

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL IV
PROCEDURE



Disusun Oleh:
Bayu Kuncoro Adi / 2311102031
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

a. Definisi Procedure

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:

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

Kedudukannya prosedur sama seperti instruksi dasar yang sudah ada sebelumnya (assignment) dan/atau 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, mulal, ..

b. Deklarasi Procedure

Notasi Algoritma	
1	procedure <nama procedure> (<params>)
2	kamus
3	{deklarasi variabel lokal dari procedure}
4	...
5	algoritma
6	{badan algoritma procedure}
7	...
8	endprocedure
Notasi dalam bahasa Go	
9	func <nama procedure> (<params>) {
10	/* deklarasi variabel lokal dari procedure */
11	...
12	/* badan algoritma procedure */
13	...
14	}

	Notasi Algoritma
1	procedure cetakNFibo(in n : integer)
2	kamus
3	f1, f2, f3, i : integer
4	algoritma
5	f2 ← 0
6	f3 ← 1
7	for i ← 1 to n do
8	output(f3)
9	f1 ← f2
10	f2 ← f3
11	f3 ← f1 + f2
12	endfor
13	endprocedure
	Notasi dalam bahasa Go
14	func cetakNFibo(n int) {
15	var f1, f2, f3 int
16	f2 = 0
17	f3 = 1
18	for i := 1; i <= n; i++ {
19	fmt.Println(f3)
20	f1 = f2
21	f2 = f3
22	f3 = f1 + f2
23	}
24	}

c. Deklarasi Procedure

Seperti yang sudah dijelaskan sebelumnya, suatu prosedur hanya akan dieksekusi apabila dipanggil baik secara langsung atau tidak langsung oleh program utama. Tidak langsung di sini maksudnya adalah prosedur dipanggil oleh program utama melalui perantara subprogram yang lain. Pemanggilan suatu prosedur cukup mudah, yaitu dengan hanya menuliskan nama beserta parameter atau argumen yang diminta dari suatu prosedur. Sebagai contoh prosedur cetakNFibo di atas dipanggil dengan menuliskan namanya, kemudian sebuah variabel atau nilai integer tertentu sebagai argumen untuk parameter n. Contoh:

	Notasi Algoritma
1	program contohprosedur
2	kamus
3	x : integer
4	algoritma
5	x ← 5
6	cetakNFibo(x) {cara pemanggilan #1}
7	cetakNFibo(100) {cara pemanggilan #2}

8	endprogram
	Notasi dalam bahasa Go
9	func main() {
10	var x int
11	x = 5
12	cetakNFibo(x) {cara pemanggilan #1}
13	cetakNFibo(100) {cara pemanggilan #2}
14	}

d. Parameter

Suatu subprogram yang dipanggil dapat berkomunikasi dengan pemanggilnya melalui argumen yang diberikan melalui parameter yang dideklarasikan pada subprogramnya. Berikut ini Jenis atau pembagian dari parameter. Berdasarkan letak penulisannya pada program, maka parameter dapat dikelompokkan menjadi dua, yaitu parameter formal dan parameter actual.

- Parameter Formal

Parameter formal adalah parameter yang ditulis pada saat deklarasi suatu subprogram, parameter ini berfungsi sebagai petunjuk bahwa argumen apa saja yang diperlukan pada saat pemanggilan subprogram. Sebagai contoh parameter Jarl_Jarl, inggl pada deklarasi fungsi volumeTabung adalah parameter formal (teks berwarna merah). Artinya ketika memanggil volumeTabung maka kita harus mempersiapkan dua integer (berapapun nilainya) sebagai Jari_Jari dan tinggi.

- Parameter Aktual

Sedangkan parameter aktual adalah argumen yang digunakan pada bagian parameter saat pemanggilan suatu subprogram. Banyaknya argumen dan tipe data yang terdapat pada parameter aktual harus mengikuti parameter formal. Sebagai contoh argumen `r,t, 15 14 dan 100` pada contoh kode di atas (teks berwarna biru) adalah parameter aktual, yang menyatakan nilai yang kita berikan sebagai jarl-jari dan tinggi.

Notasi Algoritma	
<pre> function f1(x,y : integer) → real kamus hasil : real algoritma hasil ← 2*x - 0.5*y + 3 return hasil endfunction procedure f2(in x,y : integer, in/out hasil:real) algoritma hasil ← 2*x - 0.5*y + 3 endprocedure program Contoh kamus a,b : integer c : real algoritma input(a,b) f2(a,b,c) output(c, f1(b,a)) endprogram </pre>	<p><code>x</code> dan <code>y</code> pada fungsi <code>f1</code> dan prosedur <code>f2</code> adalah pass by value,</p> <p>sedangkan variabel <code>hasil</code> pada prosedur <code>f2</code> adalah pass by reference.</p> <p>Untuk pemanggilan dengan notasi pseudocode masih sama dengan materi yang sudah dipelajari sebelumnya</p>
Notasi dalam bahasa Go	
<pre> 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.Scan(&a,&b) f2(a,b,&c) output(c, f1(b,a)) endprogram </pre>	<p><code>x</code> dan <code>y</code> pada fungsi <code>f1</code> dan prosedur <code>f2</code> adalah pass by value,</p> <p>sedangkan variabel <code>hasil</code> pada prosedur <code>f2</code> adalah pass by reference.</p> <p>Karena variabel <code>hasil</code> adalah pointer to float64, maka untuk mengaksesnya menggunakan simbol bintang (*) pada variabelnya.</p> <p>Pada bahasa Go saat pemanggilan prosedur <code>f2</code>, maka parameter aktual untuk pass by reference harus diberi ampersand "&", contohnya <code>&c</code></p>

II. GUIDED

1. Contoh program dengan Function

```
package main

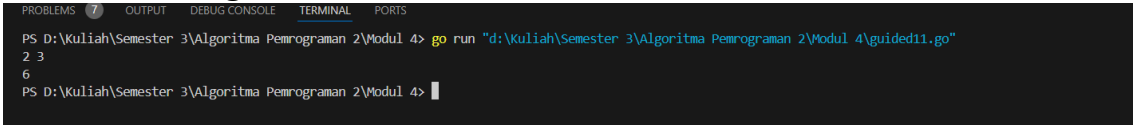
import "fmt"

func main() {
    var a, b int
    fmt.Scan(&a, &b) // Membaca input dari pengguna untuk dua nilai
    integer a dan b
    if a >= b {
        // Jika a lebih besar atau sama dengan b, panggil prosedur
        permutasi dengan parameter (a, b)
        permutasi(a, b)
    } else {
        // Jika b lebih besar dari a, panggil prosedur permutasi
        dengan parameter (b, a)
        permutasi(b, a)
    }
}

func faktorial(n int) int {
    var hasil int = 1
    // Loop untuk menghitung faktorial dari n
    for i := 1; i <= n; i++ {
        hasil *= i // Mengalikan hasil dengan nilai i pada setiap
        iterasi
    }
    return hasil // Mengembalikan hasil faktorial
}

func permutasi(n, r int) {
    // Menghitung permutasi nPr dan langsung mencetak hasilnya
    hasil := faktorial(n) / faktorial(n-r)
    fmt.Println(hasil) // Mencetak hasil permutasi
}
```

Screenshoot Program



PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4\guided11.go"

2 3

6

PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4> █

Deskripsi dan Cara Kerja Program

Program di atas menghitung dan mencetak nilai permutasi (nPr) berdasarkan input dua bilangan bulat yang dimasukkan oleh pengguna. Pertama, program meminta dua nilai input, yaitu (a) dan (b) . Jika (a) lebih besar atau sama dengan (b) , maka program akan menghitung permutasi dengan (a) sebagai (n) dan (b) sebagai (r) , sebaliknya jika (b) lebih besar, perhitungan dilakukan dengan (b) sebagai (n) dan (a) sebagai (r) . Fungsi `faktorial` menghitung nilai faktorial dari bilangan yang diberikan, dan fungsi `permutasi` menggunakan fungsi faktorial untuk menghitung permutasi dengan rumus $(nPr = \frac{n!}{(n-r)!})$. Hasil akhirnya dicetak ke layar.

2. Menhitung luas dan keliling persegi

```
package main

import "fmt"

// Prosedur untuk menghitung dan mencetak luas persegi
func hitungLuas(sisi float64) {
    luas := sisi * sisi
    fmt.Printf("Luas persegi: %.2f\n", luas) // Mencetak hasil luas
}

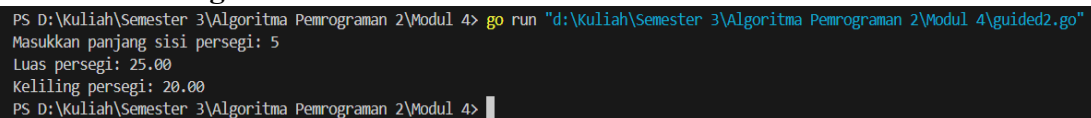
// Prosedur untuk menghitung dan mencetak keliling persegi
func hitungKeliling(sisi float64) {
    keliling := 4 * sisi
    fmt.Printf("Keliling persegi: %.2f\n", keliling) // Mencetak hasil
    keliling
}

func main() {
    var sisi float64

    fmt.Print("Masukkan panjang sisi persegi: ")
    fmt.Scan(&sisi)

    hitungLuas(sisi) // Memanggil prosedur hitungLuas
    hitungKeliling(sisi) // Memanggil prosedur hitungKeliling
}
```

Screenshoot Program



```
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4\guided2.go"
Masukkan panjang sisi persegi: 5
Luas persegi: 25.00
Keliling persegi: 20.00
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4> |
```

Deskripsi dan Cara Kerja Program

Program di atas menghitung dan mencetak luas serta keliling persegi berdasarkan panjang sisi yang dimasukkan oleh pengguna. Setelah pengguna memasukkan nilai panjang sisi, program memanggil dua prosedur: `hitungLuas` untuk menghitung luas persegi dengan rumus $(\text{sisi})^2$, dan `hitungKeliling` untuk menghitung keliling persegi dengan rumus $4 \times \text{sisi}$. Hasil dari masing-masing perhitungan dicetak ke layar dengan format dua angka di belakang koma untuk memastikan hasil yang lebih rapi.

III. 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 kalilan 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 2 cdanb 2 d. 3 Keluaran terdiri dari dua baris. Bans pertama adalah hasil permutasi dan kombinasi a terhadap c, sedangkan baris kedua adalah hasil permutasi dan kombinasi b terhadap d Catstan: permutasi (P) dan kombinast (C) dari n terhadap r ($n \geq r$) dapat dihitung dengan menggunakan persamaan berikut:

```
package main

import "fmt"

// Prosedur faktorial untuk menghitung faktorial dari n
func faktorial(n int, hasil *int) {
    *hasil = 1
    for i := 2; i <= n; i++ {
        *hasil *= i
    }
}

// Prosedur permutasi untuk menghitung permutasi P(n, r)
func permutasi(n, r int, hasil *int) {
    var faktN, faktNR int
    faktorial(n, &faktN) // Faktorial n
    faktorial(n-r, &faktNR) // Faktorial (n-r)
    *hasil = faktN / faktNR // Permutasi P(n, r)
}

// Prosedur kombinasi untuk menghitung kombinasi C(n, r)
func kombinasi(n, r int, hasil *int) {
    var faktN, faktR, faktNR int
    faktorial(n, &faktN) // Faktorial n
    faktorial(r, &faktR) // Faktorial r
    faktorial(n-r, &faktNR) // Faktorial (n-r)
    *hasil = faktN / (faktR * faktNR) // Kombinasi C(n, r)
}

func main() {
    var a, b, c, d int
    var permA, combA, permB, combB int

    // Membaca input dari pengguna
```

```

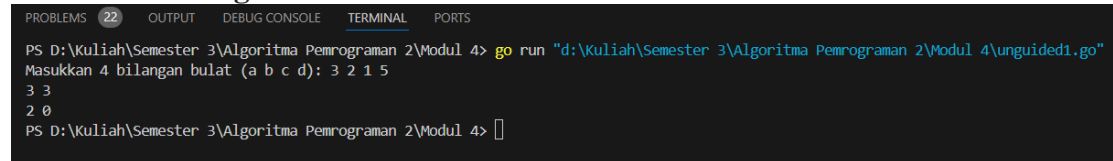
fmt.Print("Masukkan 4 bilangan bulat (a b c d): ")
fmt.Scan(&a, &b, &c, &d)

// Menghitung permutasi dan kombinasi untuk (a, c) dan (b, d)
permutasi(a, c, &permA)
kombinasi(a, c, &combA)
permutasi(b, d, &permB)
kombinasi(b, d, &combB)

// Mencetak hasil
fmt.Println(permA, combA) // Hasil permutasi dan kombinasi a
terhadap c
fmt.Println(permB, combB) // Hasil permutasi dan kombinasi b
terhadap d
}

```

Screenshoot Program



```

PROBLEMS 22 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4\unguided1.go"
Masukkan 4 bilangan bulat (a b c d): 3 2 1 5
3 3
2 0
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4>

```

Deskripsi dan Cara Kerja Program

Program di atas menghitung dan mencetak nilai permutasi dan kombinasi dari dua pasang bilangan berdasarkan input pengguna. Setelah menerima empat bilangan bulat (a) , (b) , (c) , dan (d) , program menggunakan prosedur khusus untuk menghitung **faktorial**, **permutasi** $(P(n, r))$, dan **kombinasi** $(C(n, r))$. Prosedur **faktorial** menghitung faktorial dari suatu bilangan, sementara **permutasi** menghitung permutasi $(P(n, r) = \frac{n!}{(n-r)!})$ dan **kombinasi** menghitung kombinasi $(C(n, r) = \frac{n!}{r!(n-r)!})$. Hasil permutasi dan kombinasi dari (a) terhadap (c) serta (b) terhadap (d) ditampilkan dalam dua baris output.

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 mencart 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. prosedur hitungskor(in/out soal, skor integer) Setiap baris masukan dimulai dengan satu string nama peserta tersebut diikuti dengan 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.

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

```
package main

import (
    "fmt"
    "strings"
)

const maxTime = 301 // Waktu maksimal jika soal tidak selesai

// Prosedur untuk menghitung skor (jumlah soal yang diselesaikan dan total waktu)
func hitungSkor(waktu [8]int, soal *int, skor *int) {
    *soal = 0
    *skor = 0

    for _, waktuSoal := range waktu {
        if waktuSoal < maxTime { // Jika soal diselesaikan
            *soal++
            *skor += waktuSoal
        }
    }
}

func main() {
    var pemenang string
    var soalPemenang, skorPemenang int
    soalPemenang = 0
    skorPemenang = maxTime * 8 // Nilai awal skor pemenang maksimal

    for {
        var nama string
```

```

        var waktu [8]int

        // Membaca input nama peserta
        fmt.Print("Masukkan nama peserta (atau 'Selesai' untuk
mengakhiri): ")
        fmt.Scan(&nama)
        nama = strings.TrimSpace(nama)

        // Jika input adalah 'Selesai', hentikan loop
        if nama == "Selesai" {
            break
        }

        // Membaca waktu penyelesaian 8 soal
        fmt.Print("Masukkan waktu penyelesaian 8 soal (dalam
menit): ")
        for i := 0; i < 8; i++ {
            fmt.Scan(&waktu[i])
        }

        // Hitung soal yang diselesaikan dan total waktu
        var soal, skor int
        hitungSkor(waktu, &soal, &skor)

        // Tentukan pemenang berdasarkan jumlah soal dan total
waktu
        if soal > soalPemenang || (soal == soalPemenang && skor <
skorPemenang) {
            pemenang = nama
            soalPemenang = soal
            skorPemenang = skor
        }

        // Cetak pemenang
        fmt.Printf("Pemenang: %s, Soal yang diselesaikan: %d, Total
waktu: %d menit\n", pemenang, soalPemenang, skorPemenang)
    }
}

```

Screenshoot Program

PROBLEMS 24 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4\unguided2.go"
Masukkan nama peserta (atau 'Selesai' untuk mengakhiri): Bayu
Masukkan waktu penyelesaian 8 soal (dalam menit): 20 11 23 45 68 54 33 22
Masukkan nama peserta (atau 'Selesai' untuk mengakhiri): Selesai
Pemenang: Bayu, Soal yang diselesaikan: 8, Total waktu: 276 menit
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4> █

```

Deskripsi dan Cara Kerja Program

Program di atas digunakan untuk menentukan pemenang dari kompetisi pemrograman berdasarkan jumlah soal yang diselesaikan dan total waktu pengerjaan. Setiap peserta menginputkan nama dan waktu penyelesaian 8 soal (dalam menit). Jika sebuah soal tidak diselesaikan, waktu defaultnya adalah 301 menit. Program menggunakan prosedur `hitungSkor` untuk menghitung jumlah soal yang diselesaikan dan total waktu yang diperlukan untuk soal yang selesai. Dalam setiap iterasi, program membandingkan skor peserta saat ini dengan skor pemenang sebelumnya. Peserta yang menyelesaikan lebih banyak soal menjadi pemenang, dan jika jumlah soal yang diselesaikan sama, peserta dengan waktu pengerjaan paling singkat yang menang. Program berhenti jika pengguna mengetik "Selesai", dan hasil akhir akan menampilkan nama pemenang, jumlah soal yang diselesaikan, serta total waktu pengerjaannya.

3. Diberikan sebuah persamaan sebagai berikut:

Skiena dan Revlla 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

Algoritma a o n

Pemrograman 2

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. Pencetak 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.

```
package main

import "fmt"

// Prosedur untuk mencetak deret bilangan
func cetakDeret(n int) {
    for n != 1 {
        fmt.Printf("%d ", n) // Cetak nilai n
        if n%2 == 0 {        // Jika n genap
            n = n / 2
        } else { // Jika n ganjil
            n = 3*n + 1
        }
    }
    fmt.Println(1) // Cetak nilai akhir 1
}

func main() {
    var n int
    fmt.Print("Masukkan nilai awal deret: ")
    fmt.Scan(&n) // Ambil input dari pengguna
    cetakDeret(n) // Panggil prosedur cetakDeret
}
```

Screenshot Program

```
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4> go run "d:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4\unguided3.go"
Masukkan nilai awal deret: 6
6 3 10 5 16 8 4 2 1
PS D:\Kuliah\Semester 3\Algoritma Pemrograman 2\Modul 4> █
```

Deskripsi dan Cara Kerja Program

Program ini mencetak deret bilangan berdasarkan aturan ****Collatz Conjecture****, di mana pengguna memasukkan nilai awal `n`, dan program akan terus memproses nilai tersebut hingga mencapai angka 1. Jika nilai `n` genap, maka `n` dibagi 2, dan jika `n` ganjil, `n` akan dihitung dengan rumus $3*n + 1$. Proses ini akan diulangi hingga `n` menjadi 1, yang kemudian dicetak sebagai nilai akhir. Pada setiap iterasi, nilai `n` dicetak ke layar. Program ini menggunakan prosedur `cetakDeret` untuk menjalankan logika ini, dan input awal `n` diperoleh dari pengguna melalui fungsi `fmt.Scan`.

KESIMPULAN

Pada praktikum ini, kami mempelajari penggunaan prosedur (procedure) dalam pemrograman Go. Prosedur adalah subprogram yang digunakan untuk membagi sebuah program besar menjadi bagian-bagian yang lebih kecil dan terstruktur, sehingga lebih mudah dipahami dan dipelihara. Penggunaan prosedur sangat penting dalam membuat program yang modular, sehingga setiap bagian dari program bisa berdiri sendiri dan diujikan secara independen. Kami juga mempelajari bagaimana cara kerja parameter formal dan aktual yang memungkinkan prosedur untuk menerima input dan menghasilkan output tanpa harus mengembalikan nilai seperti pada fungsi.

Melalui berbagai contoh, kami berhasil mengimplementasikan konsep-konsep ini dalam program yang menghitung faktorial, permutasi, kombinasi, serta melakukan perhitungan deret berdasarkan aturan Collatz. Selain itu, kami mempelajari cara memanfaatkan prosedur untuk menyelesaikan permasalahan kompetisi pemrograman dengan menentukan pemenang berdasarkan soal yang diselesaikan. Praktikum ini membuktikan bahwa penggunaan prosedur sangat berguna untuk mengorganisasi kode, mengurangi kompleksitas, dan meningkatkan keterbacaan serta efisiensi dalam penulisan program.

DAFTAR PUSTAKA

Modul 4 Praktikum Algoritma dan Pemrograman 2