

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL IV  
PROSEDUR**



Oleh :

NAMA : Felix Pedrosa Valentino

NIM : 103112400056

KELAS : IF – 12 – 01

**S1 TEKNIK INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO**

**2025**

## I. DASAR TEORI

### Definisi Prosedur

Prosedur dapat diartikan sebagai kumpulan instruksi program yang digabung menjadi satu instruksi baru. Tujuan pembentukan prosedur adalah untuk menyederhanakan kode program yang kompleks dalam suatu aplikasi yang besar. Ketika dipanggil dalam program utama, prosedur akan memberikan dampak langsung pada jalannya program tersebut.

Sebuah subprogram disebut sebagai prosedur jika memenuhi dua syarat berikut :

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

Posisi prosedur dalam sebuah program sama pentingnya dengan instruksi dasar yang sudah ada sebelumnya, seperti perintah penugasan, maupun instruksi dari paket (fmt), contohnya fmt. Scan dan fmt. Print. Oleh karena itu, disarankan untuk memberikan nama prosedur yang berbentuk kata kerja atau yang mencerminkan proses yang dilakukan. Contohnya: cetak, hitungRerata, cariNilai, belok, mulai, dan sebagainya.

### Cara Memanggil Prosedur

Seperti yang dijelaskan sebelumnya, suatu prosedur hanya akan dieksekusi jika dipanggil, baik secara langsung maupun tidak langsung oleh program utama. Pemanggilan tidak langsung berarti bahwa prosedur tersebut dipanggil oleh program utama melalui subprogram lain sebagai perantaranya.

Untuk memanggil suatu prosedur, caranya cukup sederhana. Anda hanya perlu menuliskan nama prosedur beserta parameter atau argumen yang diperlukan. Sebagai contoh, prosedur cetakNFibo dapat dipanggil dengan menyebutkan namanya, diikuti dengan sebuah variabel atau nilai integer tertentu yang akan dijadikan argumen untuk parameter n.

### Parameter dalam Subprogram

Subprogram yang dipanggil dapat berinteraksi dengan pemanggilnya melalui argumen yang diteruskan melalui parameter yang didefinisikan dalam subprogram tersebut. Parameter dapat dibedakan menjadi beberapa jenis berdasarkan letak penulisannya, yaitu parameter formal dan parameter aktual.

#### 1. Parameter Formal

Parameter formal adalah parameter yang dituliskan saat deklarasi suatu subprogram. Fungsinya adalah sebagai petunjuk mengenai argumen yang dibutuhkan ketika subprogram dipanggil. Contohnya, dalam deklarasi fungsi ``volumeTabung``, parameter seperti ``jari_jari`` dan ``tinggi`` merupakan contoh dari parameter formal. Ini berarti, saat kita memanggil ``volumeTabung``, kita harus menyiapkan dua nilai integer untuk ``jari_jari`` dan ``tinggi``.

#### 2. Parameter Aktual

Sebaliknya, parameter aktual adalah argumen yang digunakan saat memanggil subprogram sesuai dengan parameter yang telah didefinisikan. Jumlah dan tipe data dari parameter aktual harus sesuai dengan parameter formal yang ada. Sebagai contoh, argumen seperti ``r``, ``t``, ``15``, ``14``, dan ``100`` yang terdapat dalam kode di atas adalah parameter aktual yang menunjukkan nilai yang diberikan untuk ``jari_jari`` dan ``tinggi``.

Selain itu, parameter juga dapat dikelompokkan berdasarkan cara alokasi memorinya, yaitu pass by value dan pass by reference.

#### 1. Pass by Value

Ketika menggunakan pass by value, nilai dari parameter aktual akan disalin ke variabel lokal (parameter formal) di dalam subprogram. Dengan demikian, parameter aktual dan formal dialokasikan di memori komputer dengan alamat yang berbeda. Subprogram dapat memanfaatkan nilai dari parameter formal untuk berbagai proses, namun tidak dapat mengembalikan informasi ke pemanggil melalui parameter aktual karena pemanggil tidak memiliki akses ke memori yang digunakan oleh subprogram. Pada pseudocode, semua parameter formal dalam fungsi dianggap sebagai pass by value, sedangkan pada prosedur, penulisan parameter formal akan disertai dengan kata kunci ``in``. Dalam bahasa

pemrograman Go, tidak ada kata kunci khusus untuk parameter formal fungsi dan prosedur.

## 2. Pass by Reference (Pointer)

Jika parameter didefinisikan sebagai pass by reference, maka saat pemanggilan, parameter formal berperan sebagai pointer yang menyimpan alamat memori dari parameter aktual. Dengan demikian, setiap perubahan nilai pada parameter formal akan mempengaruhi nilai parameter aktual. Hal ini memungkinkan nilai akhir dapat dimanfaatkan oleh pemanggil setelah subprogram selesai dieksekusi. Disarankan untuk menggunakan pass by reference terutama pada prosedur.

Penulisan parameter pass by reference pada prosedur, baik di pseudocode maupun Go, memerlukan penggunaan kata kunci atau identifier tertentu. Dalam pseudocode, kata kunci 'in/out' digunakan, sedangkan dalam bahasa Go, identifier asterik (\*) diletakkan sebelum tipe data parameter formal yang ditentukan sebagai pass by reference.

Catatan:

- Sebaiknya, parameter dalam fungsi menggunakan pass by value, karena fungsi dapat mengembalikan nilai ke pemanggil tanpa memberikan dampak langsung pada program, meskipun penggunaan pass by reference tetap dimungkinkan.
- Penggunaan pass by reference lebih direkomendasikan untuk prosedur, yang tidak memiliki kemampuan untuk mengembalikan nilai ke pemanggil. Dengan metode ini, prosedur seolah-olah dapat mengirimkan nilai kepada pemanggil.

## II. GUIDED

### 1. Guided 1

Source Code :

```
// Felix Pedrosa V

package main

import "fmt"

// Prosedur untuk menghitung total gaji karyawan
func hitungGaji(nama string, gajiPokok float64, jamLembur int) {
    bonusLembur := float64(jamLembur) * 50000
    totalGaji := gajiPokok + bonusLembur

    fmt.Println("\n=== Slip Gaji ===")
    fmt.Println("Nama Karyawan :", nama)
    fmt.Printf("Gaji Pokok   : Rp%.2f\n", gajiPokok)
    fmt.Printf("Bonus Lembur  : Rp%.2f (%d jam x Rp50,000)\n",
bonusLembur, jamLembur)
    fmt.Printf("Total Gaji    : Rp%.2f\n", totalGaji)
}

func main() {
    var nama string
    var gajiPokok float64
    var jamLembur int

    // Input dari pengguna
    fmt.Print("Masukkan Nama Karyawan: ")
    fmt.Scanln(&nama)

    fmt.Print("Masukkan Gaji Pokok: ")
    fmt.Scanln(&gajiPokok)

    fmt.Print("Masukan Jumlah Jam Lembur: ")
    fmt.Scanln(&jamLembur)
```

```
// Memanggil prosedur dengan data dari pengguna  
hitungGaji(nama, gajiPokok, jamLembur)  
}
```

Output :

```
PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING -  
GOLANG - Alpro 2\alpro2_week4_modul4\103112400056_Guided1\103112400056_Guided1.go"  
Masukkan Nama Karyawan: slamet  
Masukkan Gaji Pokok: 2000000  
Masukkan Jumlah Jam Lembur: 3  
  
=== Slip Gaji ===  
Nama Karyawan : slamet  
Gaji Pokok    : Rp2000000.00  
Bonus Lembur  : Rp150000.00 (3 jam x Rp50,000)  
Total Gaji    : Rp2150000.00
```

Penjelasan Program :

Program di atas merupakan program yang menggunakan bahasa go dan bertujuan untuk menghitung total gaji karyawan berdasarkan gaji pokok dan jumlah jam lembur yang dikerjakan. Ditulis menggunakan bahasa pemrograman Go, aplikasi ini memiliki dua fungsi utama: `hitungGaji` dan `main`.

Fungsi `hitungGaji` menerima tiga parameter, yaitu nama karyawan, gaji pokok, dan jumlah jam lembur yang dilakukan. Di dalam fungsi ini, bonus lembur dihitung dengan mengalikan jumlah jam lembur dengan tarif Rp50. 000 per jam. Selanjutnya, total gaji diperoleh dengan menjumlahkan gaji pokok dan bonus lembur. Hasil perhitungan kemudian ditampilkan dalam format slip gaji, yang mencakup nama karyawan, gaji pokok, bonus lembur, dan jumlah total gaji.

Sementara itu, fungsi `'main'` bertanggung jawab untuk mengambil input dari pengguna, termasuk nama karyawan, gaji pokok, dan jumlah jam lembur. Setelah itu, fungsi ini akan memanggil `hitungGaji` untuk menampilkan hasil perhitungan tersebut.

## 2. Guided 2

Source Code :

```
// Felix Pedrosa V  
  
package main  
  
import "fmt"
```

```
// Prosedur untuk menghitung rata-rata dan menentukan kelulusan
func hitungRataRata(nama string, nilai1, nilai2, nilai3 float64) {
    rataRata := (nilai1 + nilai2 + nilai3) / 3
    status := "Tidak Lulus"
    if rataRata >= 60 {
        status = "Lulus"
    }

    fmt.Println("\n=== Hasil Akademik ===")
    fmt.Println("Nama Mahasiswa :", nama)
    fmt.Printf("Nilai 1      : %.2f\n", nilai1)
    fmt.Printf("Nilai 2      : %.2f\n", nilai2)
    fmt.Printf("Nilai 3      : %.2f\n", nilai3)
    fmt.Printf("Rata-rata    : %.2f\n", rataRata)
    fmt.Println("Status      :", status)
}

func main() {
    var nama string
    var nilai1, nilai2, nilai3 float64

    // Input dari pengguna
    fmt.Print("Masukkan Nama Mahasiswa: ")
    fmt.Scanln(&nama)

    fmt.Print("Masukkan Nilai 1: ")
    fmt.Scanln(&nilai1)

    fmt.Print("Masukkan Nilai 2: ")
    fmt.Scanln(&nilai2)

    fmt.Print("Masukkan Nilai 3: ")
    fmt.Scanln(&nilai3)

    hitungRataRata(nama, nilai1, nilai2, nilai3)
}
```

Output :

```

PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2\alpro2_week4_modul14\10311240056_Guided2\10311240056_Guided2.go"
Masukkan Nama Mahasiswa: Felix
Masukkan Nilai 1: 100
Masukkan Nilai 2: 100
Masukkan Nilai 3: 100

=== Hasil Akademik ===
Nama Mahasiswa : Felix
Nilai 1        : 100.00
Nilai 2        : 100.00
Nilai 3        : 100.00
Rata-rata      : 100.00
Status         : Lulus

```

### Penjelasan Program :

Program diatas merupakan program yang menggunakan bahasa go dan bertujuan untuk menghitung rata-rata nilai akademik mahasiswa dan menentukan status kelulusan berdasarkan nilai tersebut. Ditulis menggunakan bahasa pemrograman Go, program ini terdiri dari dua fungsi utama: `hitungRataRata` dan `main`.

Fungsi `'hitungRataRata'` menerima empat parameter, yaitu nama mahasiswa serta tiga nilai yang diperoleh. Dalam fungsi ini, rata-rata nilai dihitung dengan menjumlahkan ketiga nilai dan membaginya dengan tiga. Status kelulusan ditentukan berdasarkan hasil rata-rata; jika rata-rata mencapai 60 atau lebih, mahasiswa dinyatakan "Lulus", sedangkan jika kurang dari 60, statusnya adalah "Tidak Lulus". Hasil perhitungan ditampilkan dengan jelas, mencakup nama mahasiswa, nilai masing-masing, rata-rata, dan status kelulusan.

Fungsi `main` bertugas untuk menerima input dari pengguna, termasuk nama mahasiswa dan ketiga nilai, sebelum memanggil fungsi `hitungRataRata` untuk menampilkan hasilnya.



### III. UNGUIDED

#### 1. UnGuided 1

Source Code :

```
// Felix Pedrosa V

package main

import "fmt"

func main() {
    var x, y, z, w int
    var permCount, combCount int

    fmt.Scan(&x, &y, &z, &w)

    // Menghitung permutasi dan kombinasi untuk pasangan pertama
    if x >= z {
        calculatePermComb(x, z, &permCount, &combCount)
    } else {
        calculatePermComb(z, x, &permCount, &combCount)
    }
    fmt.Println(permCount, combCount)

    // Menghitung permutasi dan kombinasi untuk pasangan kedua
    if y >= w {
        calculatePermComb(y, w, &permCount, &combCount)
    } else {
        calculatePermComb(w, y, &permCount, &combCount)
    }
    fmt.Println(permCount, combCount)
}

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

func calculatePermComb(n, r int, perm *int, comb *int) {
```

```

var factN, factNMinusR, factR int
if r > n {
    *perm = 0
    *comb = 0
    return
}
factorial(n, &factN)
factorial(n-r, &factNMinusR)
*perm = factN / factNMinusR

factorial(r, &factR)
*comb = factN / (factNMinusR * factR)
}

```

Output :

```

PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING -
GOLANG - Alpro 2\alpro2_week4_modul4\103112400056_Unguided1\103112400056_Unguided1.go"
5 10 3 10
60 10
3628800 1
PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING -
GOLANG - Alpro 2\alpro2_week4_modul4\103112400056_Unguided1\103112400056_Unguided1.go"
8 0 2 0
56 28
1 1

```

Penjelasan Program :

Program diatas ditulis dalam bahasa Go dan bertujuan untuk menghitung jumlah permutasi dan kombinasi dari dua pasangan bilangan bulat yang diberikan oleh pengguna. Setelah pengguna menginput empat bilangan (x, y, z, dan w), program akan menghitung permutasi dan kombinasi untuk pasangan pertama, yaitu (x dan z), serta pasangan kedua, yakni (y dan w). Untuk melakukan perhitungan tersebut, program menggunakan fungsi `calculatePermComb` , yang memanfaatkan fungsi `factorial` untuk menghitung faktorial dari bilangan yang diperlukan. Hasil perhitungan permutasi dan kombinasi untuk masing-masing pasangan kemudian ditampilkan di output.

## 2. UnGuided 2

Source Code :

```

// Felix Pedrosa V

package main

```

```

import (
    "fmt"
    "strings"
)

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

// Fungsi untuk menghitung jumlah soal yang diselesaikan dan total waktu
func hitungNilai(waktuSoal [8]int, jumlahSoal *int, totalWaktu *int) {
    *jumlahSoal = 0
    *totalWaktu = 0
    for _, waktu := range waktuSoal {
        if waktu < waktuMaksimal { // Jika soal diselesaikan
            *jumlahSoal++
            *totalWaktu += waktu
        }
    }
}

func main() {
    var namaPemenang string
    var jumlahSoalPemenang, totalWaktuPemenang int
    jumlahSoalPemenang = 0
    totalWaktuPemenang = waktuMaksimal * 8 // Nilai awal skor
    pemenang maksimal

    for {
        var namaPeserta string
        var waktuPenyelesaian [8]int

        // Membaca input nama peserta
        fmt.Scan(&namaPeserta)
        namaPeserta = strings.TrimSpace(namaPeserta)

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

```

```

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

// Hitung soal yang diselesaikan dan total waktu
var jumlah, total int
hitungNilai(waktuPenyelesaian, &jumlah, &total)

// Tentukan pemenang berdasarkan jumlah soal dan total waktu
if jumlah > jumlahSoalPemenang || (jumlah == jumlahSoalPemenang
&& total < totalWaktuPemenang) {
    namaPemenang = namaPeserta
    jumlahSoalPemenang = jumlah
    totalWaktuPemenang = total
}
}

// Cetak pemenang
fmt.Printf("%s %d %d\n", namaPemenang, jumlahSoalPemenang,
totalWaktuPemenang)
}

```

Output :

```

PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING -
GOLANG - Alpro 2\alpro2_week4_modul4\103112400056_Unguided2\103112400056_Unguided2.go"
Astuti 20 50 301 301 61 71 75 10
Bertha 25 47 301 26 50 60 65 21
Selesai
Bertha 7 294

```

Penjelasan Program :

Program diatas ditulis menggunakan bahasa Go dan berfungsi untuk menentukan pemenang sebuah kompetisi berdasarkan waktu penyelesaian soal. Dalam program ini, pengguna diminta untuk memasukkan nama peserta beserta waktu penyelesaian untuk delapan soal yang ada. Melalui fungsi `hitungNilai`, program akan menghitung jumlah soal yang berhasil diselesaikan serta total waktu yang dihabiskan oleh setiap peserta. Pemenang kompetisi akan ditentukan berdasarkan jumlah soal yang diselesaikan, dengan waktu total sebagai kriteria penentu jika terjadi seri. Setelah semua peserta memasukkan data mereka, program akan

menampilkan nama pemenang, jumlah soal yang diselesaikan, serta total waktu yang dihabiskan.

### 3. UnGuided 3

Source Code :

```
// Felix Pedrosa V

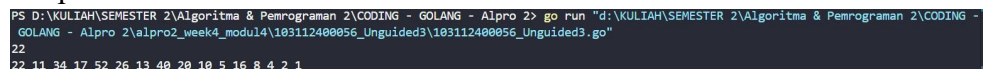
package main

import "fmt"

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

func main() {
    var nilaiAwal int
    fmt.Scan(&nilaiAwal) // Ambil input dari pengguna
    tampilkanDeret(nilaiAwal) // Panggil fungsi tampilkanDeret
}
```

Output :



```
PS D:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2> go run "d:\KULIAH\SEMESTER 2\Algoritma & Pemrograman 2\CODING - GOLANG - Alpro 2\alpro2_week4_modul4\103112400056_Unguided3\103112400056_Unguided3.go"
22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
```

Penjelasan Program :

Program ini ditulis menggunakan bahasa Go dan dirancang untuk mencetak deret bilangan sesuai dengan aturan yang dikenal sebagai konjectur Collatz. Pengguna akan diminta untuk memasukkan sebuah

bilangan bulat sebagai nilai awal. Fungsi `tampilkanDeret` kemudian akan mencetak deret bilangan tersebut hingga mencapai angka 1. Dalam fungsi ini, jika angka yang diberikan adalah genap, maka angka itu akan dibagi dua; sedangkan jika angka itu ganjil, maka akan dikalikan tiga dan ditambahkan satu. Proses ini akan diulang terus menerus hingga angka mencapai 1, yang juga akan dicetak sebagai bagian dari deret.

#### **IV. KESIMPULAN**

Pada modul ini membahas konsep prosedur dalam pemrograman bahasa Go. Prosedur merupakan sekumpulan kode yang dikemas dalam satu kesatuan dengan tujuan untuk mengurangi kompleksitas program yang besar. Prosedur tidak mengembalikan nilai secara langsung dan berfungsi untuk mengeksekusi perintah tertentu dalam program. Dalam Go, deklarasi prosedur menggunakan sintaks `func namaProcedure(params)` , yang dapat ditempatkan baik sebelum maupun setelah fungsi utama `main()` .

Pemanggilan prosedur dilakukan dengan menyebutkan nama prosedur beserta parameter yang diperlukan. Sebagai contoh, prosedur `cetakNFibo(n int)` bisa dipanggil dengan cara `cetakNFibo(5)` . Modul ini juga menjelaskan mengenai parameter dalam prosedur, yang terbagi menjadi parameter formal (yang dideklarasikan di dalam prosedur) dan parameter aktual (nilai yang diberikan saat pemanggilan). Parameter dapat dikategorikan berdasarkan alokasi memorinya menjadi dua jenis yaitu `pass by value` (di mana nilai disalin ke memori baru) dan `pass by reference` (menggunakan pointer, sehingga perubahan dalam prosedur dapat mempengaruhi variabel asli).

## **V. REFERENSI**

Modul 4 - Praktikum Alpro 2