

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL IV
PROSEDUR



Disusun Oleh :

PETRA PRIADI S.P GINTING (2311102273)

IF-06

Dosen Pengampu :

ABEDNEGO DWI SEPTIADI

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Pengertian Prosedur dalam Bahasa Go

Prosedur adalah serangkaian instruksi yang dirancang untuk melakukan tugas tertentu, di mana tugas tersebut tidak membutuhkan pengembalian nilai. Dalam bahasa pemrograman Go, prosedur biasanya ditulis sebagai fungsi, tetapi dengan ciri khas bahwa mereka tidak menggunakan pernyataan return untuk mengembalikan nilai kepada pemanggil.

Karakteristik Utama Prosedur

1. Tidak mengembalikan nilai (no return value):

- Prosedur fokus pada efek samping (side effects) seperti mencetak ke konsol atau memodifikasi variabel global, tanpa mengembalikan hasil kalkulasi atau nilai apapun ke pemanggil.

2. Parameter (Parameter):

- Prosedur bisa menerima parameter untuk menyesuaikan perilaku atau tindakan yang mereka lakukan, seperti menerima data atau nilai yang diperlukan untuk melaksanakan tugasnya.

3. Efek Samping (Side Effects):

- Efek samping adalah perubahan atau hasil yang dipengaruhi oleh prosedur di luar lingkup atau batasan prosedur itu sendiri. Contohnya, mencetak output, mengubah status variabel global, atau melakukan tindakan lain yang mempengaruhi bagian lain dari program.

Fungsi Prosedur dalam Pengembangan Perangkat Lunak

- **Modularitas (Modularitas):**

- Dengan memecah kode menjadi prosedur-prosedur kecil, pengembang dapat meningkatkan keterbacaan dan pemeliharaan kode. Ini memungkinkan pengembang untuk memahami dan mengelola setiap bagian dari program secara terpisah, tanpa perlu mengubah keseluruhan struktur program.

- **Pemeliharaan dan Debugging (Maintainability and Debugging):**

- Prosedur mempermudah proses debugging dengan memisahkan tanggung jawab ke bagian-

bagian yang lebih kecil dan spesifik. Jika ada masalah, pengembang bisa fokus pada prosedur tertentu tanpa harus menyisir seluruh kode.

- **Pengelolaan Efek Samping (Managing Side Effects):**
 - Prosedur memungkinkan pengelolaan efek samping secara lebih terkendali. Karena prosedur dapat memodifikasi variabel global atau menghasilkan output, pengembang bisa lebih mudah mengatur dan melacak perubahan yang dilakukan oleh prosedur tersebut.

Keterbatasan Prosedur

1. Kurangnya Modularitas yang Fleksibel (Limited Flexibility in Modularity):

- Karena prosedur tidak mengembalikan nilai, fleksibilitas dalam memanipulasi atau menggunakan hasil dari prosedur tersebut menjadi terbatas. Ini bisa mempersulit penggabungan hasil dari berbagai operasi atau pengolahan data lebih lanjut.

2. Kesulitan dalam Pengujian (Testing Challenges):

- Prosedur bisa lebih sulit untuk diuji secara unit (unit testing) karena ketergantungannya pada efek samping. Pengujian memerlukan kontrol yang lebih ketat atas lingkungan dan kondisi tempat prosedur tersebut dijalankan.

Implementasi Prosedur dalam Proyek Nyata

Dalam aplikasi dunia nyata, prosedur sering digunakan dalam konteks di mana interaksi langsung dengan pengguna atau lingkungan diperlukan. Ini termasuk aplikasi antarmuka pengguna grafis (GUI), aplikasi berbasis jaringan, atau sistem tertanam (embedded systems). Prosedur memungkinkan pengembang untuk menulis kode yang responsif terhadap peristiwa atau input pengguna tanpa memerlukan mekanisme pengembalian nilai yang kompleks.

Kesimpulan

Prosedur adalah alat yang kuat dalam pengembangan perangkat lunak, terutama dalam konteks di mana efek samping dan operasi langsung diperlukan. Meskipun mereka memiliki beberapa keterbatasan terkait fleksibilitas dan pengujian, pemahaman yang tepat tentang penggunaan prosedur dapat membantu pengembang menciptakan kode yang lebih modular, mudah dipelihara, dan efisien.

II. GUIDED

1. Sourcecode

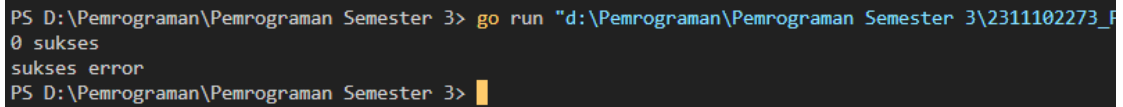
```
package main

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 D:\Pemrograman\Pemrograman Semester 3> go run "d:\Pemrograman\Pemrograman Semester 3\2311102273_F
0 sukses
sukses error
PS D:\Pemrograman\Pemrograman Semester 3> █
```

```
PS D:\Pemrograman\Pemrograman Semester 3> go run "d:\Pemrograman\Pemrograman Semester 3\2311102273_Petra Priadi S.P Ginting\
1 sukses
sukses warning
PS D:\Pemrograman\Pemrograman Semester 3>
```

```
PS D:\Pemrograman\Pemrograman Semester 3> go run "d:\Pemrograman\Pemrograman Semester 3\2311102273_Petra Priadi S.P
2 sukses
sukses informasi
PS D:\Pemrograman\Pemrograman Semester 3>
```

Deskripsi Program

Kode ini adalah program Go yang meminta input dari pengguna berupa sebuah bilangan bulat dan sebuah pesan, kemudian mencetak pesan tersebut bersama dengan jenis pesannya. Program dimulai dengan fungsi `main()` yang menggunakan `fmt.Scan()` untuk membaca input bilangan dan pesan dari pengguna. Selanjutnya, fungsi `cetakPesan()` dipanggil dengan argumen pesan dan bilangan yang telah diinput. Fungsi `cetakPesan()` menentukan jenis pesan berdasarkan nilai flag (bilangan) yang diterima: 0 untuk "error", 1 untuk "warning", dan 2 untuk "informasi". Jika nilai flag tidak sesuai dengan ketiga kondisi tersebut, jenis pesan akan berupa string kosong. Akhirnya, fungsi mencetak pesan bersama dengan jenis pesannya menggunakan `fmt.Println()`. Program ini mendemonstrasikan penggunaan fungsi, variabel, percabangan kondisional, dan input/output dasar dalam bahasa Go.

2. 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

```

PS D:\Pemrograman\Pemrograman Semester 3> go run "d:\Pemrograman\Pemrograman Semester 3\2311102273_Petra Priadi S.P Ginting\MODUL 4\
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
PS D:\Pemrograman\Pemrograman Semester 3>

```

Deskripsi Program

Fungsi `sendEmailNotification` didefinisikan untuk mencetak pesan simulasi pengiriman email ke alamat yang diberikan. Dalam fungsi `main`, sebuah slice `emails` dibuat berisi tiga alamat email contoh. Program kemudian mencetak pesan awal menggunakan `fmt.Println`. Selanjutnya, program menggunakan loop `for` dengan `range` untuk mengiterasi melalui setiap alamat email dalam slice. Pada setiap iterasi, fungsi `sendEmailNotification` dipanggil dengan alamat email saat ini sebagai argumen, mensimulasikan pengiriman notifikasi ke masing-masing pengguna. Kode ini mendemonstrasikan penggunaan slice, perulangan, fungsi, dan string formatting dalam Go untuk mensimulasikan proses pengiriman email otomatis ke beberapa penerima.

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) { //pass by preference
        *hasil = float64(2*x) - 0.5*float64(y) + 3.0
    }

    func main() {
        var x, y int
        var z float64

        fmt.Print("Enter two intergers : ")

        fmt.Scan(&x, &y)

        f2(x, y, &z)

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

        resultF1 := f1(y, x)

        fmt.Println("Result from f1 : ", resultF1)
    }

```

Screenshoot Output

```

PS D:\Pemrograman\Pemrograman Semester 3> go run "d:\Pemrograman\Pemrograman Semester 3\2311102273_Petra Priadi S.P Ginting\MODUL 4
enter two integers: 1 2
result from f2 (stored in c): 4
Result from f1: 6.5
PS D:\Pemrograman\Pemrograman Semester 3>

```

Deskripsi Program

Program dimulai dengan mendefinisikan dua fungsi: f1 dan f2. Fungsi f1 menerima dua parameter integer x dan y, melakukan perhitungan $2x - 0.5y + 3.0$, dan mengembalikan hasilnya sebagai float64. Fungsi f2 melakukan perhitungan yang sama, tetapi menggunakan pointer untuk menyimpan hasilnya. Dalam fungsi main, program meminta pengguna memasukkan dua bilangan bulat menggunakan fmt.Scan. Kemudian, f2 dipanggil dengan argumen a, b, dan alamat variabel c, menyimpan hasilnya langsung ke c. Hasil ini dicetak. Selanjutnya, f1 dipanggil dengan argumen b dan a (dibalik urutannya), dan hasilnya disimpan dalam resultF1 sebelum dicetak. Program ini mengilustrasikan perbedaan antara fungsi yang mengembalikan nilai (f1) dan fungsi yang menggunakan pointer untuk menyimpan hasil (f2).

III. UNGUIDED

1. Soal Studi Case

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.

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.

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 factN, factR, factNR int
    factorial(n, &factN)
    factorial(r, &factR)
    factorial(n-r, &factNR)
    *hasil = factN / (factR * factNR)
}

func main() {
    var a, b, c, d int
    fmt.Scan(&a, &b, &c, &d)

    var permA, combA, permB, combB int

    permutation(a, c, &permA)
    combination(a, c, &combA)
```

```

    permutation(b, d, &permB)
    combination(b, d, &combB)

    fmt.Printf("%d %d\n", permA, combA)
    fmt.Printf("%d %d\n", permB, combB)
}

```

Screenshoot Output

```

PS D:\Pemrograman\Pemrograman Semester 3> go run "d:\Pemrograman\Pemrograman Semester 3\2311102273_Petra Priadi S.P Ginting\W
5 10 3 10
60 10
3628800 1
PS D:\Pemrograman\Pemrograman Semester 3>

```

Deskripsi Program

Fungsi factorial menghitung faktorial sebuah angka menggunakan loop dan pointer untuk menyimpan hasilnya. Fungsi permutation menghitung permutasi nPr menggunakan rumus $n! / (n-r)!$, memanfaatkan fungsi factorial. Fungsi combination menghitung kombinasi nCr dengan rumus $n! / (r! * (n-r)!)$, juga menggunakan factorial. Dalam fungsi main, program meminta input empat bilangan bulat (a, b, c, d) dari pengguna. Kemudian, program menghitung permutasi dan kombinasi untuk pasangan (a,c) dan (b,d) menggunakan fungsi yang telah didefinisikan. Hasil perhitungan disimpan dalam variabel terpisah (permA, combA, permB, combB) menggunakan pointer. Akhirnya, program mencetak hasil perhitungan dalam dua baris, masing-masing berisi hasil permutasi dan kombinasi untuk setiap pasangan angka.

2. Studi Case

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 di dalam prosedur.

prosedure 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). A

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.

Keterangan:

Astuti menyelesaikan 6 soal dalam waktu 287 menit, sedangkan Bertha 7 soal dalam waktu 294 menit. Karena Bertha menyelesaikan lebih banyak, maka Bertha menang. Jika keduanya menyelesaikan sama banyak, maka pemenang adalah yang menyelesaikan dengan total waktu paling kecil.

Sourcecode

```
package main

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

func hitungSkor(waktu []int, soal *int, skor *int) {
    *soal = 0
    *skor = 0
    for _, t := range waktu {
        if t <= 300 {
            *soal++
            *skor += t
        }
    }
}
```

```

func bacaInput(scanner *bufio.Scanner) (string, []int) {

    line := scanner.Text()

    fields := strings.Fields(line)

    nama := fields[0]

    waktu := make([]int, 8)

    for i := 0; i < 8; i++ {

        waktu[i], _ = strconv.Atoi(fields[i+1])

    }

    return nama, waktu

}

func updatePemenang(nama string, soal int, skor int, pemenang *string,
maxSoal *int, minSkor *int) {

    if soal > *maxSoal || (soal == *maxSoal && skor < *minSkor) {

        *pemenang = nama

        *maxSoal = soal

        *minSkor = skor

    }

}

func main() {

    scanner := bufio.NewScanner(os.Stdin)

    var pemenang string

    var maxSoal, minSkor int

    minSkor = 9999

    for scanner.Scan() {

```

```

        if scanner.Text() == "Selesai" {

            break

        }

        nama, waktu := bacaInput(scanner)

        var soal, skor int

        hitungSkor(waktu, &soal, &skor)

        updatePemenang(nama, soal, skor, &pemenang, &maxSoal,
&minSkor)

    }

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

}

```

Screenshoot Output

```

PS D:\Pemrograman\Pemrograman Semester 3> go run "d:\Pemrograman\Pemrograman
Semester 3\2311102273_Petra Priadi S.P Ginting\MODUL 4\Unguided2_modul4.go"
Astuti 20 50 301 301 61 71 75 10
Bertha 25 47 301 26 50 60 65 21
Selesai
Bertha 7 294
PS D:\Pemrograman\Pemrograman Semester 3>

```

Deskripsi Program

Program menggunakan beberapa fungsi utama: `hitungSkor` menghitung jumlah soal yang diselesaikan dan total skor berdasarkan waktu penyelesaian, `bacaInput` membaca dan memproses input dari pengguna, dan `updatePemenang` memperbarui informasi pemenang. Fungsi main menggunakan `bufio.Scanner` untuk membaca input berulang kali sampai kata "Selesai" dimasukkan. Setiap baris input berisi nama peserta dan 8 waktu penyelesaian soal. Program memproses setiap input, menghitung skor, dan memperbarui informasi pemenang. Pemenang ditentukan berdasarkan jumlah soal terbanyak yang diselesaikan dalam waktu 300 detik atau kurang, dengan total waktu terendah sebagai penentu jika ada yang seri. Program menggunakan pointer secara ekstensif untuk

memodifikasi variabel di dalam fungsi. Akhirnya, program mencetak nama pemenang, jumlah soal yang diselesaikan, dan total skor

3. Studi Case

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 selalu mencapai suku dengan nilai 1.

Buat program sklena yang akan mencetak setiap suku dari deret yang dijelaskan di atas untuk nilai suku awal yang diberikan. Pencetakkan deret harus dibuat dalam prosedur cetak Deret yang mempunyai 1 parameter formal, yaitu nilai dari suku awal.

prosedure cetakDeret(inn 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"
    "strconv"
    "strings"
)

func cetakDeret(n int) string {
    var result []string
    result = append(result, strconv.Itoa(n))

    for n != 1 {
        if n%2 == 0 {
            n = n / 2
        } else {
            n = 3*n + 1
        }
        result = append(result, strconv.Itoa(n))
    }

    return strings.Join(result, " ")
}
```

```

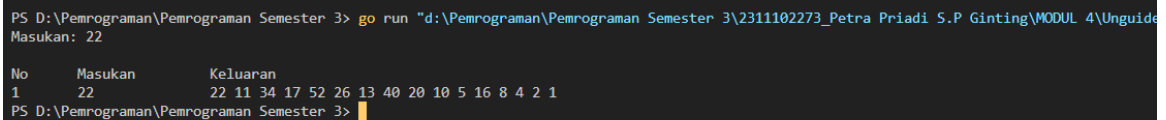
func main() {
    var input int
    fmt.Print("Masukan: ")
    fmt.Scan(&input)

    if input <= 0 || input >= 1000000 {
        fmt.Println("Masukan harus berupa bilangan integer positif
yang lebih kecil dari 1000000.")
        return
    }

    fmt.Println("\nNo\tMasukan\t\tKeluaran")
    fmt.Printf("1\t%d\t\t%s\n", input, cetakDeret(input))
}

```

Screenshoot Output



```

PS D:\Pemrograman\Pemrograman Semester 3> go run "d:\Pemrograman\Pemrograman Semester 3\2311102273_Petra Priadi S.P Ginting\MODUL 4\Unguide"
Masukan: 22
No      Masukan      Keluaran
1       22           22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS D:\Pemrograman\Pemrograman Semester 3>

```

Deskripsi Program

Program dimulai dengan mendefinisikan fungsi cetakDeret yang menerima sebuah bilangan bulat n dan mengembalikan string berisi deret Collatz. Fungsi ini menggunakan slice untuk menyimpan setiap angka dalam deret, dimulai dengan n. Kemudian, fungsi menerapkan aturan Collatz: jika n genap, bagi dengan 2; jika ganjil, kalikan dengan 3 dan tambah 1. Proses ini berlanjut hingga n mencapai 1. Setiap angka dikonversi ke string menggunakan strconv.Itoa dan ditambahkan ke slice. Akhirnya, slice digabungkan menjadi satu string dengan spasi sebagai pemisah. Dalam fungsi main, program meminta input dari pengguna, memeriksa apakah input valid (positif dan kurang dari 1.000.000), lalu memanggil cetakDeret untuk menghasilkan deret. Hasilnya dicetak dalam format tabel sederhana. Program ini mendemonstrasikan penggunaan fungsi, perulangan, percabangan, slice, konversi tipe data, dan manipulasi string dalam Go untuk menyelesaikan masalah matematika yang menarik.