

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL IV
PROSEDUR**



Disusun Oleh :

SYAHRUL ROMADHONI / 2311102261

S1 IF-11-05

Dosen Pengampu :

Arif Amrulloh, S.Kom.,M.Kom.

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Definisi Fungsi dalam Golang

Fungsi dalam Golang adalah blok kode yang dapat dipanggil ulang untuk melakukan tugas tertentu. Setiap fungsi memiliki nama, parameter, dan return value. Fungsi sangat penting dalam struktur logika program karena mereka memperbolehkan modifikasi dan reuse kode, sehingga meningkatkan efisiensi dan skalabilitas suatu proyek.

Keuntungan Menggunakan Fungsi di Golang:

- **Reusability (Dapat digunakan ulang):** Fungsi membantu mengurangi duplikasi kode.
- **Abstraksi:** Fungsi memungkinkan abstraksi yang memudahkan developer fokus pada tugas-tugas spesifik.
- **Modularitas:** Fungsi membantu memecah program besar menjadi unit-unit kecil yang lebih mudah dikelola.
- **Testing:** Fungsi yang terisolasi lebih mudah diuji dan divalidasi.

II. GUIDED

Soal Studi Case

1.

SOURCECODE

```
package main

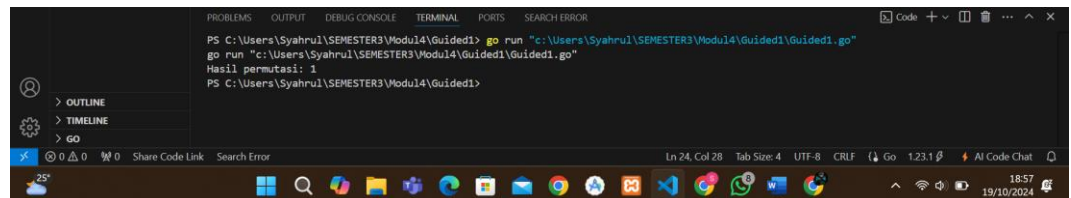
import "fmt"

func main() {
    var a, b int
    fmt.Scan(&a, &b)
    if a >= b {
        permutasi(a, b)
    } else {
        permutasi(b, a)
    }
}

func faktorial(n int, hasil *int) {
    //menggunakan pointer agar dapat mengembalikan nilai tanpa return
    *hasil = 1
    for i := 1; i <= n; i++ {
        *hasil *= i
    }
}

func permutasi(n, r int) {
    var hasiln, hasilnr int
    faktorial(n, &hasiln)
    faktorial(n-r, &hasilnr)
    fmt.Println("Hasil permutasi:", hasiln/hasilnr)
}
```

SCREENSHOOT OUTPUT



DESKRIPSI PROGRAM

Kode ini adalah program sederhana dalam bahasa Go untuk menghitung permutasi dari dua bilangan input. Program dimulai dengan mengimpor paket "fmt" untuk menangani input dan output. Di fungsi main, dua variabel a dan b dideklarasikan untuk menyimpan input dari pengguna. Setelah menerima input, program memeriksa apakah a lebih besar atau sama dengan b, lalu memanggil fungsi permutasi dengan urutan yang tepat. Fungsi

faktorial digunakan untuk menghitung nilai faktorial dari sebuah bilangan dengan memanfaatkan pointer agar nilai faktorial dapat dikembalikan tanpa return. Kemudian, fungsi permutasi menghitung permutasi berdasarkan rumus $n! / (n - r)!$ dengan memanggil faktorial untuk menghitung faktorial n dan $n-r$, dan hasilnya dicetak ke layar.

2.

SOURCECODE

```
package main

import "fmt"

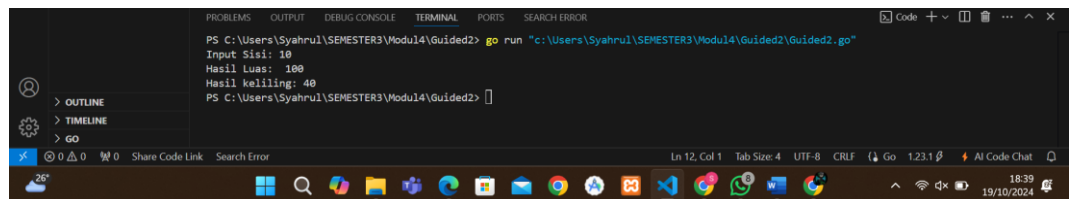
func main() {
    var s int
    fmt.Print("Input Sisi: ")
    fmt.Scan(&s)

    luasPersegi(s)
    kelilingPersegi(s)
}

func luasPersegi(s int) {
    hasil := s * s
    fmt.Println("Hasil Luas: ", hasil)
}

func kelilingPersegi(s int) {
    hasil := 4 * s
    fmt.Print("Hasil keliling: ", hasil)
}
```

SCREENSHOOT OUTPUT

A screenshot of a terminal window showing the execution of a Go program. The terminal output is as follows:

```
PS C:\Users\Syahrul\SEMESTER3\Modul4\Guided2> go run "c:\Users\Syahrul\SEMESTER3\Modul4\Guided2\Guided2.go"
Input Sisi: 10
Hasil Luas: 100
Hasil keliling: 40
PS C:\Users\Syahrul\SEMESTER3\Modul4\Guided2>
```

The terminal window has a dark theme and shows the standard Windows command prompt interface.

DESKRIPSI PROGRAM

Kode di atas adalah program sederhana dalam bahasa Go yang menghitung luas dan keliling persegi berdasarkan panjang sisi yang diinput oleh pengguna. Program dimulai dengan mendeklarasikan variabel `s` untuk menyimpan input sisi persegi dari pengguna. Setelah pengguna memasukkan nilai sisi, program memanggil dua fungsi: `luasPersegi(s)` untuk menghitung dan menampilkan luas persegi (dengan rumus $s * s$), dan `kelilingPersegi(s)` untuk menghitung serta menampilkan keliling persegi.

(dengan rumus $4 * s$). Kedua hasil ini dicetak ke layar menggunakan fungsi `fmt.Println` dan `fmt.Print`.

III. UNGUIDED

Soal Studi Case

1. BELUM JADI

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 kalian membantu Jonas? (tidak tentunya ya :p) Masukan terdiri dari empat buah bilangan asli a, b, c, dan d yang dipisahkan oleh spasi, dengan syarat $a \leq c$ 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!

$$P(n, r) = \frac{n!}{(n-r)!}, \text{ sedangkan } C(n, r) = \frac{n!}{r!(n-r)!}$$

Contoh

No	Masukan	Keluaran	Penjelasan
1	5 10 3 10	60 10 3628800 1	$P(5, 3) = 5! / 2! = 120 / 2 = 60$ $C(5, 3) = 5! / (3! \times 2!) = 120 / 12 = 10$ $P(10, 10) = 10! / 0! = 3628800 / 1 = 3628800$ $C(10, 10) = 10! / (10! \times 0!) = 10! / 10! = 1$
2	8 0 2 0	56 28 1 1	

Selesaikan program tersebut dengan memanfaatkan prosedur yang diberikan berikut ini!

```
procedure factorial(in n: integer, in/out hasil:integer)
{I.S. terdefinisi bilangan bulat positif n
 F.S. hasil berisi nilai faktorial dari n}

procedure permutation(in n, r : integer, in/out hasil:integer)
{I.S. terdefinisi bilangan bulat positif n dan r, dan  $n \geq r$ 
 F.S. hasil berisi nilai dari n permutasi r}

procedure combination(in n, r : integer, in/out hasil:integer)
{I.S. terdefinisi bilangan bulat positif n dan r, dan  $n \geq r$ 
 F.S. hasil berisi nilai dari n kombinasi r}
```

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 faktN, faktNR int
    factorial(n, &faktN)
    factorial(n-r, &faktNR)
    *hasil = faktN / faktNR
}

func combination(n, r int, hasil *int) {
    var faktN, faktR, faktNR int
    factorial(n, &faktN)
    factorial(r, &faktR)
    factorial(n-r, &faktNR)
    *hasil = faktN / (faktR * faktNR)
}

func hitungDanCetak(a, b, c, d, e, f int) {
    var p_ab, c_ab, p_cd, c_cd, p_ef, c_ef int

    permutation(a, b, &p_ab)
    combination(a, b, &c_ab)
    permutation(c, d, &p_cd)
    combination(c, d, &c_cd)
    permutation(e, f, &p_ef)
    combination(e, f, &c_ef)

    fmt.Printf("%d %d\n", p_ab, c_ab)
    fmt.Printf("%d %d\n", p_cd, c_cd)
    fmt.Printf("%d %d\n", p_ef, c_ef)
}

func main() {
    var a, b, c, d, e, f int

    fmt.Print("Masukkan nilai: ")
    fmt.Scanf("%d %d %d %d %d %d", &a, &b, &c, &d, &e, &f)

    hitungDanCetak(a, b, c, d, e, f)
}

```



Screenshoot Output

Deskripsi Program

Program di atas adalah implementasi untuk menghitung **permutasi** dan **kombinasi** dari empat bilangan yang dimasukkan pengguna, yaitu a, b, c, dan d. Program dimulai dengan meminta pengguna untuk memasukkan empat bilangan tersebut. Kemudian, terdapat fungsi factorial yang digunakan untuk menghitung faktorial dari bilangan yang diperlukan dalam perhitungan permutasi dan kombinasi. Fungsi permutation digunakan untuk menghitung permutasi $P(n,r)$ dengan rumus $P(n,r) = \frac{n!}{(n-r)!}$, sedangkan fungsi combination digunakan untuk menghitung kombinasi $C(n,r)$ dengan rumus $C(n,r) = \frac{n!}{r!(n-r)!}$. Program mengecek apakah syarat $a \geq c$ dan $b \geq d$ terpenuhi sebelum melakukan perhitungan. Jika syarat terpenuhi, program akan mencetak hasil permutasi dan kombinasi dari a terhadap c dan b terhadap d. Jika tidak, program akan mencetak pesan bahwa syarat tidak terpenuhi.

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 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).

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 (
    "bufio"
    "fmt"
    "os"
    "strings"
)

func hitungSkor(masukan string) (nama string, soalDiselesaikan, totalWaktu int) {
    data := strings.Fields(masukan)
    nama = data[0]
    soalDiselesaikan = 0
    totalWaktu = 0

    for i := 1; i < len(data); i++ {
        var waktu int
        fmt.Sscanf(data[i], "%d", &waktu)
        if waktu <= 300 {
            soalDiselesaikan++
            totalWaktu += waktu
        }
    }
    return nama, soalDiselesaikan, totalWaktu
}

func main() {
    scanner := bufio.NewScanner(os.Stdin)

    var masukan []string

    fmt.Println("Masukkan data peserta (ketik 'Selesai' untuk berhenti):")
    for {
        scanner.Scan()
        input := scanner.Text()

        if strings.ToLower(input) == "selesai" {
            break
        }
    }
}
```

```

    }
    masukan = append(masukan, input)
}

var pemenang string
maxSoal := 0
minWaktu := 0

for _, m := range masukan {
    nama, soal, waktu := hitungSkor(m)

    if soal > maxSoal || (soal == maxSoal && waktu < minWaktu) {
        pemenang = nama
        maxSoal = soal
        minWaktu = waktu
    }
}

fmt.Printf("Pemenang: %s\nJumlah soal diselesaikan: %d\nTotal waktu: %d\n", pemenang, maxSoal, minWaktu)
}

```

Screenshoot Output

```

PS C:\Users\Syahrul\SEMESTER3\Modul4\Unguided2> go run "c:\Users\Syahrul\SEMESTER3\Modul4\Unguided2\Unguided2.go"
i):
Astuti 20 50 301 301 61 71 75 10
Bertha 25 47 301 26 50 60 65 21
Selesai
Pemenang: Bertha
Jumlah soal diselesaikan: 7
Total waktu: 294
PS C:\Users\Syahrul\SEMESTER3\Modul4\Unguided2>

```

Deskripsi Program

Kode di atas adalah program Golang yang memungkinkan pengguna memasukkan data peserta secara manual, menghitung skor berdasarkan jumlah soal yang diselesaikan dan waktu yang dihabiskan, lalu menentukan pemenang. Program menggunakan prosedur `hitungSkor` untuk menghitung jumlah soal yang diselesaikan dan total waktu yang dibutuhkan, di mana soal yang diselesaikan dalam waktu kurang dari atau sama dengan 300 menit dianggap valid. Input dimasukkan melalui terminal, dan pengguna dapat menambahkan beberapa peserta hingga mengetikkan "Selesai". Program kemudian menentukan pemenang berdasarkan siapa yang menyelesaikan soal terbanyak. Jika jumlah soal sama, peserta dengan waktu penyelesaian paling

sedikit akan menang. Hasil akhirnya berupa nama pemenang, jumlah soal yang diselesaikan, dan total waktu penyelesaian..

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 $1/2n$, 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. Pencetakan deret harus dibuat dalam prosedur cetak Deret yang mempunyai 1 parameter formal, yaitu nilai dari suku awal.

prosedure cetak Deret (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) {
    for n != 1 {
        fmt.Printf("%d ", n)
        if n%2 == 0 {
            n = n / 2
        } else {
            n = 3*n + 1
        }
    }
}
```

```

        n = 3*n + 1
    }
}

fmt.Printf("I\n")
}


func main() {
    var n int

    fmt.Print("Masukkan bilangan bulat positif: ")
    fmt.Scanln(&n)

    if n > 0 && n < 1000000 {
        cetakDeret(n)
    } else {
        fmt.Println("Masukan harus bilangan positif lebih kecil dari 1000000")
    }
}

```

Screenshoot



```

PS C:\Users\Syahrul\SEMESTER3\Modul4\Unguided3> go run "c:\Users\Syahrul\SEMESTER3\Modul4\Unguided3\Unguided3.go"
Masukkan bilangan bulat positif: 22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS C:\Users\Syahrul\SEMESTER3\Modul4\Unguided3>

```

Deskripsi Program

Program di atas mencetak deret bilangan berdasarkan *Collatz Conjecture*. Pengguna memasukkan bilangan bulat positif, dan program mencetak deret dengan aturan: jika bilangan genap, dibagi dua; jika ganjil, dikalikan tiga dan ditambah satu, hingga mencapai 1. Fungsi cetakDeret mengolah deret tersebut, sedangkan di fungsi main, program memvalidasi input agar berada di antara 1 dan kurang dari 1.000.000. Jika tidak, program menampilkan pesan kesalahan.

KESIMPULAN

Fungsi di Golang merupakan blok kode yang dirancang untuk melakukan tugas tertentu dan dapat dipanggil berulang kali di berbagai bagian program. Penggunaan fungsi membantu dalam menyusun program dengan lebih terstruktur, meningkatkan keterbacaan, dan memudahkan pemeliharaan kode. Fungsi dapat menerima parameter untuk memproses data yang dikirimkan dan dapat mengembalikan nilai setelah eksekusi, yang membuatnya sangat fleksibel. Dalam Golang, fungsi dapat didefinisikan dengan tipe data parameter yang jelas, memungkinkan penanganan berbagai tipe data dengan cara yang aman dan efisien. Selain itu, Golang juga mendukung fungsi sebagai tipe data pertama (first-class citizens), yang berarti fungsi dapat disimpan dalam variabel, dipassing sebagai argumen ke fungsi lain, atau bahkan mengembalikan fungsi dari fungsi lain. Hal ini mendukung pengembangan paradigma pemrograman fungsional dan menjadikan Golang alat yang kuat untuk mengembangkan aplikasi yang kompleks dengan cara yang lebih modular dan terorganisir. Dengan menggunakan fungsi secara efektif, pengembang dapat meningkatkan produktivitas dan kualitas kode, menjadikannya lebih mudah untuk dikembangkan dan diuji.

DAFTAR PUSTAKA

<https://blog.myskill.id/istilah-dan-tutorial/golang-pengertian-fungsi-dan-keunggulannya/>