

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL IV
PROSEDUR



Disusun Oleh :

Tegar Aji Pangestu / 2311102021

IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi, S.Kom., M.Kom

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Prosedur (procedure) dalam pemrograman adalah blok kode yang dapat dipanggil untuk menjalankan satu atau beberapa perintah secara berulang kali. Prosedur memungkinkan pengembang untuk memecah program menjadi bagian-bagian kecil yang dapat digunakan kembali dan lebih mudah diatur. Dalam bahasa pemrograman seperti Golang, prosedur juga dikenal sebagai *function* atau *method* tergantung pada konteks penggunaannya.

Di Go, prosedur dikenal sebagai fungsi (*functions*). Fungsi di Go adalah bagian penting dari desain program karena Go adalah bahasa yang berbasis pada fungsi (*function-oriented*). Fungsi digunakan untuk mengeksekusi sejumlah perintah atau logika secara terpisah dan dapat dipanggil di berbagai bagian kode.

Tipe Data Fungsi: Golang mendukung fungsi dengan tipe data yang jelas. Parameter dan nilai balik (return value) harus dideklarasikan dengan tipe datanya.

Fungsi Tanpa Kembalian (Void): Jika sebuah fungsi tidak mengembalikan nilai, tipe kembalian tidak perlu dideklarasikan.

Multiple Return Values: Salah satu fitur yang menonjol dari Golang adalah kemampuannya untuk mengembalikan lebih dari satu nilai dari sebuah fungsi. Ini sangat berguna dalam berbagai skenario, misalnya ketika fungsi perlu mengembalikan hasil sekaligus error.

Prosedur atau fungsi dalam Go adalah komponen penting yang memungkinkan kode ditulis dengan cara yang modular, terstruktur, dan mudah dipelihara. Dengan fitur seperti *multiple return values*, *first-class functions*, serta mekanisme kontrol eksekusi seperti defer, Go memberikan fleksibilitas tinggi dalam membangun program yang efisien dan tangguh.

II. GUIDED

1. Guided I

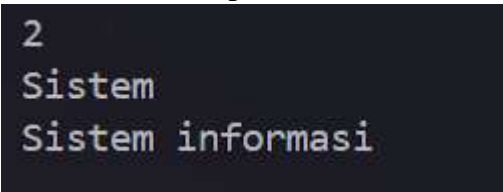
Sourcecode

```
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



```
2
Sistem
Sistem informasi
```

Deskripsi Program

bilangan: Variabel bertipe integer untuk menyimpan input angka. pesan: Variabel bertipe string untuk menyimpan input pesan dari pengguna. Program ini adalah contoh sederhana tentang bagaimana cara menerima input dari pengguna dan memprosesnya untuk menghasilkan output yang sesuai berdasarkan kondisi tertentu.

2. Guided II

Sourcecode

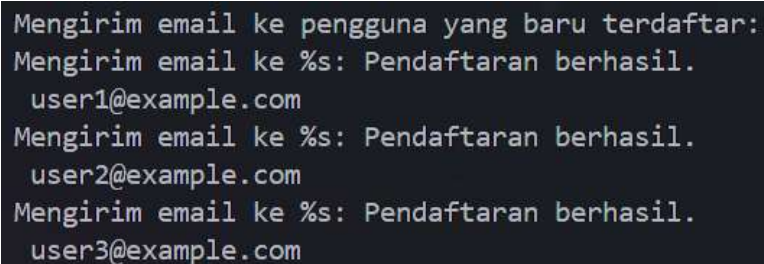
```
package main
import "fmt"

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

func main(){
    emails := []string{"user1@example.com",
"user2@example.com", "user3@example.com"}

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

Screenshoot Output



```
Mengirim email ke pengguna yang baru terdaftar:
Mengirim email ke %s: Pendaftaran berhasil.
user1@example.com
Mengirim email ke %s: Pendaftaran berhasil.
user2@example.com
Mengirim email ke %s: Pendaftaran berhasil.
user3@example.com
```

Deskripsi Program

emails: Sebuah slice (array dinamis) yang berisi daftar alamat email pengguna yang baru terdaftar. Untuk setiap email, fungsi `sendEmailNotification(email)` dipanggil, yang akan mengirimkan notifikasi kepada pengguna. Program ini adalah contoh sederhana dari pengiriman notifikasi melalui email kepada beberapa pengguna.

3. Guided III

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

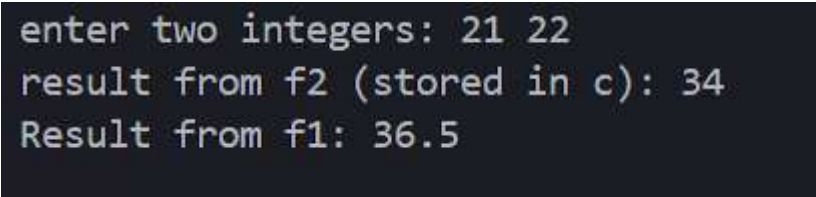
    fmt.Print("enter two integers: ")
    fmt.Scan(&a, &b)

    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



```
enter two integers: 21 22
result from f2 (stored in c): 34
Result from f1: 36.5
```

Deskripsi Program

Fungsi f1 ini menerima dua parameter bertipe integer (x dan y). Fungsi f2 ini menerima dua parameter integer (x dan y), serta satu parameter pointer (hasil) yang menunjuk ke variabel bertipe float64. Program ini menunjukkan bagaimana cara menggunakan fungsi di Golang untuk

melakukan perhitungan matematis sederhana dengan memanfaatkan parameter dan pointer untuk mengelola data antar fungsi.

III. UNGUIDED

Unguided 1

Minggu ini, mahasiswa Fakultas Informatika mendapatkan tugas dari mata kuliah matematika diskrit untuk mempelajari kombinasi dan permutasi. Jonas salah seorang mahasiswa, iseng untuk mengimplementasikannya ke dalam suatu program. Oleh karena itu bersediakah kalla membantu Jonas? (tidak tentunya ya :p) Masukan terdiri dari empat buah bilangan asli a , b , c , dan d yang dipisahkan oleh spasi, dengan syarat $a \geq c$ dan $b \geq d$ Keluaran terdiri dari dua baris. Baris pertama adalah hasil permutasi dan kombinasi a terhadap c , sedangkan baris kedua adalah hasil permutasi dan kombinasi b terhadap d . Catatan : permutasi (P) dan kombinasi (C) dari n terhadap r ($n \geq r$) dapat dihitung dengan menggunakan persamaan berikut! $P(n, r) = \frac{n!}{(n-r)!}$, sedangkan $C(n, r) = \frac{n!}{r!(n-r)!}$

Sourcecode

```
package main
import "fmt"
//Tegar Aji Pangestu
var a, b, c, d int

func faktorial(n int) int {
    hasil := 1
    for i := 1; i <= n; i++ {
        hasil = hasil * i
    }
    return hasil
}

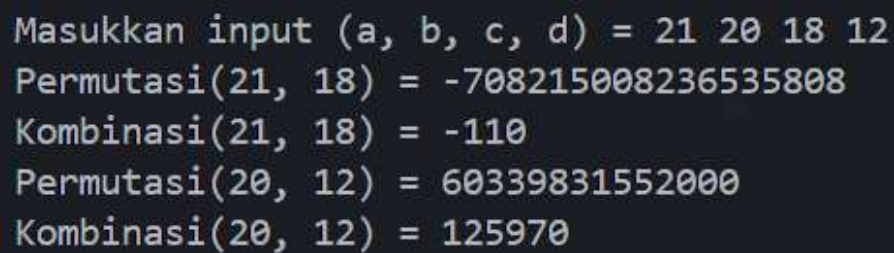
func hitungPermutasi(n, r int) {
    hasil := faktorial(n) / faktorial(n-r)
    fmt.Printf("Permutasi(%d, %d) = %d\n", n, r, hasil)
}

func hitungKombinasi(n, r int) {
    hasil := faktorial(n) / (faktorial(r) * faktorial(n-r))
    fmt.Printf("Kombinasi(%d, %d) = %d\n", n, r, hasil)
}

func main() {
    fmt.Print("Masukkan input (a, b, c, d) = ")
    fmt.Scan(&a, &b, &c, &d)
    if a >= c && b >= d {
```

```
        hitungPermutasi(a, c)
        hitungKombinasi(a, c)
        hitungPermutasi(b, d)
        hitungKombinasi(b, d)
    } else {
        fmt.Println("Syarat tidak terpenuhi: a harus >= c
dan b harus >= d")
    }
}
```

Screenshoot Output



```
Masukkan input (a, b, c, d) = 21 20 18 12
Permutasi(21, 18) = -708215008236535808
Kombinasi(21, 18) = -110
Permutasi(20, 12) = 60339831552000
Kombinasi(20, 12) = 125970
```

Deskripsi Program

Variabel a, b, c, dan d dideklarasikan di luar fungsi untuk menyimpan input dari pengguna. Program meminta pengguna untuk memasukkan empat bilangan bulat (a, b, c, dan d). Jika terpenuhi, program akan menghitung dan mencetak permutasi dan kombinasi untuk pasangan (a, c) dan (b, d). Program ini adalah implementasi sederhana dari perhitungan permutasi dan kombinasi dalam matematika. Ini menunjukkan bagaimana cara menggunakan fungsi untuk menghitung nilai matematis berdasarkan input pengguna serta melakukan validasi input sebelum melakukan perhitungan.

Unguided 2

Kompetisi pemrograman tingkat nasional berlangsung ketat. Setiap peserta diberikan 8 soal yang harus dapat diselesaikan dalam waktu 5 jam saja. Peserta yang berhasil menyelesaikan soal paling banyak dalam waktu paling singkat adalah pemenangnya. Buat program gema yang mencari pemenang dari daftar peserta yang diberikan. Program harus dibuat modular, yaitu

dengan membuat prosedur hitungSkor yang mengembalikan total soal dan total skor yang dikerjakan oleh seorang peserta, melalui parameter formal. Pembacaan nama peserta dilakukan di program utama, sedangkan waktu pengerjaan dibaca didalam prosedur. prosedur hitungSkor(in/out soal, skor : integer) Setiap baris masukan dimulai dengan satu string nama peserta tersebut diikuti dengan adalah 8 integer yang menyatakan berapa lama (dalam menit) peserta tersebut menyelesaikan soal. Jika tidak berhasil atau tidak mengirimkan jawaban maka otomatis dianggap menyelesaikan dalam waktu 5 jam 1 menit (301 menit). Satu baris keluaran berisi nama pemenang, jumlah soal yang diselesaikan, dan nilai yang diperoleh. Nilai adalah total waktu yang dibutuhkan untuk menyelesaikan soal yang berhasil diselesaikan.

Sourcecode

```
package main
import "fmt"

//Tegar Aji Pangestu

func hitungSkor(jumlah_soal_selesai * int, skor * int)
bool {
    const jumlah_soal int = 8
    var jumlah_soal_selesai_lokal int = jumlah_soal
    var t = [jumlah_soal] int {}
    skor_lokal := 0
    for i := 0; i<jumlah_soal; i++ {
        fmt.Scan(&t[i])
        if t[i] != 301 {
            skor_lokal += t[i]
        } else {
            jumlah_soal_selesai_lokal --
        }
    }
    if jumlah_soal_selesai_lokal > *jumlah_soal_selesai {
        *skor = skor_lokal
        *jumlah_soal_selesai = jumlah_soal_selesai_lokal
        return true
    } else if jumlah_soal_selesai_lokal ==
    *jumlah_soal_selesai {
        if skor_lokal > *skor {
            *skor = skor_lokal
            *jumlah_soal_selesai =
            jumlah_soal_selesai_lokal
        }
        return true
    }
```

```

    }
}
return false
}

func main() {
    var jumlah_soal_selesai, skor int
    var nama_pemenang, nama string
    for {
        fmt.Scan(&nama)
        if nama == "Selesai" { break }
        if hitungSkor(&jumlah_soal_selesai, &skor) {
            nama_pemenang = nama
        }
    }
    fmt.Print("\n")
    fmt.Println(nama_pemenang, jumlah_soal_selesai, skor)
}

```

Screenshoot Output

```

Astuti 20 50 301 301 61 71 75 10
Bertha 25 46 301 26 50 60 65 21
Selesai

Bertha 7 293

```

Deskripsi Program

Variabel `jumlah_soal_selesai` dan `skor` untuk menyimpan data tentang berapa banyak soal yang selesai dikerjakan dan berapa total skornya. Variabel `nama_pemenang` untuk menyimpan nama pemenang. Program ini adalah implementasi sederhana dari sistem penilaian kontes, yang menunjukkan bagaimana cara mengelola input pengguna dan menghitung hasil berdasarkan beberapa kriteria

Unguided 3

Skiena dan Revilla dalam *Programming Challenges* mendefinisikan sebuah deret bilangan. Deret dimulai dengan sebuah bilangan bulat n . Jika bilangan n saat itu genap, maka suku berikutnya adalah $\frac{1}{2}n$, tetapi jika ganjil maka

suku berikutnya bernilai $3n+1$. Rumus yang sama digunakan terus menerus untuk mencari suku berikutnya. Deret berakhir ketika suku terakhir bernilai 1. Sebagai contoh jika dimulai dengan $n=22$, maka deret bilangan yang diperoleh adalah: 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1 Untuk suku awal sampai dengan 1000000, diketahui deret ini selalu mencapai suku dengan nilai 1. Buat program skiena yang akan mencetak setiap suku dari deret yang dijelaskan diatas untuk nilai suku awal yang diberikan. Pencetakan deret harus dibuat dalam prosedur cetakDeret yang mempunyai 1 parameter formal, yaitu nilai dari suku awal. prosedur cetakDeret(in n : integer) Masukan berupa satu bilangan integer positif yang lebih kecil dari 1000000. Keluaran terdiri dari satu baris saja. Setiap suku dari deret tersebut dicetak dalam baris yang dan dipisahkan oleh sebuah spasi.

Sourcecode

```
package main
import "fmt"

//Tegar Aji Pangestu
func cetakDeret(n int) {
    fmt.Print(n, " ")
    prev := n
    current := n
    for i:=0; i<n; i++ {
        if prev % 2 == 0 {
            current = prev / 2

        } else {
            current = 3*prev + 1
        }
        prev = current
        fmt.Print(current, " ")
        if current == 1 { break }
    }
}

func main() {
    var n int
    fmt.Print("n: ")
    fmt.Scan(&n)
    if n > 1000000 {
        fmt.Println("Input tidak boleh melebihi 1000000")
        return
    }
    cetakDeret(n)
```

```
}
```

Screenshoot Output

```
n: 21  
21 64 32 16 8 4 2 1
```

Deskripsi Program

Variabel prev dan current diinisialisasi dengan nilai n. Variabel prev menyimpan nilai sebelumnya, sedangkan current akan menyimpan nilai saat ini yang dihitung. Program ini adalah implementasi sederhana dari algoritma Collatz, yang menunjukkan bagaimana bilangan bulat positif dapat dikendalikan dengan aturan tertentu hingga mencapai angka satu.