

**LAPORAN PRAKTIKUM**  
**ALGORITMA DAN PEMROGRAMAN 2**  
**MODUL IV**  
**PROSEDUR**



**Disusun Oleh :**

**Maulana Ghani Rolanda / 2311102012**

**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 adalah blok kode yang dikelompokkan untuk menjalankan satu atau lebih instruksi yang telah ditentukan. Prosedur bertindak sebagai unit yang dapat dipanggil secara independen dan berfungsi untuk mengurangi pengulangan kode, meningkatkan keterbacaan, dan mempermudah pemeliharaan program. Dalam bahasa pemrograman modern, prosedur dikenal juga sebagai fungsi (*functions*) atau subrutin. Golang, yang dikenal sebagai bahasa yang efisien dan sederhana, menggunakan konsep fungsi sebagai mekanisme untuk menerapkan prosedur. Fungsi di Golang adalah blok kode yang menerima input, menjalankan logika tertentu, dan menghasilkan output (bila diperlukan). Fungsi dalam Go dideklarasikan dengan kata kunci `func`, diikuti dengan nama fungsi, parameter (jika ada), dan tipe pengembalian. Salah satu keunggulan Golang adalah kinerjanya yang cepat dan efisien, termasuk dalam penggunaan fungsi. Fungsi dalam Go dieksekusi dengan overhead minimal, sehingga efisiensi waktu eksekusi dapat tetap terjaga. Selain itu, Go mendukung *inline functions* pada saat kompilasi untuk lebih meningkatkan performa. Prosedur dalam Golang, yang diimplementasikan melalui fungsi, menawarkan pendekatan modular, mudah diatur, dan fleksibel dalam pengembangan program. Dengan dukungan terhadap fitur-fitur seperti *multiple return values*, *anonymous functions*, dan *higher-order functions*, Golang memungkinkan pengembang untuk menulis kode yang bersih, efisien, dan mudah dikelola. Selain itu, penggunaan mekanisme kontrol seperti `defer`, `panic`, dan `recover` menambah kemampuan Go dalam menangani kesalahan dengan lebih aman.

## II. GUIDED

### 1. Guided 1

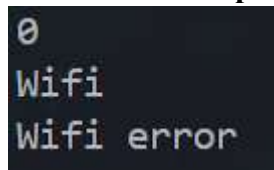
#### Sourcecode

```
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



```
0
Wifi
Wifi error
```

#### Deskripsi Program

Di dalam fungsi main, ada dua variabel yang dideklarasikan: bilangan bertipe int (integer) dan pesan bertipe string. Program meminta input dari pengguna menggunakan `fmt.Scan`, yang akan mengambil dua input dari keyboard: satu angka dan satu string. Misalnya, pengguna bisa mengetik sesuatu seperti 1 "Pesan contoh". Setelah input diterima, program memanggil fungsi `cetakPesan` dengan dua argumen, yaitu pesan dan bilangan. Jadi, pesan berisi teks yang dimasukkan oleh pengguna, dan bilangan adalah angka yang dimasukkan.

## 2. Guided 2

### Sourcecode

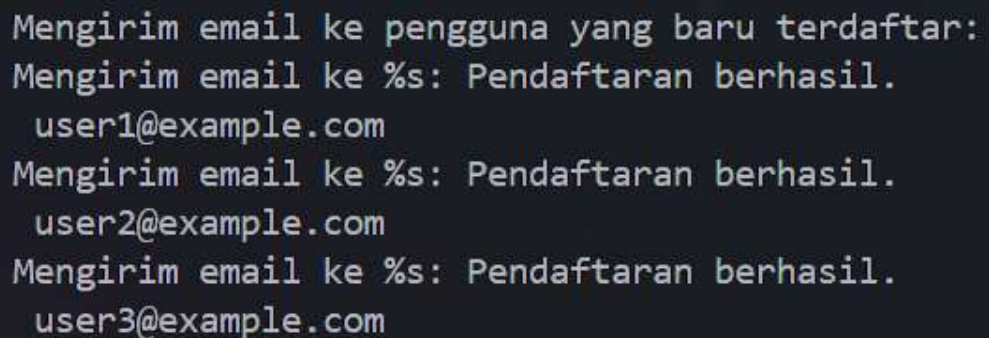
```
package main
import "fmt"

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



```
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

Fungsi ini menerima satu parameter berupa email yang bertipe string. Di dalam fungsi, seharusnya `fmt.Println` digunakan untuk mencetak pesan. Namun, ada kesalahan di sini: `%s` tidak akan berfungsi di dalam `fmt.Println` karena itu adalah format spesifik untuk `Printf`. Fungsi ini dimaksudkan untuk menampilkan pesan bahwa email notifikasi sedang dikirim, dengan memasukkan email pengguna ke dalam pesan.

### 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

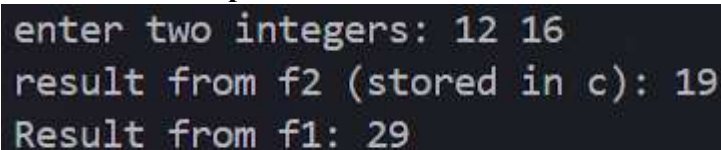
    fmt.Print("enter two integers: ")
    fmt.Scan(&a, &b)

    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: 12 16
result from f2 (stored in c): 19
Result from f1: 29
```

#### Deskripsi Program

Program ini menunjukkan dua cara yang berbeda untuk mengelola hasil perhitungan di Go: Dengan return value (f1): Fungsi mengembalikan nilai dan disimpan ke dalam variabel di luar fungsi. Dengan pointer (f2): Fungsi mengubah nilai variabel secara langsung menggunakan pointer,

memungkinkan hasil perhitungan disimpan langsung ke variabel yang ada di luar 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"
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)
}

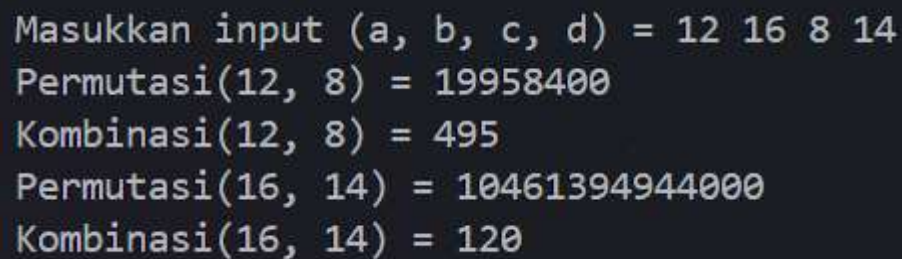
func hitungKombinasi(n, r int) {
    hasil := faktorial(n) / (faktorial(r) * faktorial(n-r))
    fmt.Printf("Kombinasi(%d, %d) = %d\n", n, r, hasil)
}

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")
    }
}

//Maulana Ghani Rolanda 2311102012
```

### Screenshoot Output



```
Masukkan input (a, b, c, d) = 12 16 8 14
Permutasi(12, 8) = 19958400
Kombinasi(12, 8) = 495
Permutasi(16, 14) = 10461394944000
Kombinasi(16, 14) = 120
```

### Deskripsi Program

Variabel a, b, c, dan d dideklarasikan secara global dengan tipe int. Variabel-variabel ini akan diisi dengan input dari pengguna untuk digunakan dalam perhitungan permutasi dan kombinasi. Fungsi ini juga menggunakan faktorial untuk menghitung faktorial dari n, r, dan n-r, lalu membagi hasil sesuai dengan rumus kombinasi. Hasil perhitungan kombinasi kemudian dicetak menggunakan `fmt.Printf`.

### 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"
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
}
```

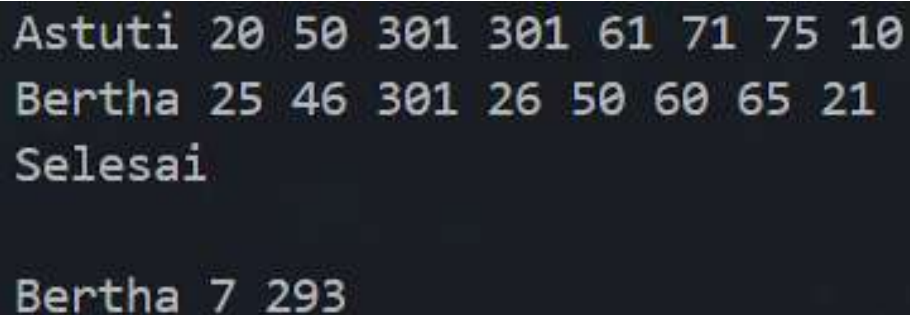
```

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)
}

//Maulana Ghani Rolanda 2311102012

```

### 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

Jumlah\_soal adalah konstanta yang berisi jumlah total soal, di sini ditetapkan 8 soal. jumlah\_soal\_selesai\_lokal adalah salinan lokal dari jumlah\_soal yang digunakan untuk menghitung berapa banyak soal yang selesai oleh peserta. skor\_lokal adalah variabel yang menyimpan nilai skor total untuk setiap peserta.

### 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

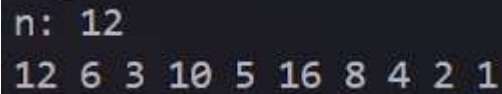
```
package main
import "fmt"
func cetakDeret(n int) {
    fmt.Print(n, " ")
    prev := n
    current := n
    for i:=0; i<n; i++ {
        if prev % 2 == 0 {
            current = prev / 2

        } 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)
}
```

```
//Maulana Ghani Rolanda 2311102012
```

### Screenshoot Output



```
n: 12  
12 6 3 10 5 16 8 4 2 1
```

### Deskripsi Program

Deret angka dimulai dengan angka  $n$  dan dicetak terlebih dahulu (`fmt.Print(n, " ")`). Loop berjalan hingga angka mencapai 1 atau hingga sebanyak  $n$  iterasi (meskipun biasanya deret akan berakhir lebih cepat). Setiap iterasi menghitung angka selanjutnya berdasarkan aturan: Jika angka sebelumnya (`prev`) adalah bilangan genap, maka angka selanjutnya adalah setengah dari angka tersebut (`current = prev / 2`). Jika angka sebelumnya adalah bilangan ganjil, maka angka selanjutnya adalah  $3 * prev + 1$ . Setelah menghitung angka selanjutnya, angka tersebut dicetak, dan loop akan berakhir jika angka `current` mencapai 1 (`if current == 1 { break }`).