikasi program sebelumnya, proses input akan berhenti apabila user mengetikkan 'SELESAI'. Kemudian tampilkan isi pita beserta banyaknya bunga yang ada di dalam pita

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

#### Source Code

```
package main

import (
    "fmt"
    "strings"
)

func main() {
    var (
        n      = 5
        bunga  string
        Pita   []string
        banyak int
    )
    //fmt.Print("N : ")input awal yang tidak terpakai
    //fmt.Scanln(&n) input awal yang tidak terpakai
    for i := 1; i <= n; i++ {
        fmt.Printf("Bunga %d: ", i)
        fmt.Scanln(&bunga)
        if bunga == "Selesai" || bunga == "SELESAI" {
            break
        }
        Pita = append(Pita, bunga)
    }
}
```

```
//digunakan untuk menginput nilai bunga kedalam array
Pita

        banyak++
    }
    temp := strings.Join(Pita, " - ")
//digunakan untuk menuliskan semua nilai array dengan
imbuan "-"
    fmt.Println("Pita : " + temp + " - ")
    fmt.Print("Bunga : ", banyak)
}
```

## Screenshoot Output

### a. Program a

```
PS D:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2> go run "d:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2\main.go"
N : 3
Bunga 1: Kertas
Bunga 2: Mawar
Bunga 3: Tulip
Pita : Kertas - Mawar - Tulip -
PS D:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2> go run "d:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2\main.go"
N : 0
Pita : -
PS D:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2> █
```

### b. Program b

```
PS D:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2> go run "d:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2\main.go"
Bunga 1: Kertas
Bunga 2: Mawar
Bunga 3: Tulip
Bunga 4: Selesai
Pita : Kertas - Mawar - Tulip -
Bunga : 3
PS D:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2> go run "d:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2\main.go"
Bunga 1: Selesai
Pita : -
Bunga : 0
PS D:\Semester 3\P Alpro\Laparak 1\Unguided 2B.2> █
```

## Deskripsi Program

Program ini dibuat untuk menginputkan nama sebuah bunga, kemudian bunga tersebut diletakkan ke sebuah Pita. Program diatas memiliki variable

- **n** diset dengan nilai 5 (yang awalnya sebuah inputan user) yang merepresentasikan jumlah iterasi maksimal untuk input bunga.
- **bunga** adalah variabel sementara yang digunakan untuk menyimpan input **bunga** dari pengguna.
- **Pita** adalah *slice/array* dari tipe *string* yang digunakan untuk menyimpan daftar bunga yang diinput.
- **banyak** digunakan untuk menghitung jumlah bunga yang dimasukkan ke dalam *array Pita*.

Program diatas memiliki sebuah kondisi dimana jika user menginputkan "SELESAI", Program akan berhenti.

Cara kerja program ini adalah *user* akan menginputkan nama bunga, kemudian data inputan *user* akan diterima oleh array **Pita** dan setiap inputan *user* yang masuk ke *array Pita* akan dihitung dengan variable **banyak**, kemudian terdapat sebuah percabangan jika *user* menginputkan selain "Selesai atau SELESAI" program akan terus menerus meminta *user* untuk menginputkan nama bunga yang kemudian nama bunga tersebut akan dipanggil pada tampilan **Pita**. Namun apabila *user* menginputkan "Selesai atau SELESAI" Program langsung berhenti, dan akan menampilkan bunga yang telah di isi.

## 2. Soal Study 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.
- 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"

func main() {
    var (
        a, b float32
    )
    for {
        fmt.Print("Masukan berat belanjaan di kedua kantong : ")
        fmt.Scanln(&a, &b)
        // if a == 9 || b == 9 {
        //     break
        // } else if a-b >= 9 {
```

```

        //      break
        // }
// modifikasi dari a
        if a < b {
            a, b = b, a
        }

        if a-b <= 9 {
            fmt.Println("Sepeda motor pak Andi akan
oleng : false")
        } else if a+b >= 150 || a < 0 || b < 0 {
            break
        } else if a-b >= 9 {
            fmt.Println("Sepeda motor pak Andi akan
oleng : true")
        }
    }
    fmt.Print("Proses Selesai.")
}

```

## Screenshoot Output

### a. Program a

```

PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2B.3> go run "d:\Semester 3\P Alpro\Laprak 1\Unguided 2B.3\main.go"
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.

```

### b. Program b

```

PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2B.3> go run "d:\Semester 3\P Alpro\Laprak 1\Unguided 2B.3\main.go"
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 D:\Semester 3\P Alpro\Laprak 1\Unguided 2B.3> 

```

## Deskripsi Program

Program diatas digunakan untuk menghitung selisih berat pada kedua kantong belanjaan, Program diatas memiliki beberapa algoritma

- **Variable**

Variable a dan b dengan tipe data *float32*, yang digunakan untuk menyimpang berat kantong kiri(a) dan kanan(b)

- **Looping Infinite**

Memiliki looping yang tidak terbatas, seperti pada perulangan *for* yang memiliki kondisi jika berat a dan b > 150kg atau a, b = negatif

- **IF**

Memiliki sebuah kondisi pertama yang dimana  $a-b \leq 9$ , output akan berupa “Sepeda motor pak Andi akan oleng : false”, kondisi kedua  $a+b \geq 150 \parallel a < 0 \parallel b < 0$  jika kondisi terpenuhi program akan berhenti dan mengakhiri *infinite looping*, kondisi ketiga yaitu  $a-b \geq 9$ , yang dimana akan keluar output “Sepeda motor pak Andi akan oleng : true”

Cara kerja dari program ini pertama tama *user* menginputkan nilai berat kantong kiri(a) dan kanan(b), kemudian program akan mengecek apakah berat  $a < b$ , maka nilai  $a, b = b, a$ . Program akan menukan nilai dari a menjadi b, untuk terhindar nilai negative disaat menghitung selisih. Kemudian inputan tadi akan melalui 3 kondisi yang ada. Setelah itu jika inputan memenuhi kondisi ke dua, maka program akan berhenti.

### 3. Soal Study Case

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

Buatlah sebuah program yang menerima Input sebuah bilangan sebagai K, kemudian menghitung dan menampilkan nilai  $f(K)$  sesuai persamaan di atas.

#### Source Code

```
package main

import (
    "fmt"
    "math"
)

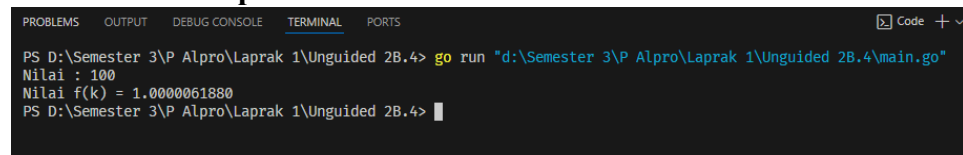
func main() {
    var (
        K, f float64
        //kenapa menggunakan 64, karena Pow, hanya menggunakan
        64
    )

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

    f = math.Pow(4*K+2, 2) / ((4*K + 1) * (4*K + 3))
```

```
//Powa adalah sebuah perhitungan math yang digunakan
untuk perpangkatan
    fmt.Printf("Nilai f(k) = %.10f", f)
//%.10f digunakan untuk membatasi panjang output hasil
}
```

## Screenshoot Output



## Deskripsi Program

Program ini digunakan untuk menghitung nilai fungsi dari inputan pengguna. Fungsi ini memiliki rumus  $f = 4 \cdot K^2 / ((4 \cdot K)(4 \cdot K + 3))$ . Program ini memiliki algoritma matematik sederhana yaitu :

- Variable  
K, F bertipe data float64, kenapa 64 math.pow hanya dapat menerima float dengan ukuran 64.
- Math.pow  
Eksetensi dari Import Math, yang digunakan untuk menghitung pangkat

Cara kerja program diatas adalah user akan menginputkan nilai dari K, kemudian program akan mengeksekusi nilai dari K tersebut dengan rumus fungsi yang ada lalu akan dikembalikan ke output dengan pembatas 10digit angka.

## 4. Soal Study Case

PT POS membutuhkan aplikasi perhitungan biaya kirim berdasarkan berat parcel. Maka, buatlah program BlayaPos untuk menghitung blaya 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 500am, 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



## Source Code

```
package main

import (
    "fmt"
)

func main() {
    var (
        total          int
        sisa, diskon, hargakg int
    )

    fmt.Print("Berat persel (gram) : ")
    fmt.Scanln(&total)
    kg := total / 1000
    grm := total % 1000
    fmt.Printf("Detail berat : %d kg + %d gram", kg,
grm)

    if total > 10000 {
        hargakg = kg * 10000
        if grm >= 500 {
            sisa = grm * 5
        } else if grm < 500 {
            sisa = grm * 15
        }
        fmt.Println("\nDetail biaya : Rp.", hargakg, "
+ Rp.", sisa)
        if sisa <= 1000 {
            diskon = 0
        }
        totalbiaya := hargakg + diskon
        fmt.Println("Total biaya : Rp.", totalbiaya)
    } else {
        hargakg = kg * 10000
        if grm >= 500 {
            sisa = grm * 5
        } else if grm < 500 {
            sisa = grm * 15
        }
        fmt.Println("\nDetail biaya : Rp.", hargakg, "
+ Rp.", sisa)
        totalbiaya := hargakg + sisa
        fmt.Println("Total biaya : Rp.", totalbiaya)
    }
}
```

## Screenshoot Output

```
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.1> go run "d:\Semester 3\P Alpro\Laprak 1\Unguided 2C.1\main.go"
Berat persel (gram) : 8250
Detail berat : 8 kg + 250 gram
Detail biaya : Rp. 80000 + Rp. 3750
Total biaya : Rp. 83750
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.1> go run "d:\Semester 3\P Alpro\Laprak 1\Unguided 2C.1\main.go"
Berat persel (gram) : 9250
Detail berat : 9 kg + 250 gram
Detail biaya : Rp. 90000 + Rp. 3750
Total biaya : Rp. 93750
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.1> go run "d:\Semester 3\P Alpro\Laprak 1\Unguided 2C.1\main.go"
Berat persel (gram) : 11750
Detail berat : 11 kg + 750 gram
Detail biaya : Rp. 110000 + Rp. 3750
Total biaya : Rp. 110000
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.1> █
```

## Deskripsi Program

Program ini digunakan untuk mengetahui berat gram menjadi kilogram (Kg) kemudian program akan menentukan harga perkg dan gramnya. Kemudian program akan memberikan diskon apabila pembelian lebih dari 10kg. Program ini memiliki algoritma :

- Input nilai pada variable total
- Program melakukan konversi ke kg dan gram
- Kemudian program akan menampilkan hasil konversi
- Setelah itu program akan memberikan biaya ke setiap kg, dan gram
- Program akan mengecek apakah pembelian pada variable total melebihi 10.000 atau 10kg, jika iya maka akan mendapatkan diskon untuk sisa gram yang kurang dari 1000 gram
- Kemudian program akan menampilkan hasil keseluruhan

## 5. Soal Study 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

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

## Source Code

```
package main

import "fmt"

func main() {
    var (
        b      int
        faktor []int
    )
    fmt.Print("Bilangan: ")
    fmt.Scanln(&b)
    if b > 0 {
        for i := 1; i <= b; i++ {
            if b%i == 0 {
                faktor = append(faktor, i)
            }
        }
        //append input nilai array
        fmt.Print("Faktor: ")
        for j := 0; j < len(faktor); j++ {
            //len merupakan panjang array
            fmt.Print(faktor[j], " ")
        }
        if len(faktor) == 2 {
            fmt.Print("\nPrima: true")
        } else {
            fmt.Print("\nPrima: false")
        }
    }
}
```

## Screenshot Output

### a. Program a

```
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3> go run "d:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3\main.go"
Bilangan: 12
Faktor: 1 2 3 4 6 12
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3> go run "d:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3\main.go"
Bilangan: 7
Faktor: 1 7
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3> █
```

### b. Program b

```
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3> go run "d:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3\main.go"
Bilangan: 12
Faktor: 1 2 3 4 6 12
Prima: false
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3> go run "d:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3\main.go"
Bilangan: 7
Faktor: 1 7
Prima: true
PS D:\Semester 3\P Alpro\Laprak 1\Unguided 2C.3> █
```

## Deskripsi Program

Program diatas digunakan untuk mengetahui nilai faktornya dan apakah bilangan tersebut sebuah bilangan prima. Algoritma program diatas adalah

- Inisialisasi sebuah variable (b merupakan inputan pengguna, dan faktor adalah sebuah array/slice)
- Menginputkan nilai dari sebuah bilangan (b)
- Program akan mengecek apakah nilai  $b > 0$
- Kemudian jika kondisi terpenuhi maka program akan melakukan perulangan untuk mengetahui faktorisasi dari bilangan tersebut
- Kemudian program akan menampilkan nilai faktor tersebut menggunakan perulangan yang dimana batas perulangan tersebut menggunakan panjang pada array faktor (len)
- Kemudian program akan menentukan apakah bilangan tersebut merupakan bilangan prima

#### **IV. DAFTAR PUSTAKA**

"Effective Go," Go Programming Language. [https://golang-id.org/doc/effective\\_go.html](https://golang-id.org/doc/effective_go.html). Diakses pada 6 Oktober 2024.