

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL I
REWIEV STRUKTUR KONTROL**



Disusun Oleh :

Siti Madina Halim Siregar / 2311102243

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 merupakan elemen fundamental dalam pemrograman yang memungkinkan programmer untuk mengendalikan alur program. Dengan struktur kontrol, programmer dapat menentukan kapan dan bagaimana kode program dijalankan. Dua jenis struktur kontrol utama adalah percabangan dan perulangan.

- **Percabangan (if-else)**

Struktur percabangan if-else memungkinkan program untuk mengambil keputusan berdasarkan kondisi tertentu. Berikut adalah contoh sintaks if-else dalam struktur if-else dapat digunakan dalam berbagai situasi, seperti:

- Memvalidasi input pengguna
- Menjalankan kode berdasarkan pilihan pengguna
- Mengatur alur program berdasarkan kondisi tertentu

- **Perulangan (for, while, do-while)**

Struktur perulangan memungkinkan program untuk menjalankan blok kode berulang kali. Ada tiga jenis struktur perulangan yang umum digunakan:

- **for:** Digunakan untuk mengulang blok kode sebanyak N kali.
- **while:** Digunakan untuk mengulang blok kode selama kondisi tertentu terpenuhi.
- **do-while:** Digunakan untuk mengulang blok kode minimal sekali, dan kemudian terus mengulangnya selama kondisi tertentu terpenuhi.

Golang, atau Go Language, adalah bahasa pemrograman yang dikembangkan oleh Google. Dikenalkan pada tahun 2009, Golang dirancang untuk menyederhanakan pengembangan perangkat lunak dengan fokus pada efisiensi, kejelasan, dan kecepatan. Berikut adalah penjelasan lebih mendalam tentang Golang.

Pengertian Golang

Golang adalah bahasa pemrograman open-source yang memiliki sintaksis sederhana namun kuat, memungkinkan pengembang untuk menulis kode dengan cepat dan efisien. Bahasa ini menggunakan tipe data statis dan menghasilkan kode biner yang dikompilasi, sehingga dapat berjalan dengan cepat dan efisien.

II. GUIDED

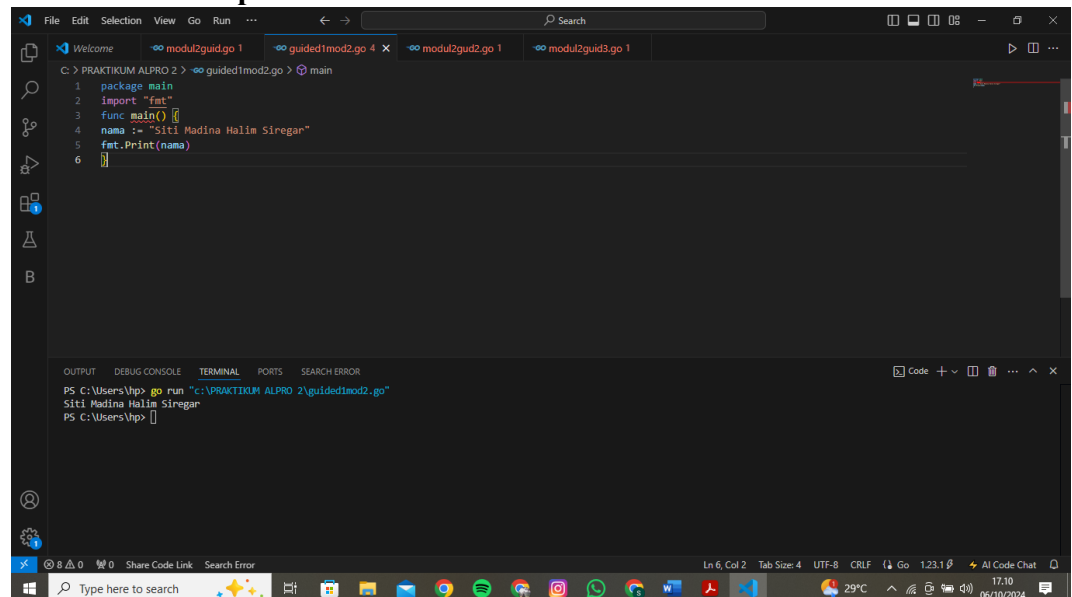
Soal Studi Case

Program membaca dan menampilkan nama.

Sourcecode

```
package main
import "fmt"
func main() {
    nama := "Siti Madina Halim Siregar"
    fmt.Print(nama)
}
```

Screenshoot Output



Deskripsi Program

- package main: Menyatakan bahwa file ini adalah bagian dari package utama, yang merupakan entry point program.
- import "fmt": Mengimpor package fmt, yang menyediakan fungsi untuk format input dan output.
- func main() : Mendefinisikan fungsi main, yang merupakan titik awal eksekusi program.
- nama = "Siti Madina Halim Siregar": Mendeklarasikan variabel nama dan menginisialisasinya dengan string "Siti Madina Halim Siregar" menggunakan operator penugasan pendek.
- fmt.Print(nama): Mencetak isi variabel nama ke konsol tanpa menambahkan newline di akhir.

Soal Studi Case

Program meminta pengguna untuk memasukkan urutan warna yang benar dalam lima percobaan. Jika pengguna berhasil memasukkan urutan warna yang benar dalam semua percobaan, Maka percobaan true dan jika salah maka false.

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

The screenshot shows a Go IDE with the following code in the editor:

```

1 package main
2
3 import (
4     "bufio"
5     "fmt"
6     "os"
7     "strings"
8 )
9
10 func main() {
11     //urutan warna yang benar
12     correctOrder := []string{"merah", "kuning", "hijau", "ungu"}
13
14     //membaca input untuk 5 percobaan
15     reader := bufio.NewReader(os.Stdin)
16     success := true
17
18     for i := 1; i <= 5; i++ {
19         fmt.Printf("Percobaan %d: ", i)

```

The terminal output shows the program running and the user inputting "merah kuning hijau ungu" for five consecutive attempts. The output indicates that the first attempt was successful (BERHASIL: true) and the subsequent attempts were not (BERHASIL: false).

```

PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\modul2guid.go"
Percobaan 1: merah kuning hijau ungu
BERHASIL: true
PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\modul2guid.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: false
PS C:\Users\hvp>

```

Deskripsi Program

- Package dan Import:
 - package main: Menyatakan bahwa ini adalah package utama.
 - import: Mengimpor beberapa package standar:
 - bufio: Untuk membaca input dari pengguna.
 - fmt: Untuk format output ke konsol.
 - os: Untuk akses ke sistem operasi, dalam hal ini untuk membaca input dari stdin.

- strings: Untuk manipulasi string.
- Fungsi Main:
 - Fungsi main() adalah titik masuk program.
- Deklarasi Variabel:
 - correctOrder: Sebuah slice string yang menyimpan urutan warna yang benar: {"merah", "kuning", "hijau", "ungu"}.
 - reader: Menggunakan bufio.NewReader untuk membuat reader yang akan membaca input dari pengguna.
 - success: Variabel boolean yang diinisialisasi dengan nilai true, digunakan untuk melacak apakah semua percobaan berhasil.
- Loop untuk Percobaan:
 - Menggunakan loop for untuk melakukan hingga 5 percobaan.
 - Pada setiap iterasi, program mencetak nomor percobaan menggunakan fmt.Printf.
- Membaca Input Pengguna:
 - Input = reader.ReadString('\n'): Membaca satu baris input dari pengguna hingga newline.
 - input = strings.TrimSpace(input): Menghapus spasi di awal dan akhir input.
- Memisahkan Input:
 - colors : strings.Split(input, " "): Memisahkan string input berdasarkan spasi dan menyimpannya dalam slice colors.
- Mengecek Urutan Warna:
 - Menggunakan loop untuk membandingkan setiap warna dalam slice colors dengan urutan warna yang benar (correctOrder).
 - Jika ada warna yang tidak sesuai, set variabel success ke false dan keluar dari loop.
- Menampilkan Hasil:
 - Setelah semua percobaan, program memeriksa nilai variabel success.
 - Jika semua percobaan berhasil, mencetak "BERHASIL: true".
 - Jika ada kesalahan pada salah satu percobaan, mencetak "BERHASIL: false".

Soal Studi Case

Program melakukan penjumlahan dari lima angka sekaligus

Sourcecode

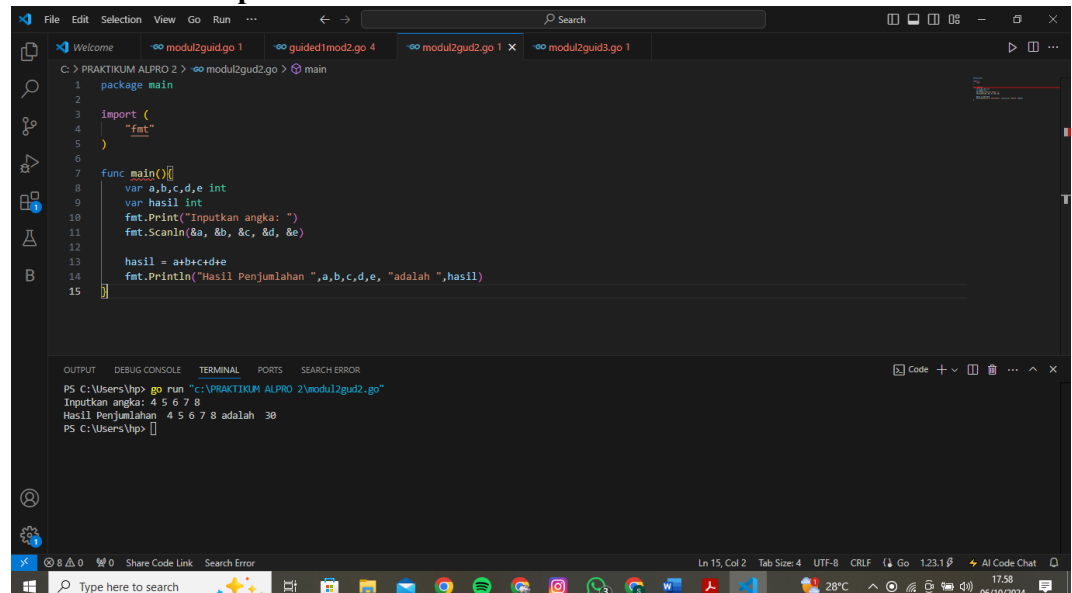
```
package main

import (
    "fmt"
)

func main(){
    var a,b,c,d,e int
    var hasil int
    fmt.Print("Inputkan angka: ")
    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



The screenshot displays a code editor with a Go program and its execution output. The code defines a `main` function that prompts the user for five integers, calculates their sum, and prints the result. The terminal output shows the program being run, the input `4 5 6 7 8`, and the resulting sum `30`.

```
C:\PRAKTIKUM ALPRO 2> go run "c:\PRAKTIKUM ALPRO 2\modul2gud2.go"
package main
1
2
3 import (
4     "fmt"
5 )
6
7 func main(){
8     var a,b,c,d,e int
9     var hasil int
10    fmt.Print("Inputkan angka: ")
11    fmt.Scanln(&a, &b, &c, &d, &e)
12
13    hasil = a+b+c+d+e
14    fmt.Println("Hasil Penjumlahan ",a,b,c,d,e, "adalah ",hasil)
15 }
```

OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

```
PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\modul2gud2.go"
Inputkan angka: 4 5 6 7 8
Hasil Penjumlahan 4 5 6 7 8 adalah 30
PS C:\Users\hvp>
```

Deskripsi Program

- package main: Menyatakan bahwa ini adalah package utama.
- import "fmt": Mengimpor package fmt untuk format input dan output.
- Deklarasi Variabel:
 - var a, b, c, d, e int: Mendeklarasikan lima variabel bertipe integer untuk menyimpan angka yang akan dijumlahkan.
 - var hasil int: Mendeklarasikan variabel untuk menyimpan hasil penjumlahan.
- Input Pengguna:
 - fmt.Print("Inputkan angka: "): Mencetak pesan untuk meminta pengguna memasukkan angka.
 - fmt.Scanln(&a, &b, &c, &d, &e): Membaca lima angka dari input pengguna dan menyimpannya dalam variabel a, b, c, d, dan e.
- Proses Penjumlahan:
 - hasil = a + b + c + d + e: Menjumlahkan semua angka yang dimasukkan dan menyimpannya dalam variabel hasil.

Soal Studi Case

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

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

```

1 package main
2
3 import "fmt"
4
5 func main() {
6     var nam float32
7     var nmk string
8
9     // Meminta input nilai
10    fmt.Print("Masukkan nilai : ")
11    fmt.Scan(&nam)
12
13    // Logika penentuan nilai huruf berdasarkan nilai numerik
14    if nam > 80 {
15        nmk = "A"
16    } else if nam > 72.5 {
17        nmk = "B"
18    } else if nam > 65 {
19        nmk = "C"
20    }
21
22    // Menampilkan hasil
23    fmt.Printf("Nilai Indeks untuk nilai %.2f adalah %s\n",
24    nam, nmk)
25 }

```

OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

```

PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\modul2guid3.go"
Masukkan nilai : 89
Nilai Indeks untuk nilai 89.00 adalah A
PS C:\Users\hvp>

```

Deskripsi Program

- Package dan Import:
 - package main: Menyatakan bahwa ini adalah package utama.
 - import "fmt": Mengimpor package fmt untuk format input dan output.
- Deklarasi Variabel:
 - var nam float32: Mendeklarasikan variabel nam bertipe float32 untuk menyimpan nilai numerik yang dimasukkan oleh pengguna.
 - var nmk string: Mendeklarasikan variabel nmk bertipe string untuk menyimpan nilai huruf (indeks) berdasarkan nilai numerik.
- Input Pengguna:

- `fmt.Print("Masukkan nilai : ")`: Mencetak pesan untuk meminta pengguna memasukkan nilai.
- `fmt.Scan(&nam)`: Membaca input dari pengguna dan menyimpannya dalam variabel `nam`.
- Logika Penentuan Nilai Huruf:
 - Menggunakan serangkaian pernyataan `if-else` untuk menentukan indeks huruf berdasarkan nilai yang dimasukkan:
 - Jika `nam` lebih dari 80, maka `nmk` di-set ke "A".
 - Jika lebih dari 72.5, maka di-set ke "B".
 - Jika lebih dari 65, maka di-set ke "C".
 - Jika lebih dari 50, maka di-set ke "D".
 - Jika lebih dari 40, maka di-set ke "E".
 - Jika tidak memenuhi semua kondisi di atas, maka di-set ke "F".

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 – 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.

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.

N: <u>3</u>	N : <u>0</u>
Bunga 1: <u>Kertas</u>	Pita :
Bunga 2: <u>Mawar</u>	
Bunga 3: <u>Tulip</u>	
Pita: Kertas – Mawar – Tulip –	

Modifikasi program sebelumnya, proses input akan berhenti apabila user 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: <u>Kertas</u>	Bunga 1: <u>SELESAI</u>
Bunga 2: <u>Mawar</u>	Pita :
Bunga 3: <u>Tulip</u>	Bunga: 0
Bunga 4: <u>SELESAI</u>	
Pita: Kertas – Mawar – Tulip –	
Bunga: 3	

Sourcecode

```
package main
package main

import (
    "fmt"
    "strings"
)

func main() {
```

```

var banyakBunga int
var bunga []string

for {
    var input string
    fmt.Print("Bunga (ketik 'SELESAI' untuk berhenti):
")

    fmt.Scanln(&input)

    if strings.ToUpper(input) == "SELESAI" {
        break
    }

    bunga = append(bunga, input)
    banyakBunga++
}

fmt.Println("Pita:", strings.Join(bunga, " - "))
fmt.Println("Banyaknya bunga:", banyakBunga)
}

```

Screenshoot Output

The screenshot displays a Go IDE with a code editor and a terminal window. The code in the editor is a Go program that counts the number of flowers entered by the user. The terminal shows the execution of the program, with the user entering 'Kertas', 'Mawar', and 'Tulip' as flower names, and 'SELESAI' as the termination signal. The program outputs the list of flowers and the total count, which is 3.

```

C:\> PRAKTIKUM ALPRO 2 > -o unguid1.go > main
1 package main
2
3 import (
4     "fmt"
5     "strings"
6 )
7
8 func main() {
9     var banyakBunga int
10    var bunga []string
11
12    for {
13        var input string
14        fmt.Print("Bunga (ketik 'SELESAI' untuk berhenti): ")
15        fmt.Scanln(&input)
16
17        if strings.ToUpper(input) == "SELESAI" {
18            break
19        }
20
21        bunga = append(bunga, input)
22        banyakBunga++
23    }
24
25    fmt.Println("Pita:", strings.Join(bunga, " - "))
26    fmt.Println("Banyaknya bunga:", banyakBunga)
27 }

```

OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

```

PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\unguid1.go"
Bunga (ketik 'SELESAI' untuk berhenti): Kertas
Bunga (ketik 'SELESAI' untuk berhenti): Mawar
Bunga (ketik 'SELESAI' untuk berhenti): Tulip
Bunga (ketik 'SELESAI' untuk berhenti): SELESAI
Pita: Kertas - Mawar - Tulip
Banyaknya bunga: 3
PS C:\Users\hvp>

```

Deskripsi Program

- Package dan Import:
 - package main: Menyatakan bahwa ini adalah package utama.
 - import: Mengimpor dua package:
 - fmt: Untuk format input dan output.
 - strings: Untuk manipulasi string, termasuk fungsi untuk mengubah string menjadi huruf besar.
- Deklarasi Variabel:
 - var banyakBunga int: Mendeklarasikan variabel bertipe integer untuk menghitung jumlah bunga yang dimasukkan.
 - var bunga []string: Mendeklarasikan slice bertipe string untuk menyimpan nama-nama bunga.
- Loop Tanpa Henti:
 - Menggunakan loop for yang akan terus berjalan hingga kondisi tertentu terpenuhi.
 - Di dalam loop, program meminta pengguna untuk memasukkan nama bunga.
- Input Pengguna:
 - fmt.Print: Mencetak pesan untuk meminta pengguna memasukkan nama bunga.
 - fmt.Scanln(&input): Membaca input dari pengguna dan menyimpannya dalam variabel input.
- Pemeriksaan Input:
 - if strings.ToUpper(input) == "SELESAI": Memeriksa apakah input yang diberikan pengguna adalah "SELESAI" (dalam huruf besar). Jika ya, loop akan dihentikan dengan perintah break.
- Menambahkan Nama Bunga:
 - bunga = append(bunga, input): Menambahkan nama bunga yang dimasukkan ke dalam slice bunga.
 - banyakBunga++: Meningkatkan jumlah bunga yang dimasukkan.

2. Soal Studi Case

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

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

Bilangan bulat $b > 0$ merupakan bilangan prima p jika dan hanya jika memiliki persis dua faktor bilangan saja, yaitu 1 dan dirinya sendiri. Lanjutkan program sebelumnya. Setelah menerima masukan sebuah bilangan bulat $b > 0$. Program tersebut mencari dan menampilkan semua faktor bilangan tersebut. Kemudian, program menentukan apakah b merupakan bilangan prima. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

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

Sourcecode

```
package main

import (
    "fmt"
)

func main() {
    var berat1, berat2 float64

    for {
        fmt.Print("Masukkan berat belanjaan di kedua kantong: ")
        _, err := fmt.Scan(&berat1, &berat2)
        if err != nil {
            fmt.Println("Input tidak valid. Harap masukkan dua angka.")
        }
    }
}
```

```

        fmt.Scanln()
        continue
    }

    if berat1 < 0 || berat2 < 0 {
        fmt.Println("Salah satu kantong memiliki berat
negatif. Program berhenti.")
        break
    }
    totalBerat := berat1 + berat2
    if totalBerat > 150 {
        fmt.Println("Total berat lebih dari 150 kg.
Program berhenti.")
        break
    }

    selisih := berat1 - berat2
    if selisih < 0 {
        selisih = -selisih
    }

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

    if berat1 >= 9 || berat2 >= 9 {
        fmt.Println("Salah satu kantong lebih dari atau
sama dengan 9 kg. Program berhenti.")
        break
    }
}

fmt.Println("Program selesai.")
}

```

Screenshoot Output


```
7 func main() {
25     totalBerat := berat1 + berat2
26     if totalBerat > 150 {
27         fmt.Println("Total berat lebih dari 150 kg. Program berhenti.")
28         break
29     }
30
31     selisih := berat1 - berat2
32     if selisih < 0 {
33         selisih = -selisih
34     }
35
36     if selisih >= 9 {
37         fmt.Println("Sepeda motor akan oleng: true")
38     } else {
39         fmt.Println("Sepeda motor akan oleng: false")
40     }
41
42     if berat1 >= 9 || berat2 >= 9 {
```

OUTPUT

```
PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\unguided2.go"
Masukkan berat belanja di kedua kantong: 5.0 1.0
Sepeda motor akan oleng: false
Masukkan berat belanja di kedua kantong: 2.6
Sepeda motor akan oleng: false
Masukkan berat belanja di kedua kantong: 7.6 1
Sepeda motor akan oleng: false
Masukkan berat belanja di kedua kantong: 50 100
Sepeda motor akan oleng: true
Salah satu kantong lebih dari atau sama dengan 9 kg. Program berhenti.
Program selesai.
PS C:\Users\hvp>
```

Deskripsi Program

- package main: Menyatakan bahwa ini adalah package utama.
- import "fmt": Mengimpor package fmt untuk format input dan output.
- var berat1, berat2 float64: Mendeklarasikan dua variabel bertipe float64 untuk menyimpan berat belanjaan di kedua kantong.
- Menggunakan loop for yang akan terus berjalan hingga kondisi tertentu terpenuhi.
- fmt.Print: Mencetak pesan untuk meminta pengguna memasukkan berat belanjaan.
- err := fmt.Scan(&berat1, &berat2): Membaca dua angka dari input pengguna dan menyimpannya dalam variabel berat1 dan berat2. Jika terjadi kesalahan, variabel err akan berisi informasi tentang kesalahan tersebut.
- Jika ada kesalahan (misalnya, pengguna tidak memasukkan dua angka), program mencetak pesan kesalahan dan menggunakan fmt.Scanln() untuk membersihkan input sebelum melanjutkan ke iterasi berikutnya.
- totalBerat : berat1 + berat2: Menghitung total berat dari kedua kantong.
- Jika total berat lebih dari 150 kg, program mencetak pesan dan menghentikan loop.
- selisih : berat1 - berat2: Menghitung selisih antara kedua berat.
- Jika selisih negatif, maka diubah menjadi positif dengan selisih = -selisih.
- Jika selisih lebih besar atau sama dengan 9 kg, program mencetak "Sepeda motor akan oleng: true".

- Jika tidak, mencetak "Sepeda motor akan oleng: false".
- Jika salah satu kantong memiliki berat 9 kg atau lebih, program mencetak pesan dan menghentikan loop.

3. Soal Studi 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. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

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

Sourcecode

```
package main

import (
    "fmt"
)

func calculate(k float64) float64 {

    part1 := (4*k + 2) * (4*k + 2)
    part2 := (4*k + 1) * (4*k + 3)

    return part1 * part2
}

func main() {
    var k float64

    fmt.Print("Nilai K = ")
    _, err := fmt.Scanln(&k)
```

```

    if err != nil {
        fmt.Println("Input tidak valid, harap masukkan
angka.")
        return
    }

    f_k := calculateF(k)

    fmt.Printf("Nilai akar 2 = %.10f\n", f_k)
}

```

Screenshoot Output

The screenshot shows a Go IDE with the following code in the editor:

```

1 package main
2
3 import (
4     "fmt"
5 )
6
7 func calculateF(k float64) float64 {
8
9     part1 := (4*k + 2) * (4*k + 2)
10    part2 := (4*k + 1) * (4*k + 3)
11
12    return part1 * part2
13 }
14
15 func main() {
16     var k float64
17
18     fmt.Print("Nilai K = ")
19     , err := fmt.Scanln(&k)
20 }

```

The terminal output shows the program being run three times with different values of k:

```

PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\unguided3.go"
Nilai K = 10
Nilai akar 2 = 3109932.0000000000
PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\unguided3.go"
Nilai K = 100
Nilai akar 2 = 26115691212.0000000000
PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\unguided3.go"
Nilai K = 1000
Nilai akar 2 = 256512368112012.0000000000
PS C:\Users\hvp>

```

Deskripsi Program

- import "fmt": Mengimpor package fmt untuk format input dan output.
- Menerima parameter k bertipe float64.
- Menghitung nilai berdasarkan rumus yang melibatkan dua bagian (part1 dan part2) dan mengembalikan hasil perkalian keduanya.
- Deklarasi variabel k untuk menyimpan input dari pengguna.
- Meminta pengguna untuk memasukkan nilai k.
- Jika input tidak valid, mencetak pesan kesalahan dan keluar dari program.
- Memanggil fungsi calculateF dengan nilai k dan menyimpan hasilnya dalam f_k.

- Mencetak hasil dengan format 10 desimal.

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

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

Sourcecode

```
package main

import (
    "fmt"
)

func main() {
    var berat int
```

```
fmt.Print("Berat parsel (gram): ")
fmt.Scan(&berat)

totalKg := berat / 1000
sisagram := berat % 1000

biayaKg := totalKg * 10000
biayaSisa := 0

if totalKg > 10 {
    biayaSisa = 0
} else {
    if sisagram >= 500 {

        biayaSisa = sisagram * 5
    } else {
        biayaSisa = sisagram * 15
    }
}

totalBiaya := biayaKg + biayaSisa

fmt.Printf("\nDetail berat: %d kg + %d gr\n", totalKg,
sisagram)
fmt.Printf("Detail biaya: Rp. %d + Rp. %d\n", biayaKg,
biayaSisa)
fmt.Printf("Total biaya: Rp. %d\n", totalBiaya)
}
```

Screenshoot Output

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 func main() {
8     var berat int
9
10    fmt.Print("Berat parcel (gram): ")
11    fmt.Scan(&berat)
12
13    totalKg := berat / 1000
14    sisaGram := berat % 1000
15
16    biayaKg := totalKg * 10000
17    biayaSisa := 0
18
19    if totalKg > 10 {
20        // Additional logic for high weight
21    }
22 }
```

OUTPUT

```
PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\unguided4.go"
Berat parcel (gram): 5800
Detail berat: 5 kg + 800 gr
Detail biaya: Rp. 50000 + Rp. 4000
Total biaya: Rp. 54000
PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\unguided4.go"
Berat parcel (gram): 9250
Detail berat: 9 kg + 250 gr
Detail biaya: Rp. 90000 + Rp. 3750
Total biaya: Rp. 93750
PS C:\Users\hvp>
```

Deskripsi Program

- import "fmt": Mengimpor package fmt untuk format input dan output.
- Menghitung total berat dalam kilogram (totalKg) dan sisa berat dalam gram (sisaGram).
- Menghitung biaya berdasarkan berat kilogram, dengan tarif Rp. 10.000 per kilogram.
- Jika berat lebih dari 10 kg, biaya sisa diatur ke 0.
- Jika tidak, menghitung biaya untuk sisa gram:
- Jika sisa gram ≥ 500 , dikenakan tarif Rp. 5 per gram.
- Jika kurang dari 500 gram, dikenakan tarif Rp. 15 per gram.
- Menghitung total biaya dengan menjumlahkan biaya kilogram dan biaya sisa.

5. Soal Studi Case

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

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

Bilangan bulat $b > 0$ merupakan bilangan prima p jika dan hanya jika memiliki persis dua faktor bilangan saja, yaitu 1 dan dirinya sendiri. Lanjutkan program sebelumnya. Setelah menerima masukan sebuah bilangan bulat $b > 0$. Program tersebut mencari dan menampilkan semua faktor bilangan tersebut. Kemudian, program menentukan apakah b merupakan bilangan prima. Perhatikan contoh sesi interaksi program seperti di bawah ini (teks bergaris bawah adalah input/read):

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

Sourcecode

```
package main

import (
    "fmt"
)

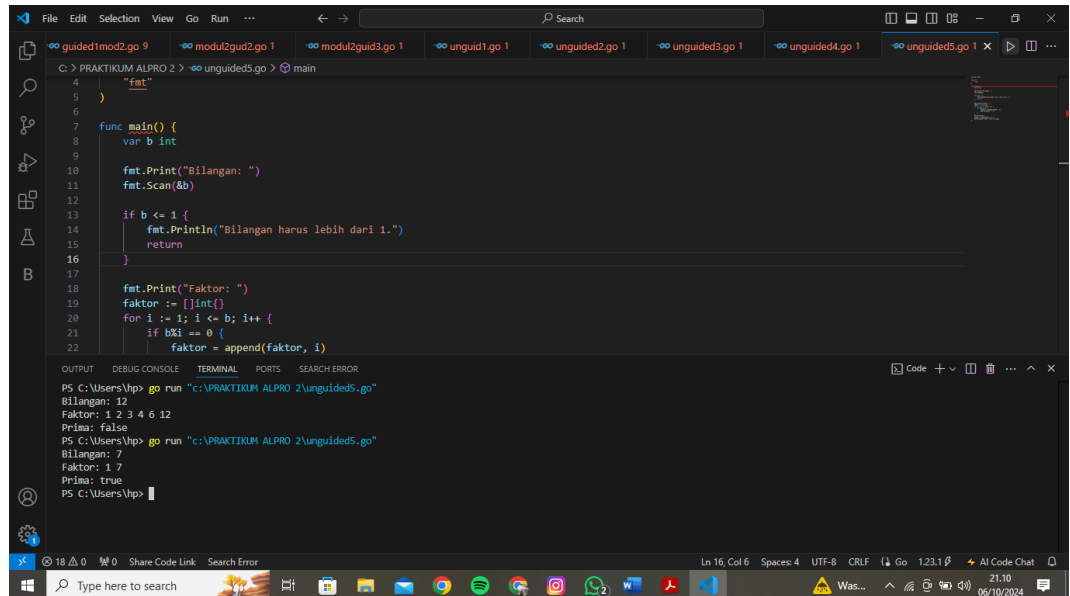
func main() {
    var b int

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

    if b <= 1 {
        fmt.Println("Bilangan harus lebih dari 1.")
        return
    }

    fmt.Print("Faktor: ")
    faktor := []int{}
    for i := 1; i <= b; i++ {
        if b%i == 0 {
            faktor = append(faktor, i)
            fmt.Print(i, " ")
        }
    }
    fmt.Println()
    prima := len(faktor) == 2
    fmt.Printf("Prima: %t\n", prima)
}
```

Screenshoot Output



The screenshot shows a Go IDE with a file explorer at the top displaying several files: `guided1mod2.go`, `modul2gud2.go`, `modul2gud3.go`, `unguid1.go`, `unguid2.go`, `unguid3.go`, `unguid4.go`, and `unguid5.go`. The main editor window shows the source code for `unguid5.go`. The code defines a `main` function that takes an integer `b` as input. It prints the input, checks if `b` is less than or equal to 1 (printing an error if so), and then finds all factors of `b` using a loop. The factors are stored in a slice and printed. Finally, it checks if the slice has exactly two elements (1 and `b`) to determine if `b` is a prime number.

```
4  "fmt"
5  )
6
7  func main() {
8      var b int
9
10     fmt.Print("Bilangan: ")
11     fmt.Scan(&b)
12
13     if b <= 1 {
14         fmt.Println("Bilangan harus lebih dari 1.")
15         return
16     }
17
18     fmt.Print("Faktor: ")
19     faktor := []int{}
20     for i := 1; i <= b; i++ {
21         if b%i == 0 {
22             faktor = append(faktor, i)
23         }
24     }
25
26     fmt.Print(faktor)
27     if len(faktor) == 2 {
28         fmt.Println("Prima: true")
29     } else {
30         fmt.Println("Prima: false")
31     }
32 }
```

The terminal output shows two runs of the program. The first run takes input 12 and outputs factors [1, 2, 3, 4, 6, 12], which is not prime. The second run takes input 7 and outputs factors [1, 7], which is prime.

```
PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\unguid5.go"
Bilangan: 12
Faktor: 1 2 3 4 6 12
Prima: false
PS C:\Users\hvp> go run "c:\PRAKTIKUM ALPRO 2\unguid5.go"
Bilangan: 7
Faktor: 1 7
Prima: true
PS C:\Users\hvp>
```

Deskripsi Program

- Mengimpor package `fmt` untuk melakukan input dan output.
- Mendeklarasikan variabel `b` bertipe integer untuk menyimpan bilangan yang diinput oleh pengguna.
- Menggunakan `fmt.Print` untuk meminta pengguna memasukkan bilangan.
- Menggunakan `fmt.Scan` untuk membaca input dari pengguna dan menyimpannya dalam variabel `b`.
- Memeriksa apakah bilangan yang dimasukkan kurang dari atau sama dengan 1. Jika iya, program akan mencetak pesan kesalahan dan keluar.
- Menginisialisasi slice faktor untuk menyimpan faktor-faktor dari bilangan `b`.
- Menggunakan loop dari 1 hingga `b` untuk mencari faktor-faktor dengan memeriksa apakah `b` dapat dibagi habis oleh `i`.
- Jika ya, faktor tersebut ditambahkan ke slice dan dicetak.
- Bilangan dianggap prima jika memiliki tepat dua faktor (1 dan bilangan itu sendiri).
- Program memeriksa panjang slice faktor. Jika panjangnya 2, maka bilangan tersebut adalah prima, dan hasilnya dicetak.

Daftar Pustaka

<https://an-nur.ac.id/struktur-kontrol-percabangan-dan-perulangan/>