

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL IV  
PROSEDUR**



**Disusun Oleh :**

Maulisa Elvita Sari / 2311102259

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

Prosedur dalam pemrograman adalah sekelompok perintah atau instruksi yang didefinisikan untuk melakukan tugas tertentu, dan dapat dipanggil berkali-kali dari bagian berbeda dalam program. Dalam bahasa Go (Golang), prosedur umumnya direpresentasikan oleh **fungsi**. Fungsi dalam Go dapat berperan sebagai prosedur jika mereka tidak mengembalikan nilai atau hanya mengerjakan suatu tindakan.

Prinsip Dasar Pemrograman Prosedural:

- Modularitas: Program dibagi menjadi bagian-bagian yang lebih kecil yang disebut modul atau fungsi.
- Urutan Instruksi: Program dijalankan dengan urutan instruksi secara linear.
- Reusabilitas Kode: Fungsi-fungsi yang dibuat bisa dipanggil berulang kali sehingga mengurangi duplikasi kode.
- Struktur Kontrol: Meliputi penggunaan kondisi (if-else) dan perulangan (loops).

## II. GUIDED

### Soal Studi Case

#### Sourcecode

```
package main

import "fmt"

func main() {
    var x, y int
    fmt.Print("Masukkan dua angka: ")
    fmt.Scanln(&x, &y)

    if x < y {
        x, y = y, x
    }

    hitungPermutasi(x, y)
}

func hitungPermutasi(n, r int) {
    fmt.Printf("Menghitung P(%d, %d)\n", n, r)
    permutasi(n, r)
}

func faktorial(n int, hasil *int) {
    *hasil = 1
    for j := 2; j <= n; j++ {
        *hasil *= j
    }
}

func permutasi(n, r int) {
    var factN, factNMinR int
    faktorial(n, &factN)
    faktorial(n-r, &factNMinR)
    fmt.Printf("Hasil permutasi(%d, %d) adalah: %d\n", n, r,
factN/factNMinR)
}
```

## Screenshoot Output

```
Masukkan dua angka: 2 5
Menghitung P(5, 2)
Hasil permutasi(5, 2) adalah: 20
```

## Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program tersebut merupakan program dalam bahasa GO dimana program di atas meminta pengguna untuk menghitung permutasi dari dua bilangan bulat dengan menggunakan prosedur.

Permutasi adalah cara menghitung berapa banyak cara mengatur objek dari kumpulan yang lebih besar tanpa memperhatikan urutan. Pada permutasi, jumlah urutan yang dipilih penting. Program ini akan meminta dua angka dari pengguna, menghitung permutasi, dan menampilkan hasilnya. Langkah pertama yaitu input data, pengguna diminta memasukkan dua angka, misalnya  $x = 5$  dan  $y = 3$ . program kemudian memastikan bahwa  $x$  selalu lebih besar dari  $y$ . Jika tidak, kedua angka akan ditukar secara otomatis. Langkah kedua yaitu pemanggilan fungsi, hitungpermutasi: setelah nilai  $x$  dan  $y$  ditentukan, fungsi `hitungPermutasi(x, y)` dipanggil untuk memulai proses penghitungan permutasi. Langkah ketiga yaitu penghitungan faktorial, dalam fungsi `permutasi(n, r)` program memanggil fungsi `faktorial()` dua kali. Langkah keempat yaitu menghitung dan menampilkan permutasi, setelah faktorial  $n$  dan  $n - r$  dihitung, hasil permutasi dihitung dengan membagi  $n!$  Dengan  $(n - r)!$  Hasilnya kemudian ditampilkan di layar.

## Sourcecode

```
package main

import "fmt"

func main() {
    var sisi int
    fmt.Print("Masukkan panjang sisi persegi: ")
    fmt.Scanln(&sisi)

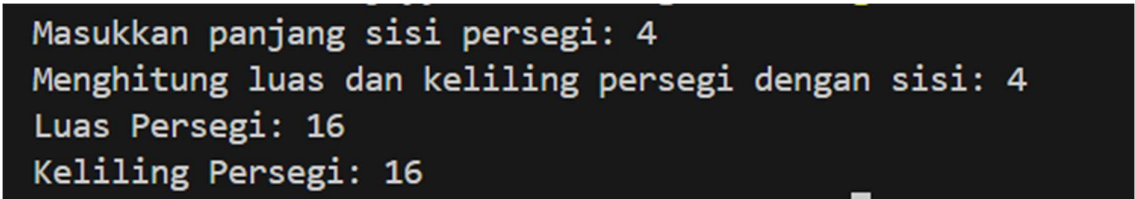
    tampilkanHasil(sisi)
}

func tampilkanHasil(s int) {
    fmt.Println("Menghitung luas dan keliling persegi dengan sisi:", s)
    fmt.Println("Luas Persegi:", hitungLuas(s))
    fmt.Println("Keliling Persegi:", hitungKeliling(s))
}

func hitungLuas(s int) int {
    return s * s
}

func hitungKeliling(s int) int {
    return 4 * s
}
```

## Screenshoot Output



```
Masukkan panjang sisi persegi: 4
Menghitung luas dan keliling persegi dengan sisi: 4
Luas Persegi: 16
Keliling Persegi: 16
```

## Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program tersebut merupakan program dalam bahasa GO dimana program di atas meminta pengguna untuk menghitung luas dan keliling dari sebuah persegi berdasarkan panjang sisi yang dimasukkan oleh pengguna. Dengan menggunakan dua fungsi terpisah untuk menghitung luas dan keliling, program ini memproses input pengguna, menghitung hasilnya, dan menampilkannya di layar. Pengguna hanya perlu memasukkan nilai sisi persegi, dan program akan menghitung serta menampilkan hasilnya.

### 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. 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  $a \cdot c \geq d$  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!

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

#### Sourcecode

```
package main

import "fmt"

func main() {
    var x, y, z, w int
    fmt.Print("Masukkan nilai a, b, c, d: ")
    fmt.Scan(&x, &y, &z, &w)

    if validasiSyarat(x, y, z, w) {
        hitungHasil(x, y, z, w)
    } else {
        fmt.Println("Kondisi tidak terpenuhi.")
    }
}

func validasiSyarat(a, b, c, d int) bool {
    return a*c >= d && b >= d
}

func hitungHasil(a, b, c, d int) {
    var permAC, combAC, permBD, combBD int

    permutation(a, c, &permAC)
    combination(a, c, &combAC)
    permutation(b, d, &permBD)
    combination(b, d, &combBD)
```

```

        fmt.Println("Hasil Permutasi dan Kombinasi untuk (a, c):",
permAC, combAC)
        fmt.Println("Hasil Permutasi dan Kombinasi untuk (b, d):",
permBD, combBD)
    }

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

    func permutation(n, r int, hasil *int) {
        var fn, fnr int
        factorial(n, &fn)
        factorial(n-r, &fnr)
        *hasil = fn / fnr
    }

    func combination(n, r int, hasil *int) {
        var fn, fr, fnr int
        factorial(n, &fn)
        factorial(r, &fr)
        factorial(n-r, &fnr)
        *hasil = fn / (fr * fnr)
    }
}

```

### Screenshoot Output

```

Masukkan nilai a, b, c, d: 8 0 2 0
Hasil Permutasi dan Kombinasi untuk (a, c): 56 28
Hasil Permutasi dan Kombinasi untuk (b, d): 1 1

```

### Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program di atas merupakan program menggunakan bahasa Go yang dimana digunakan untuk menghitung permutasi dan kombinasi dari dua pasangan angka, yaitu (a, c) dan (b, dan d), yang dimasukkan oleh pengguna. Program ini menggunakan beberapa fungsi terpisah untuk menghitung faktorial, permutasi, dan kombinasi, serta melakukan pengecekan apakah nilai yang dimasukkan memenuhi syarat tertentu sebelum melakukan perhitungan. Dengan langkah pertama yang dimana program meminta pengguna untuk memasukkan empat angka, dilanjutkan dengan program kemudian mengecek apakah syarat  $a * c \geq d$  dan  $b \geq d$  terpenuhi.

Jika ya, program akan melanjutkan ke perhitungan permutasi dan kombinasu. Jika tidak, akan tampil pesan. Langkah ketiga, jika kondisi terpenuhi, program akan memanggil fungsi `permutation()` dan `combination()` untuk menghitung permutasi dan kombinasi dari pasangan (a, c) dan (b, d). Setelah semua perhitungan selesai maka program akan menampilkan hasil ke layar.



## 2. Soal 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 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 tau 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.

No	Masukan	Keluaran
1	Astuti 20 50 301 301 61 71 75 10 Bertha 25 47 301 26 50 60 65 21 Selesai	Bertha 7 294

### 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 "fmt"

func hitungSkor(waktu []int) (int, int) {
    totalSoal, totalSkor := 0, 0
    for _, w := range waktu {
        if w < 301 {
            totalSoal++
            totalSkor += w
        }
    }
}
```

```

        return totalSoal, totalSkor
    }

    func main() {
        var peserta int
        fmt.Print("Input jumlah peserta: ")
        fmt.Scan(&peserta)

        pemenang := ""
        maxSoal := -1
        minSkor := 99999

        for i := 0; i < peserta; i++ {
            var nama string
            var waktu [8]int

            fmt.Print("Input nama dan waktu: ")
            fmt.Scan(&nama, &waktu[0], &waktu[1], &waktu[2], &waktu[3],
                &waktu[4], &waktu[5], &waktu[6], &waktu[7])

            totalSoal, totalSkor := hitungSkor(waktu[:])

            if totalSoal > maxSoal || (totalSoal == maxSoal &&
                totalSkor < minSkor) {
                pemenang, maxSoal, minSkor = nama, totalSoal, totalSkor
            }
        }

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

```

### Screenshot Output

```

Input jumlah peserta: 2
Input nama dan waktu: Astuti 20 50 301 301 61 71 75 10
Input nama dan waktu: Bertha 25 47 301 26 50 60 65 21
Pemenang: Bertha
Jumlah soal: 7
Total waktu: 294

```

### Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program di atas merupakan program menggunakan bahasa Go yang dimana digunakan untuk menghitung pemenang dari suatu kompetisi berdasarkan waktu yang dihabiskan oleh peserta

untuk menyelesaikan serangkaian soal. Setelah mengumpulkan data, program akan menentukan pemenang berdasarkan kriteria tertentu, yaitu jumlah soal yang diselesaikan dan total waktu yang dihabiskan. Pemenang adalah peserta yang berhasil menyelesaikan paling banyak soal dalam waktu yang tercepat.

### 3. Soal 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  $2n$ , 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 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. 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 dipisahkan oleh sebuah spasi.

#### Sourcecode

```
package main

import "fmt"

func cetakDeret(n int) {
    for n != 1 {
        fmt.Print(n, " ")
        if n%2 == 0 {
            n = n / 2
        } else {
            n = 3*n + 1
        }
    }
    fmt.Print(n)
}

func main() {
    var input int
    fmt.Print("Masukkan bilangan bulat positif (n < 1000000): ")
    fmt.Scan(&input)

    if input > 0 && input < 1000000 {
        cetakDeret(input)
    } else {
        fmt.Println("Input tidak valid. Pastikan n adalah bilangan positif dan kurang dari 1000000.")
    }
}
```

## Screenshoot Output

```
Masukkan bilangan bulat positif (n < 1000000): 22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
```

## Deskripsi Program

(deskripsikan program apa yang dibuat, memakai algoritma, dan cara kerja program sampai ke output yang dihasilkan dengan bahasa sendiri)

Program di atas merupakan program menggunakan bahasa Go yang dimana digunakan untuk menyatakan bahwa setiap bilangan bulat positif, jika mengikuti aturan tertentu, maka akan selalu berakhir pada angka 1. program ini menerima input dari pengguna dan mencetak urutan bilangan berdasarkan aturan tersebut. Dan program inidirancang untuk, menerima input berupa bilangan positif dari pengguna, dengan batasan bahwa bilangan harus kurang dari 1.000.000. menghitung dan mencetak urutan angka berdasarkan konjektur collatz hingga mencapau angka 1. setelah mencapai angka 1, program mencetak angka tersebut, menandakan akhir dari deret. Maka output deret yang dihasilkan akan terlihat seperti angka yang dihasilkan oleh aturan Collatz.

**Kesimpulan:**

prosedur dalam Go adalah bagian integral dari pemrograman yang membantu dalam mengorganisir kode, meningkatkan keterbacaan, dan memfasilitasi pemrograman yang efisien. Memahami cara mendefinisikan, memanggil, dan menggunakan fungsi adalah dasar yang penting bagi setiap pengembang yang menggunakan bahasa Go. Penggunaan fungsi sebagai prosedur juga membantu dalam penerapan prinsip-prinsip pemrograman modular dan pemrograman berbasis fungsi.