

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

MODUL IV

PROSEDUR



Disusun Oleh :

Daffa Aryaputra / 2311102272

IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Konsep prosedur diimplementasikan melalui fungsi. Go adalah bahasa yang mengutamakan kesederhanaan dan efisiensi, dan pendekatan fungsinya mencerminkan filosofi ini. Fungsi dalam Go didefinisikan menggunakan kata kunci `func`, diikuti oleh nama fungsi, daftar parameter (jika ada), tipe pengembalian (jika ada), dan blok kode yang membentuk isi fungsi.

Go mendukung beberapa fitur unik terkait fungsi yang membuatnya powerful dan fleksibel. Salah satu fiturnya adalah kemampuan untuk mengembalikan multiple values, yang sangat berguna untuk menangani error secara eksplisit. Go juga mendukung fungsi anonim dan closures, memungkinkan pembuatan fungsi di dalam fungsi lain dan capture variabel dari lingkup sekitarnya. Selain itu, Go memperkenalkan konsep defer, yang memungkinkan penundaan eksekusi suatu pernyataan hingga fungsi selesai, sangat berguna untuk clean-up resources.

Penggunaan fungsi dalam Go mempromosikan gaya pemrograman yang bersih dan modular. Fungsi-fungsi biasanya dirancang untuk melakukan satu tugas spesifik, mengikuti prinsip "Do One Thing and Do It Well". Go juga mendorong penggunaan package untuk mengorganisir kode ke dalam unit-unit yang dapat digunakan kembali, di mana fungsi-fungsi terkait dikelompokkan bersama. Pendekatan ini, dikombinasikan dengan sistem tipe statis Go dan kompilasi cepat, membuat pengembangan aplikasi skala besar menjadi lebih efisien dan mudah dikelola.

II. GUIDED

1.

Sourcecode

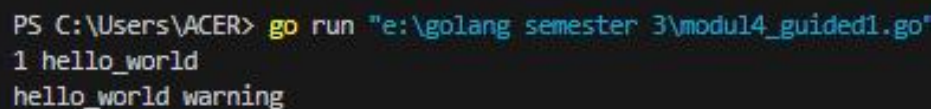
```
package main

import "fmt"

func main() {
    var bilangan int
    var pesan string
    fmt.Scan(&bilangan, &pesan)
    cetakPesan(pesan, bilangan)
}

func cetakPesan(M string, flag int) {
    var jenis string = ""
    if flag == 0 {
        jenis = "error"
    } else if flag == 1 {
        jenis = "warning"
    } else if flag == 2 {
        jenis = "informasi"
    }
    fmt.Println(M, jenis)
}
```

Screenshoot Output



```
PS C:\Users\ACER> go run "e:\golang semester 3\modul4_guided1.go"
1 hello_world
hello_world warning
```

Deskripsi Program

Ini program sederhana dalam bahasa Golang yang bertujuan untuk mencetak pesan dengan kategori tertentu berdasarkan input dari pengguna. Program ini menggunakan algoritma percabangan sederhana untuk menentukan jenis pesan yang akan dicetak.

Cara kerja program ini dimulai dengan meminta pengguna memasukkan dua nilai: sebuah angka (bilangan) dan sebuah pesan (string). Setelah menerima input, program memanggil fungsi cetakPesan dengan dua argumen tersebut. Fungsi cetakPesan kemudian menggunakan struktur

percabangan if-else untuk menentukan jenis pesan berdasarkan nilai flag (bilangan) yang dimasukkan. Jika nilai flag adalah 0, jenis pesan ditetapkan sebagai "error". Jika nilai flag adalah 1, jenis pesan menjadi "warning". Jika nilai flag adalah 2, jenis pesan dikategorikan sebagai "informasi". Akhirnya, program mencetak pesan yang dimasukkan pengguna bersama dengan jenis pesannya ke layar menggunakan `fmt.Println`. Namun, perlu diperhatikan bahwa ada kesalahan sintaks dalam penugasan jenis "error" yang mungkin menyebabkan program tidak berfungsi sebagaimana mestinya untuk kasus tersebut.

2.

Sourcecode

```
package main

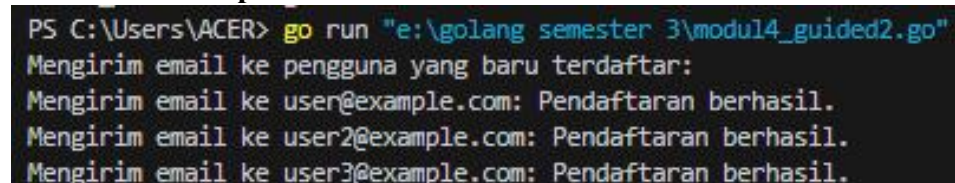
import "fmt"

func sendEmailNotification(email string) {
    fmt.Printf("Mengirim email ke %s: Pendaftaran berhasil.\n", email)
}

func main() {
    emails := []string{"user@example.com", "user2@example.com",
"user3@example.com"}

    fmt.Println("Mengirim email ke pengguna yang baru terdaftar: ")
    for _, email := range emails {
        sendEmailNotification(email)
    }
}
```

Screenshot Output



```
PS C:\Users\ACER> go run "e:\golang semester 3\modul4_guided2.go"
Mengirim email ke pengguna yang baru terdaftar:
Mengirim email ke user@example.com: Pendaftaran berhasil.
Mengirim email ke user2@example.com: Pendaftaran berhasil.
Mengirim email ke user3@example.com: Pendaftaran berhasil.
```

Deskripsi Program

Program ini adalah sebuah simulasi sistem notifikasi email sederhana yang ditulis dalam bahasa Go. Tujuan utamanya adalah untuk mengirimkan pemberitahuan melalui email kepada beberapa pengguna yang baru saja

mendaftar. Program ini menggunakan algoritma perulangan sederhana untuk memproses daftar email pengguna dan mengirimkan notifikasi ke masing-masing alamat.

Cara kerja program dimulai dengan mendefinisikan sebuah slice (array dinamis) yang berisi tiga alamat email. Kemudian, program menggunakan perulangan for untuk mengiterasi melalui setiap alamat email dalam slice tersebut. Untuk setiap email, program memanggil fungsi `sendEmailNotification` yang menerima alamat email sebagai parameter. Fungsi ini mensimulasikan pengiriman email dengan mencetak pesan ke konsol yang menunjukkan bahwa email telah dikirim ke alamat tertentu. Output program akan menampilkan pesan "Mengirim email ke pengguna yang baru terdaftar: " diikuti oleh tiga baris, masing-masing menunjukkan pengiriman email ke setiap alamat dalam daftar. Meskipun program ini tidak benar-benar mengirim email, ia memberikan gambaran bagaimana sistem notifikasi email otomatis bisa bekerja dalam aplikasi yang lebih besar.

3.

Sourcecode

```
package main

import "fmt"

func f1(x, y int) float64 {
    var hasil float64
    hasil = float64(2*x) - 0.5*float64(y) + 3.0
    return hasil
}

func f2(x, y int, hasil *float64) {
    *hasil = float64(2*x) - 0.5*float64(y) + 3.0
}

func main() {
    var a, b int
    var c float64

    fmt.Printf("Enter two integers: ")
    fmt.Scan(&a, &b)

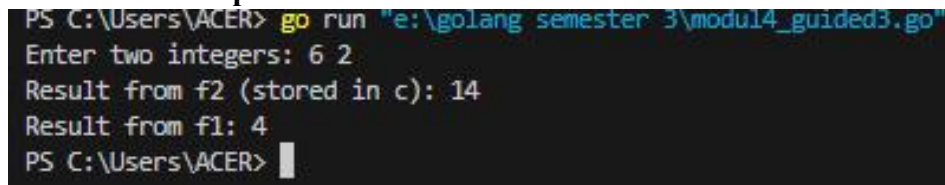
    // Pass the address of c to the f2 function
```

```
f2(a, b, &c)

fmt.Println("Result from f2 (stored in c):", c)

resultF1 := f1(b, a)
fmt.Println("Result from f1:", resultF1)
}
```

Screenshoot Output



```
PS C:\Users\ACER> go run "e:\golang semester 3\modul4_guided3.go"
Enter two integers: 6 2
Result from f2 (stored in c): 14
Result from f1: 4
PS C:\Users\ACER> 
```

Deskripsi Program

Ini Program sederhana dalam bahasa Go yang mendemonstrasikan dua cara berbeda untuk menghitung dan mengembalikan hasil dari sebuah rumus matematika. Program ini menggunakan dua fungsi, f1 dan f2, yang keduanya menghitung nilai berdasarkan rumus yang sama: $2x - 0.5y + 3$, di mana x dan y adalah bilangan bulat yang dimasukkan oleh pengguna. Perbedaan utama antara kedua fungsi ini terletak pada cara mereka mengembalikan hasil perhitungan.

Cara kerja program dimulai dengan meminta pengguna memasukkan dua bilangan bulat. Kemudian, program memanggil fungsi f2, yang menggunakan pointer untuk menyimpan hasil perhitungan langsung ke variabel c di fungsi main. Hasil ini kemudian dicetak. Selanjutnya, program memanggil fungsi f1 dengan argumen yang dibalik (b dan a, bukan a dan b), yang mengembalikan hasil perhitungan sebagai nilai balik fungsi. Hasil ini juga dicetak. Penggunaan pointer dalam f2 menunjukkan cara langsung untuk memodifikasi variabel di luar fungsi, sementara f1 mendemonstrasikan cara tradisional mengembalikan nilai dari sebuah fungsi. Output program akan menampilkan hasil dari kedua perhitungan ini, memungkinkan perbandingan antara kedua metode dan hasil yang diperoleh dari urutan input yang berbeda.

III. UNGUIDED

1.

Sourcecode

```
package main

import (
    "fmt"
)

func factorial(n int, hasil *int) {
    *hasil = 1
    for i := 2; i <= n; i++ {
        *hasil *= i
    }
}

func permutation(n, r int, hasil *int) {
    var factN, factNR int
    factorial(n, &factN)
    factorial(n-r, &factNR)
    *hasil = factN / factNR
}

func combination(n, r int, hasil *int) {
    var perm, factR int
    permutation(n, r, &perm)
    factorial(r, &factR)
    *hasil = perm / factR
}

func main() {
    var a, b, c, d int
    fmt.Println("Masukkan empat bilangan bulat positif (a b c d):")
    fmt.Scan(&a, &b, &c, &d)

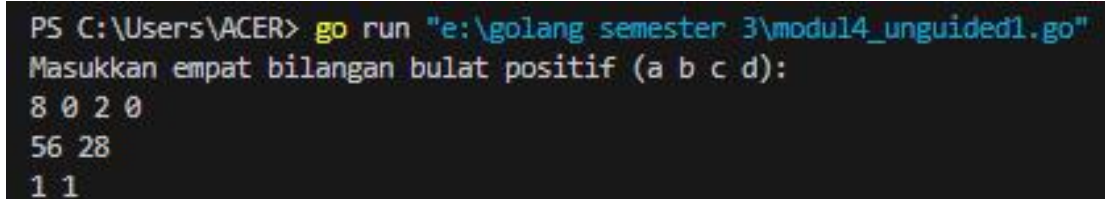
    if a >= c && b >= d {
        var permAC, combAC, permBD, combBD int
        permutation(a, c, &permAC)
        combination(a, c, &combAC)
        permutation(b, d, &permBD)
        combination(b, d, &combBD)
```

```

        fmt.Printf("%d %d\n", permAC, combAC)
        fmt.Printf("%d %d\n", permBD, combBD)
    } else {
        fmt.Println("Input tidak memenuhi syarat a >= c dan b >=
d")
    }
}

```

Screenshot Output



```

PS C:\Users\ACER> go run "e:\golang semester 3\modul4_unguided1.go"
Masukkan empat bilangan bulat positif (a b c d):
8 0 2 0
56 28
1 1

```

Deskripsi Program

Program ini adalah sebuah aplikasi sederhana untuk menghitung permutasi dan kombinasi berdasarkan input dari pengguna. Program ini dirancang untuk membantu mahasiswa Fakultas Informatika dalam mempelajari dan mengimplementasikan konsep kombinasi dan permutasi dalam matematika diskrit. Program menggunakan tiga fungsi utama: factorial untuk menghitung faktorial, permutation untuk menghitung permutasi, dan combination untuk menghitung kombinasi. Algoritma yang digunakan dalam program ini adalah perhitungan faktorial secara iteratif dan penerapan rumus permutasi dan kombinasi menggunakan hasil faktorial.

Cara kerja program dimulai dengan meminta pengguna memasukkan empat bilangan bulat positif (a, b, c, d). Program kemudian memeriksa apakah input memenuhi syarat $a \geq c$ dan $b \geq d$. Jika syarat terpenuhi, program menghitung permutasi dan kombinasi a terhadap c, serta b terhadap d menggunakan fungsi-fungsi yang telah didefinisikan. Perhitungan dilakukan dengan memanfaatkan pointer untuk menyimpan hasil, sesuai dengan spesifikasi prosedur yang diminta. Hasil perhitungan kemudian ditampilkan dalam dua baris output: baris pertama menunjukkan hasil permutasi dan kombinasi a terhadap c, sedangkan baris kedua menunjukkan hasil untuk b terhadap d. Jika input tidak memenuhi syarat, program akan menampilkan pesan kesalahan. Program ini mendemonstrasikan penggunaan prosedur, pointer, dan perhitungan matematika dasar dalam Go, serta interaksi sederhana dengan pengguna melalui input/output konsol.

2.

Sourcecode

```
package main

import (
    "fmt"
    "strings"
)

func hitungSkor(soal *int, skor *int) string {
    var nama string
    var waktu [8]int
    *soal = 0
    *skor = 0

    fmt.Scan(&nama)
    for i := 0; i < 8; i++ {
        fmt.Scan(&waktu[i])
        if waktu[i] != 301 {
            (*soal)++
            *skor += waktu[i]
        }
    }
    return nama
}

func main() {
    var pemenang string
    var maxSoal, minWaktu int

    for {
        var soal, skor int

        // Panggil fungsi hitungSkor dan ambil nama
        nama := hitungSkor(&soal, &skor)

        // Cek apakah nama adalah "selesai"
        if strings.ToLower(nama) == "selesai" {
            break
        }
    }
}
```

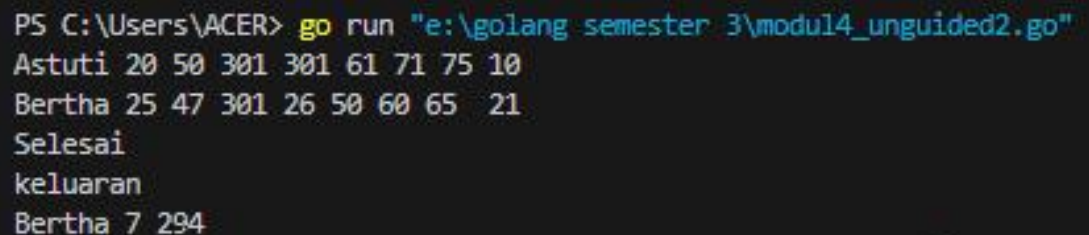
```

        // Update pemenang jika memenuhi kriteria
        if soal > maxSoal || (soal == maxSoal && skor <
minWaktu) {
            pemenang = nama
            maxSoal = soal
            minWaktu = skor
        }
    }

    fmt.Printf("%s %d %d\n", pemenang, maxSoal, minWaktu)
}

```

Screenshoot Output



```

PS C:\Users\ACER> go run "e:\golang semester 3\modul4_unguided2.go"
Astuti 20 50 301 301 61 71 75 10
Bertha 25 47 301 26 50 60 65 21
Selesai
keluaran
Bertha 7 294

```

Deskripsi Program

Program yang dibuat adalah program untuk menentukan pemenang dari sebuah kompetisi yang melibatkan penyelesaian soal dalam waktu tertentu. Setiap peserta akan diberikan sejumlah soal, dan mereka akan dicatat waktunya untuk setiap soal. Jika peserta menyelesaikan soal dalam waktu kurang dari 301 detik, soal itu dianggap terjawab dan skornya dihitung berdasarkan waktu yang dibutuhkan. Program ini terus menerima input nama peserta dan waktu pengerjaan soal hingga peserta dengan nama "selesai" dimasukkan. Pada akhir program, pemenang adalah peserta yang menyelesaikan soal terbanyak, atau jika ada jumlah soal yang sama, peserta dengan waktu total paling singkat yang menang.

Algoritma program bekerja sebagai berikut: Pertama, program meminta input nama peserta dan waktu pengerjaan soal untuk delapan soal. Jika peserta menyelesaikan soal dengan waktu di bawah 301 detik, soal dianggap valid dan waktu tersebut ditambahkan ke skor total. Setiap kali data peserta dimasukkan, program membandingkan jumlah soal yang dijawab peserta dengan peserta sebelumnya. Jika ada peserta yang menjawab lebih banyak soal, atau memiliki waktu lebih sedikit dengan jumlah soal yang sama, maka peserta tersebut menjadi pemenang sementara. Proses ini berulang hingga nama "selesai" dimasukkan, lalu program menampilkan nama pemenang, jumlah soal yang berhasil

diselesaikan, dan total waktu yang digunakan untuk menyelesaikan soal-soal tersebut.

3.

Sourcecode

```
package main

import (
    "fmt"
)

func cetakDeret(n int) {
    for n != 1 {
        fmt.Printf("%d ", n)
        if n%2 == 0 {
            n = n / 2
        } else {
            n = 3*n + 1
        }
    }
    fmt.Println("1")
}

func main() {
    var n int
    fmt.Print("Masukkan bilangan awal (1-999999): ")
    fmt.Scan(&n)

    if n > 0 && n < 1000000 {
        cetakDeret(n)
    } else {
        fmt.Println("Masukan tidak valid. Masukkan bilangan antara 1 dan 999999.")
    }
}
```

Screenshoot Output

```
PS C:\Users\ACER> go run "e:\golang semester 3\modul3_unguided3.go"
Masukkan bilangan awal (1-999999): 22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS C:\Users\ACER> █
```

Deskripsi Program

Program ini dibuat untuk mengimplementasikan Programming Challenge yang didefinisikan oleh Skiena dan Revilla. Program ini menghasilkan deret bilangan berdasarkan aturan tertentu: jika bilangan genap, suku berikutnya adalah setengahnya; jika ganjil, suku berikutnya adalah $3n+1$. Deret berakhir ketika mencapai angka 1. Program ini menggunakan algoritma sederhana yang menerapkan aturan tersebut secara berulang untuk menghasilkan deret bilangan.

Cara kerja program dimulai dengan meminta pengguna memasukkan bilangan awal antara 1 dan 999999. Program kemudian memanggil prosedur cetakDeret yang mengimplementasikan logika untuk menghasilkan deret. Prosedur ini menggunakan loop untuk terus menghasilkan bilangan berikutnya berdasarkan aturan yang ditetapkan, sambil mencetak setiap bilangan. Loop berhenti ketika bilangan mencapai 1. Setiap bilangan dalam deret dicetak dalam satu baris, dipisahkan oleh spasi. Program ini mendemonstrasikan penggunaan prosedur, kondisional, dan loop dalam Go untuk menyelesaikan masalah matematika sederhana. Output program adalah deret bilangan yang dihasilkan, dimulai dari bilangan input hingga mencapai 1, sesuai dengan aturan yang ditetapkan dalam soal.