

**LAPORAN**  
**PRAKTIKUM**  
**ALGORITMA DAN PEMROGRAMAN 2**  
**MODUL 4**  
**PROSEDUR**



Oleh:

NAMA: NUFAIL ALAUDDIN TSAQIF

NIM: 103112400084

KELAS: IF-12-01

**S1 TEKNIK INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2025**

## **I. DASAR TEORI**

### **1. Prosedur dalam Pemrograman**

Prosedur, yang juga dikenal sebagai subprogram atau fungsi, adalah blok kode yang dirancang untuk melakukan tugas tertentu. Tujuan utama prosedur adalah untuk menyederhanakan program yang kompleks dengan membaginya menjadi unit-unit yang lebih kecil dan mudah dikelola. Dalam konteks bahasa pemrograman seperti Go dan pseudocode, prosedur memungkinkan program untuk menjalankan serangkaian instruksi setiap kali dipanggil, yang membantu mengurangi redundansi dan meningkatkan penggunaan kembali kode.

### **2. Deklarasi dan Pemanggilan Prosedur**

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 argument untuk parameter `n`.

### **3. Parameter dalam Prosedur**

Prosedur dapat menerima parameter yang berfungsi sebagai data input yang diperlukan untuk pemrosesan. Parameter ini dapat dikategorikan menjadi dua jenis:

- Parameter formal: Parameter yang dideklarasikan dalam prosedur dan berfungsi sebagai tempat untuk nilai yang akan diberikan selama pemanggilan.
- Parameter aktual: Nilai yang diberikan saat memanggil prosedur.
- Pass by Value: Hanya menyalin nilai, lalu perubahan di dalam prosedur tidak memengaruhi nilai aslinya.
- Pass by Reference: Menggunakan alamat memori, perubahan yang ada didalam prosedur memengaruhi nilai aslinya.

## SOURCE CODE:

```
guided1 > .\guided1.go > ...
1 package main
2
3 import "fmt"
4
5 func hitungGaji(nama string, gajiPokok Float64, jamLembur int) {
6     var totalGaji float64
7     totalGaji := gajiPokok + bonusLembur
8
9     fmt.Println("\n=== Slip Gaji ===")
10    fmt.Println("Nama Karyawan:", nama)
11    fmt.Printf("Gaji Pokok : Rp%.2f\n", gajiPokok)
12    fmt.Printf("Bonus Lembur: Rp%.2f (%d jam x Rp50,000)\n", bonusLembur, jamLembur)
13    fmt.Printf("Total Gaji : Rp%.2f\n", totalGaji)
14 }
15
16 func main() {
17     var nama string
18     var gajiPokok float64
19     var jamLembur int
20
21     fmt.Print("Masukkan Nama Karyawan: ")
22     fmt.Scanln(&nama)
23     fmt.Print("Masukkan Gaji Pokok: ")
24     fmt.Scanln(&gajiPokok)
25     fmt.Print("Masukkan Jam Lembur: ")
26     fmt.Scanln(&jamLembur)
27
28     hitungGaji(nama, gajiPokok, jamLembur)
29 }
30
```

## OUTPUT:

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4> go run "d:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4\guided1.go"
Masukkan Nama Karyawan: Nufail
Masukkan Gaji Pokok: 12000000
Masukkan Jam Lembur: 2

=== Slip Gaji ===
Nama Karyawan: Nufail
Gaji Pokok : Rp12000000.00
Bonus Lembur: Rp100000.00 (2 jam x Rp50,000)
Total Gaji : Rp12100000.00
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4>
```

## DEKSRIPSI:

Program ini menghitung gaji karyawan dengan memasukkan nama, gaji pokok, dan jam lembur. Bonus lembur dihitung berdasarkan Rp50.000 per jam, kemudian program menampilkan slip gaji yang mencakup gaji pokok, bonus lembur, dan total gaji.

## GUIDED 2

### SOURCE CODE:

```
guided2 > -go guided2.go > hitungRataRata
1  package main
2
3  import "fmt"
4
5  // Prosedur untuk menghitung rata-rata dan menentukan kelulusan
6  func hitungRataRata(nama string, nilai1, nilai2, nilai3 float64) {
7      ratarata := (nilai1 + nilai2 + nilai3) / 3
8      status := "Tidak Lulus"
9      if ratarata >= 60 {
10         status = "Lulus"
11     }
12
13     fmt.Println("\n=== Hasil Akademik ===")
14     fmt.Println("Nama Mahasiswa :", nama)
15     fmt.Printf("Nilai 1 : %.2f\n", nilai1)
16     fmt.Printf("Nilai 2 : %.2f\n", nilai2)
17     fmt.Printf("Nilai 3 : %.2f\n", nilai3)
18     fmt.Printf("Rata-rata : %.2f\n", ratarata)
19     fmt.Println("Status:", status)
20 }
21 func main() {
22     var nama string
23     var nilai1, nilai2, nilai3 float64
24
25     // Input dari pengguna
26     fmt.Print("Masukkan Nama Mahasiswa: ")
27     fmt.Scanln(&nama)
28
29     fmt.Print("Masukkan Nilai 1: ")
30     fmt.Scanln(&nilai1)
31
32     fmt.Print("Masukkan Nilai 2: ")
33     fmt.Scanln(&nilai2)
34
35     fmt.Print("Masukkan Nilai 3: ")
36     fmt.Scanln(&nilai3)
37
38     hitungRataRata(nama, nilai1, nilai2, nilai3)
39
40 }
41
```

## OUTPUT:

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4> go run "d:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4\main.go"
Masukkan Nama Mahasiswa: Nufail
Masukkan Nilai 1: 80
Masukkan Nilai 2: 77
Masukkan Nilai 3: 82

=== Hasil Akademik ===
Nama Mahasiswa : Nufail
Nilai 1 : 80.00
Nilai 2 : 77.00
Nilai 3 : 82.00
Rata-rata : 79.67
Status: Lulus
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4> |
```

## DEKSRIPSI:

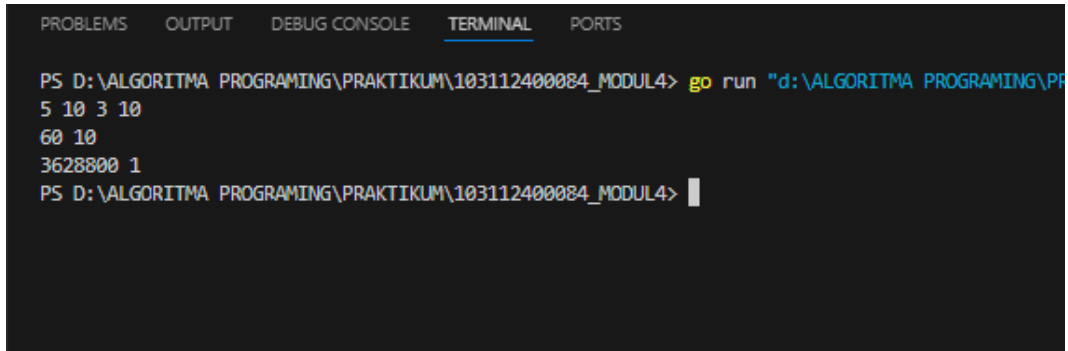
Program ini menghitung rata-rata nilai mahasiswa dari tiga ujian dan menentukan kelulusan. Jika rata-rata nilai  $\geq 60$ , statusnya "Lulus", jika tidak, "Tidak Lulus". Program menampilkan nama mahasiswa, nilai ujian, rata-rata, dan status kelulusan.

## II. UNGUIDED

### UNGUIDED 1

```
unguided1 > ~\go\src\unguided1.go > ...
1 //103112400084
2 //Nufail Alauddin Tsaqif
3 package main
4
5 import "fmt"
6
7 func faktorial(n int) int {
8     hasil := 1
9     for i := 1; i <= n; i++ {
10         hasil *= i
11     }
12     return hasil
13 }
14
15 func permutasi(n, r int) int {
16     if r > n {
17         return 0
18     }
19     return faktorial(n) / faktorial(n-r)
20 }
21
22 func kombinasi(n, r int) int {
23     if r > n {
24         return 0
25     }
26     return faktorial(n) / (faktorial(n-r) * faktorial(r))
27 }
28
29 func hitung(n1, n2 int) {
30     permut := permutasi(n1, n2)
31     komb := kombinasi(n1, n2)
32     fmt.Println(permut, komb)
33 }
34
35 func main() {
36     var a, b, c, d int
37     fmt.Scan(&a, &b, &c, &d)
38
39     if a >= c {
40         hitung(a, c)
41     } else {
42         hitung(c, a)
43     }
44
45     if b >= d {
46         hitung(b, d)
47     } else {
48         hitung(d, b)
49     }
50 }
51
```

## OUTPUT



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4> go run "d:\ALGORITMA PROGRAMING\PF
5 10 3 10
60 10
3628800 1
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4> |
```

## DEKSRIPSI

Program ini menghitung permutasi dan kombinasi menggunakan fungsi faktorial. Fungsi faktorial( $n$  int) menghitung nilai faktorial dari suatu angka  $n$  dengan cara mengalikan semua angka dari 1 hingga  $n$ . Fungsi permutasi( $n$ ,  $r$  int) menghitung permutasi dengan rumus faktorial dari  $n$  dibagi dengan faktorial dari  $n-r$ . Fungsi kombinasi( $n$ ,  $r$  int) menghitung kombinasi menggunakan rumus faktorial dari  $n$  dibagi dengan hasil perkalian faktorial dari  $n-r$  dan  $r$ . Di dalam fungsi hitung( $n1$ ,  $n2$  int), program memanggil fungsi permutasi atau kombinasi tergantung pada urutan input angka  $a$ ,  $b$ , dan  $c$ . Input diminta dari pengguna untuk tiga angka, dan program menentukan kombinasi atau permutasi yang tepat berdasarkan perbandingan angka-angka tersebut. Output menampilkan hasil perhitungan permutasi atau kombinasi sesuai dengan input yang diberikan.

## UNGUIDED 2

## SOURCE CODE

```
1 // package main
2 // @author: astuti
3 // @date: 2020-08-08
4 // @description: Program for calculating the sum of numbers
5 // @language: Go
6
7 package main
8
9 import "fmt"
10
11 func hitungSkor(small int, totalaspek1 *int, totalaspek2 *int) {
12     if small <= 100 {
13         *totalaspek1 +=
14         *totalaspek2 += small
15     }
16 }
17
18 func handlingScore(small string, totalaspek1, totalaspek2 int, nama2 string, totalaspek12, totalaspek22 int) (string, int, int) {
19     if totalaspek1 > totalaspek2 {
20         *totalaspek12 = totalaspek1 + totalaspek2
21         return nama2, totalaspek12, totalaspek22
22     }
23     return nama2, totalaspek12, totalaspek22
24 }
25
26 func main() {
27     var nama1, nama2 string
28     var small, small2 int
29     var totalaspek1, totalaspek2, totalaspek12, totalaspek22 int
30     var totalaspek1temp, totalaspek2temp int
31     var namatemp string
32
33     fmt.Scan(&nama1)
34     if nama1 != "Selesai" {
35         for i := 0; i < 5; i++ {
36             fmt.Scan(&small)
37             hitungSkor(small, &totalaspek1, &totalaspek2)
38         }
39     }
40
41     for {
42         fmt.Scan(&nama2)
43         if nama2 == "Selesai" {
44             break
45         }
46         totalaspek12, totalaspek22 = 0, 0
47         for i := 0; i < 5; i++ {
48             fmt.Scan(&small2)
49             hitungSkor(small2, &totalaspek12, &totalaspek22)
50         }
51         nama2temp, totalaspek1temp, totalaspek2temp = handlingScore(nama2, totalaspek1, totalaspek2, nama2, totalaspek12, totalaspek22)
52     }
53     fmt.Println(namatemp, totalaspek1temp, totalaspek2temp)
54 }
```

## OUTPUT

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4> go run "d:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4\main.go"
astuti 20 50 301 301 61 71 75 10
Bertha 25 47 301 26 50 60 65 21
Selesai
Bertha 7 294
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4>
```




## **DEKSRIPSI**

Program ini digunakan untuk membandingkan dua pemain berdasarkan skor dan waktu yang mereka habiskan. Fungsi `hitungSkor` menghitung skor dari setiap pemain dengan menambahkan nilai soal ke dalam `totalsoal` jika nilai soal lebih besar dari 300. Fungsi `bandingkanPemain` digunakan untuk membandingkan dua pemain berdasarkan total skor dan waktu yang mereka habiskan, dan mengembalikan pemain dengan skor lebih tinggi atau, jika skor sama, pemain dengan waktu lebih sedikit. Dalam fungsi `main`, program meminta input dari dua pemain, termasuk nama, skor, dan waktu yang mereka habiskan, dan kemudian membandingkan keduanya menggunakan fungsi `bandingkanPemain`. Hasilnya akan menampilkan nama pemain yang menang bersama dengan total skor dan waktu yang mereka habiskan.

## UNGUIDED 3

### SOURCE CODE

```
unguided3 >  unguided3.go > ...
1  //103112400084
2  //Nufail Alauddin Tsaqif
3  package main
4
5  import "fmt"
6
7  func cetakDeret(n int) {
8      for n != 1 {
9          fmt.Print(n, " ")
10         if n%2 == 0 {
11             n = n / 2
12         } else {
13             n = 3*n + 1
14         }
15     }
16     fmt.Print(n)
17 }
18
19 func main() {
20     var n int
21     fmt.Scan(&n)
22     if n < 1000000 {
23         cetakDeret(n)
24     } else {
25         fmt.Println("Input Tidak Valid")
26     }
27 }
28
```

### OUTPUT

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4> go run "d:\ALGORITMA PRO
22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\103112400084_MODUL4> 
```

## **DEKSRIPSI**

Program ini mencetak deret angka berdasarkan aturan yang mirip dengan masalah Collatz. Fungsi cetakDeret menerima input angka  $n$  dan kemudian menghasilkan deret dengan mengikuti aturan: jika  $n$  adalah bilangan genap, maka dibagi 2; jika  $n$  adalah bilangan ganjil, maka dikalikan 3 dan ditambah 1. Proses ini diulang hingga nilai  $n$  menjadi 1. Dalam fungsi main, program meminta input angka dari pengguna dan memastikan bahwa angka yang dimasukkan kurang dari 100.000. Jika input valid, fungsi cetakDeret dipanggil untuk mencetak deret tersebut; jika tidak, program mencetak pesan "Input Tidak Valid".

### **III. KESIMPULAN**

Kesimpulan dari laporan praktikum ini menunjukkan betapa pentingnya pemahaman dan penerapan prosedur dalam pemrograman. Prosedur dalam pemrograman berfungsi untuk membagi kode menjadi bagian-bagian yang lebih kecil dan mudah dikelola, sehingga memudahkan programmer dalam menulis, mengorganisir, dan memelihara program. Salah satu keuntungan utama dari prosedur adalah kemampuannya untuk mengurangi pengulangan kode dengan memungkinkan penggunaan kembali blok kode yang sudah dibuat untuk berbagai tujuan, yang tentunya meningkatkan efisiensi pengembangan perangkat lunak.

Selama praktikum, berbagai contoh program telah dipelajari, seperti penghitungan gaji karyawan, rata-rata nilai mahasiswa, permutasi dan kombinasi, serta perbandingan pemain dalam sebuah pertandingan. Masing-masing program ini menunjukkan bagaimana prosedur dan fungsi digunakan untuk menyelesaikan masalah secara sistematis dan efisien. Dalam program penghitungan gaji karyawan, misalnya, prosedur digunakan untuk memisahkan perhitungan gaji tetap, tunjangan, dan potongan, sehingga memudahkan pemeliharaan dan pemahaman kode. Begitu juga dalam program lainnya, prosedur dapat digunakan untuk menangani perhitungan matematis yang kompleks seperti permutasi dan kombinasi tanpa harus menulis ulang rumus yang sama berulang kali.

Selain itu, penggunaan prosedur memungkinkan pemrogram untuk menyusun program dengan lebih modular, artinya setiap bagian dari program dapat diuji dan diperbaiki secara terpisah. Ini mempermudah debugging dan pengembangan lebih lanjut di masa depan, karena setiap fungsi atau prosedur dapat diperbaiki tanpa mempengaruhi bagian lain dari program. Hal ini sangat penting dalam pengembangan perangkat lunak besar yang melibatkan banyak modul atau fungsi.

Secara keseluruhan, prosedur dan fungsi bukan hanya alat untuk menyelesaikan masalah pemrograman yang kompleks, tetapi juga memberikan struktur yang lebih jelas dalam pengembangan perangkat lunak. Pemahaman yang baik tentang penggunaan prosedur dapat meningkatkan kualitas kode, mempermudah pemeliharaan, dan memastikan bahwa perangkat lunak yang dikembangkan dapat terus berkembang dan disesuaikan dengan kebutuhan yang ada. Oleh karena itu, penguasaan teknik pemrograman berbasis prosedur ini sangat penting bagi programmer untuk menciptakan aplikasi yang efisien, terstruktur, dan mudah dikelola.

## **REFERENSI**

MODUL 4\_PROSEDUR ALGORITMA PEMOGRAMAN 2