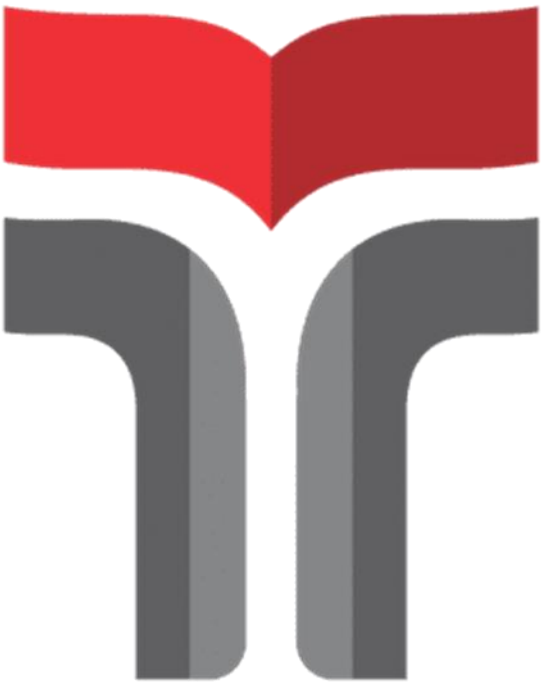
LAPORAN PRAKTIKUM
ALGORITMA PEMROGRAMAN 2

MODUL 4

PROSEDUR



Oleh:

TSAQIF KANZ AHMAD
2311102075
IF-11-02

S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2024

I. DASAR TEORI

A. Definisi Prosedur

Prosedur dapat dianggap sebagai potongan beberapa instruksi program menjadi suatu instruksi baru yang dibuat untuk mengurangi kerumitan dari kode program yang kompleks pada suatu program yang besar. Prosedur akan menghasilkan suatu akibat atau efek langsung pada program ketika dipanggil pada program utama. Suatu subprogram dikatakan prosedur apabila:

1. Tidak ada deklarasi tipe nilai yang dikembalikan, dan
2. Tidak terdapat kata kunci return dalam badan subprogram.

Kedudukannya prosedur sama seperti instruksi dasar yang sudah ada sebelumnya (assignment) danlatau instruksi yang berasal dari paket (fmt), seperti fmt.Scan dan fmt.Print. Karena itu selalu pilih nama prosedur yang berbentuk kata kerJa atau sesuatu yang merepresentasikan proses sebagai nama dari prosedur. Contoh: **cetak, hitungRerata, cariNilai, belok, mulai,...**

B. Dekalarasi Prosedur

Prosedur secara teknis adalah fungsi yang tidak mengembalikan nilai. Untuk mendeklarasikan prosedur, kita cukup mendefinisikan fungsi tanpa tipe kembalian. Deklarasi ini menggunakan kata kunci func diikuti oleh nama fungsi, parameter (jika ada), dan tanpa menyebutkan tipe kembalian.

```
func namaFungsi(parameter1 tipeData1, parameter2 tipeData2, ...)
tipeDataHasil {
    // Blok kode fungsi
    return nilaiHasil
}
```

- **func:** Kata kunci untuk mendeklarasikan fungsi.
- **namaFungsi:** Nama yang diberikan untuk fungsi, harus unik dalam scope.
- **parameter:** Variabel yang menerima nilai saat fungsi dipanggil.
- **tipeData:** Tipe data dari parameter atau nilai kembalian.
- **return:** Kata kunci untuk mengembalikan nilai dari fungsi.

Contoh :

```
func hitungLuasPersegiPanjang(panjang, lebar float64) float64 {
    luas := panjang * lebar
    return luas
}
```

Fungsi di atas menerima dua parameter bertipe float64 (panjang dan lebar) dan mengembalikan nilai luas persegi panjang yang dihitung.

C. Karakteristik dan Tujuan Penggunaan Prosedur

* Karakteristik :

- **Tidak mengembalikan nilai:** Prosedur hanya menjalankan tugas, dan tidak ada nilai yang dikembalikan.

- **Parameter opsional:** Seperti fungsi, prosedur dapat menerima parameter untuk menjalankan tugas tertentu, tetapi tidak diharuskan untuk memiliki parameter.
- **Efek samping:** Prosedur sering kali dimanfaatkan untuk memodifikasi variabel global atau memanggil fungsi-fungsi lain.

* Tujuan Penggunaan Prosedur :

- **Modularitas:** Membagi program menjadi bagian-bagian yang lebih kecil dan lebih mudah dikelola.
- **Reusabilitas:** Kode dalam suatu fungsi dapat digunakan berulang kali di berbagai bagian program.
- **Abstraksi:** Menyembunyikan detail implementasi dari pengguna fungsi.
- **Peningkatan keterbacaan:** Membuat kode program lebih mudah dibaca dan dipahami.

D. Cara Pemanggilan Prosedur

Pemanggilan prosedur dilakukan dengan menuliskan nama prosedur diikuti oleh tanda kurung () beserta argumen (jika ada). Karena prosedur dalam Go hanyalah fungsi tanpa nilai kembalian, cara pemanggilannya sama seperti memanggil fungsi biasa.

1. **Prosedur tanpa parameter:** Jika prosedur tidak menerima parameter, cukup memanggilnya dengan menuliskan nama prosedur diikuti dengan tanda kurung kosong ().

```
func cetakHello() {
      fmt.Println("Hello, World!")
}

func main() {
       cetakHello()  // Memanggil prosedur cetakHello
}
```

2. **Prosedur dengan parameter:** Jika prosedur memiliki parameter, maka harus memanggilnya dengan mengirimkan nilai yang sesuai dengan tipe parameter yang dideklarasikan.

```
func cetakPesan(pesan string) {
            fmt.Println(pesan)
}

func main() {
   cetakPesan("Selamat datang di Golang!")  // Memanggil prosedur
dengan argumen
}
```

3. **Prosedur dengan beberapa parameter:** Jika prosedur menerima beberapa parameter, maka harus memanggilnya dengan mengirimkan nilai yang sesuai dengan urutan dan tipe parameternya.

```
func cetakIdentitas(nama string, umur int) {
       fmt.Printf("Nama: %s, Umur: %d\n", nama, umur)
```

```
}

func main() {
      cetakIdentitas("Andi", 25)  // Memanggil prosedur dengan dua
argumen
}
```

4. Aturan dalam Pemanggilan Prosedur:

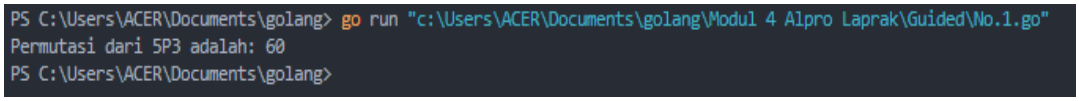
- **Urutan Argumen:** Saat memanggil prosedur, pastikan urutan argumen sesuai dengan urutan parameter yang dideklarasikan dalam prosedur.
- **Tipe Data yang Cocok:** Tipe argumen yang dikirimkan harus cocok dengan tipe parameter prosedur.

II. GUIDED

1) SOURCE CODE

```go
package main

import "fmt"

// Fungsi untuk menghitung faktorial
       func factorial(n int) int {
    if n == 0 {
        return 1
    }
    result := 1
    for i := 1; i <= n; i++ {
        result *= i
    }
    return result
}

// Prosedur untuk menghitung dan menampilkan permutasi
func permutasi(n, r int) {
    hasilPermutasi := factorial(n) / factorial(n-r)
    fmt.Printf("Permutasi dari %dP%d adalah: %d\n", n, r,
hasilPermutasi)
}

func main() {
    // Memanggil prosedur untuk menghitung dan menampilkan
permutasi
  n, r := 5, 3
    permutasi(n, r)
}
```

SCREENSHOT OUTPUT :



PENJELASAN PROGRAM :

Program tersebut adalah program untuk menghitung dan menampilkan hasil permutasi dari dua bilangan n dan r menggunakan factorial. Pada program tersebut terdapat fungsi factorial untuk menghitung faktorial dari bilangan n. Faktorial n! dihitung dengan mengalikan semua bilangan dari 1 hingga n. Jika n adalah 0, faktorial bernilai 1 (karena 0! = 1). Kemudian terdapat prosedur permutasi untuk menghitung permutasi dan memanggil fungsi faktorial untuk menghitung faktorial n dan faktorial (n-r), lalu hasil permutasi dicetak dilayar. Pada fungsi utama nilai n dan r diberikan secara langsung yaitu n = 5 dan r = 3. Prosedur permutasi dipanggil untuk menghitung permutasi dari 5P3 dan hasilnya dicetak.

III. UNGUIDED

1). Buat program skiena yang akan mencetak setiap suku dari deret yang dijelaskan di atas untuk nilai suku awal yang diberikan. Pencetakan deret harus dibuat dalam prosedur cetakDeret yang mempunyai 1 parameter formal, yaitu nilai dari suku awal.

```
prosedure 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.

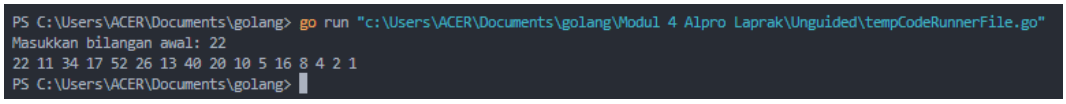
No	Masukan	Keluaran
1	22	22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

SOURCE CODE

```go
package main

import "fmt"

// Fungsi untuk mencetak deret Collatz
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(n) // Cetak angka 1 di akhir
}

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

    // Validasi input
    if n > 0 && n < 1000000 {
        cetakDeret(n)
    } else {
        fmt.Println("Input harus bilangan positif dan kurang
dari 1000000.")
```

```
    }
}
```

SCREENSHOT OUTPUT



PENJELASAN PROGRAM

*Program tersebut adalah program yang mencetak deret berdasarkan aturan Collatz conjecture, yang sering disebut juga "3n + 1 problem". Deret ini dimulai dari bilangan positif n dan akan terus berlanjut hingga mencapai angka 1. Pada program tersebut terdapat fungsi cetakDeret untuk mencetak deret Collatz dari bilangan n. Pada fungsi tersebut proses perulangannya Jika n genap, dibagi 2 (n = n / 2). Jika n ganjil, kalikan 3 lalu tambahkan 1 (n = 3*n + 1). Perulangan berhenti ketika nilai n menjadi 1, dan angka 1 dicetak di akhir. Pada fungsi utama Program meminta input bilangan n dari pengguna. Kemudian melakukan validasi agar n merupakan bilangan positif dan kurang dari 1 juta. Jika valid, program memanggil fungsi cetakDeret untuk mencetak deret. Jika tidak valid, pesan kesalahan akan ditampilkan.*