

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 4
PROSEDUR**



Oleh:

BERTHA ADELA

103112400041

IF-12-01

**S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO**

2025

I. DASAR TEORI

Definisi Prosedur: 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.

Deklarasi Prosedur:

	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	}

Pemanggilan Prosedur:

	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	}

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.

```
func volumeTabung(jari_jari, tinggi int) float64 {  
    var luasAlas, volume float64  
  
    luasAlas = 3.14 * float64(jari_jari * jari_jari)  
    volume = luasAlas * tinggi  
    return volume  
}  
  
func main() {  
    var r, t int  
    var v1, v2 float64  
    r = 5; t = 10  
    v1 = volumeTabung(r, t)  
    v2 = volumeTabung(r, t) + volumeTabung(15, t)  
    fmt.Println(volumeTabung(14, 100))  
}
```

1. 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 `jari_jari`, `tinggi` 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`.

2. 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 `jari-jari` dan `tinggi`.

Berdasarkan alokasi memorinya:

1. Pass by Value

Nilai pada parameter aktual akan disalin ke variabel lokal (parameter formal) pada subprogram. Artinya parameter aktual dan formal dialokasikan di dalam memori komputer dengan alamat memori yang berbeda. Subprogram dapat menggunakan nilai pada parameter formal

tersebut untuk proses apapun, tetapi tidak dapat mengembalikan informasinya ke pemanggil melalui parameter aktual karena pemanggil tidak dapat mengakses memori yang digunakan oleh subprogram. Pass by value bisa digunakan baik oleh fungsi ataupun prosedur. Pada notasi pseudocode, secara semua parameter formal pada fungsi adalah pass by value, sedangkan pada prosedur diberi kata kunci in pada saat penulisan parameter formal. Sedangkan pada bahasa pemrograman Go sama seperti fungsi pada pseudocode, tidak terdapat kata kunci khusus untuk parameter formal fungsi dan prosedur.

2. Pass by Reference (Pointer)

Ketika parameter didefinisikan sebagai pass by reference, maka pada saat pemanggilan parameter formal akan berperan sebagai pointer yang menyimpan alamat memori dari parameter aktual. Sehingga perubahan nilai yang terjadi pada parameter formal tersebut akan berdampak pada parameter aktual. Artinya nilai terakhirnya akan dapat diketahui oleh si pemanggil setelah subprogram tersebut selesai dieksekusi. Pass by reference sebaiknya digunakan hanya untuk prosedur. Penulisan parameter pass by reference pada prosedur baik pseudocode dan Go menggunakan kata kunci atau identifier khusus. Pada pseudocode menggunakan kata kunci in/out, sedangkan pada bahasa Go diberi identifier asterik (*) sebelum tipe data di parameter formal yang menjadi pass by reference.

II. GUIDED

• GUIDED 1

Code:

```
SMT2 > Pertemuan4 > 103112400041_MODUL4 > 103112400041_Guided1.go > main
1  package main
2  import "fmt"
3  //Prosedur untuk menghitung gaji karyawan
4  func hitungGaji(nama string, gajiPokok float64, jamLembur int) {
5      bonusLembur:= float64(jamLembur) * 50000
6      totalGaji:= gajiPokok+bonusLembur
7
8      fmt.Println("\n=== Slip Gaji ===")
9      fmt.Println("Nama Karyawan :", nama)
10     fmt.Printf("Gaji Pokok : Rp%.2f\n", gajiPokok)
11     fmt.Printf("Bonus Lembur : Rp%.2f (%d jam x Rp50,000)\n", bonusLembur, jamLembur)
12     fmt.Printf("Total Gaji : Rp%.2f\n", totalGaji)
13 }

14
15 func main() {
16     var nama string
17     var gajiPokok float64
18     var jamLembur int
19
20     //Input dari pengguna
21     fmt.Print("Masukkan Nama Karyawan: ")
22     fmt.Scanln(&nama)
23
24     fmt.Print("Masukkan Gaji Pokok: ")
25     fmt.Scanln(&gajiPokok)
26
27     fmt.Print("Masukkan Jumlah Jam Lembur: ")
28     fmt.Scanln(&jamLembur)
29
30     //Memanggil prosedur dengan data dari pengguna
31     hitungGaji(nama, gajiPokok, jamLembur)
32 }
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDrive\Documents\golang\103112400041_Guided1.go"
Masukkan Nama Karyawan: Beta
Masukkan Gaji Pokok: 999999999
Masukkan Jumlah Jam Lembur: 10

=== Slip Gaji ===
Nama Karyawan : Beta
Gaji Pokok : Rp999999999.00
Bonus Lembur : Rp500000.00 (10 jam x Rp50,000)
Total Gaji : Rp1000499999.00
PS C:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program diatas berguna untuk menghitung total gaji karyawan.

• GUIDED 2

Code:

```
SMT2 > Pertemuan4 > 103112400041_MODUL4 > 103112400041_Guided2.go > main
1  package main
2  import "fmt"
3  //Prosedur untuk menghitung rata-rata dan menentukan kelulusan
4  func hitungRataRata(nama string, nilai1, nilai2, nilai3 float64) {
5      rataRata:=(nilai1+nilai2+nilai3) / 3
6      status:= "Tidak Lulus"
7      if rataRata >= 60 {
8          status="Lulus"
9      }
10
11      fmt.Println("\n=== Hasil Akademik ===")
12      fmt.Println("Nama Mahasiswa:", nama)
13      fmt.Printf("Nilai 1 : %.2f\n", nilai1)
14      fmt.Printf("Nilai 2 : %.2f\n", nilai2)
15      fmt.Printf("Nilai 3 : %.2f\n", nilai3)
16      fmt.Printf("Rata-rata : %.2f\n", rataRata)
17      fmt.Println("Status :", status)
18  }
19
20  func main() {
21      var nama string
22      var nilai1, nilai2, nilai3 float64
23
24      //Input dari pengguna
25      fmt.Print("Masukkan Nama Mahasiswa: ")
26      fmt.Scanln(&nama)
27
28      fmt.Print("Masukkan Nilai 1: ")
29      fmt.Scanln(&nilai1)
30
31      fmt.Print("Masukkan Nilai 2: ")
32      fmt.Scanln(&nilai2)
33
34      fmt.Print("Masukkan Nilai 3: ")
35      fmt.Scanln(&nilai3)
36
37      // Memanggil prosedur dengan data dari pengguna
38      hitungRataRata(nama, nilai1, nilai2, nilai3)
39  }
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDrive\Documents\golang\
DUL4\103112400041_Guided2.go"
Masukkan Nama Mahasiswa: Beta
Masukkan Nilai 1: 80
Masukkan Nilai 2: 85
Masukkan Nilai 3: 90

=== Hasil Akademik ===
Nama Mahasiswa: Beta
Nilai 1 : 80.00
Nilai 2 : 85.00
Nilai 3 : 90.00
Rata-rata : 85.00
Status : Lulus
PS C:\Users\levina\OneDrive\Documents\golang>
```


Penjelasan:

Program ini berguna untuk menyatakan apakah mahasiswa lulus(untuk nilai ≥ 60) setelah dicari rata rata dari ketiga nilai yang dimasukkan.

III. UNGUIDED

- UNGUIDED 1

Code:

```
MT2 > 103112400041_MODUL3 > -go 103112400041_UNGUIDED1.go >  Permutasi

1  //BERTHA ADELA
2  //103112400041
3  package main
4  import "fmt"
5  func main() {
6      var a, b, c, d int
7      fmt.Scan(&a,&b,&c,&d)
8
9      if a >= c && b >= d{
10         fmt.Print(Permutasi(a,c), " ")
11         fmt.Println(Kombinasi(a,c))
12         fmt.Print(Permutasi(b,d), " ")
13         fmt.Println(Kombinasi(b,d))
14     } else {
15         fmt.Print(Permutasi(c,a), " ")
16         fmt.Println(Kombinasi(c,a))
17         fmt.Print(Permutasi(d,b), " ")
18         fmt.Println(Kombinasi(d,b))
19     }
20 }

22 //BERTHA ADELA
23 // 103112400041
24 func Faktorial(n int) int {
25     hasil := 1
26     for i := 1; i<=n; i++ {
27         hasil *= i
28     }
29     return hasil
30 }
31 func Kombinasi(n,r int) int {
32     if r > n {
33         return 0
34     }
35     return Faktorial(n) / (Faktorial(r) * Faktorial(n-r))
36 }
37 func Permutasi(n,r int) int {
38     if r > n {
39         return 0
40     }
41     return Faktorial(n) / Faktorial(n-r)
42 }
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levi
400041_UNGUIDED1.go"
5 10 3 10
60 10
3628800 1
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levi
400041_UNGUIDED1.go"
8 0 2 0
56 28
1 1
```

Penjelasan:

Program ini berguna untuk mencari permutasi dan kombinasi dari 4 bilangan (a, b, c, d). dengan syarat $a \geq c$ dan $b \geq d$. Output baris pertama berupa permutasi dan kombinasi a terhadap c, sedangkan output baris kedua berupa permutasi dan kombinasi b terhadap d.

• UNGUIDED 2

Code:

```
SMT2 > Pertemuan4 > 103112400041_MODUL4 > -go 103112400041_Unguided2.go > hitungSkor
1 //BERTHA ADELA
2 //103112400041
3 package main
4 import "fmt"
5 func main() {
6     var nama, pemenang string
7     var skorTertinggi, total, soal, skorPeserta, soalTerjawab int
8     soal = 8
9     selesai := false
10    for selesai == false {
11        fmt.Scan(&nama)
12        if nama == "Selesai" {
13            selesai = true
14        } else {
15            skorPeserta, soalTerjawab = hitungSkor(soal)
16            total += skorPeserta
17            if skorPeserta > skorTertinggi {
18                skorTertinggi = skorPeserta
19                pemenang = nama
20            }
21        }
22    }
23    if pemenang != "" {
24        fmt.Print(pemenang, " ")
25        fmt.Print(soalTerjawab, " ")
26        fmt.Print(skorTertinggi, " ")
27    }
28 }

30 //BERTHA ADELA
31 //103112400041
32 func hitungSkor(soal int) (int, int) {
33     var totalSkor, totalSoal, skor int
34     soal = 8
35     for i := 1; i <= soal; i++ {
36         fmt.Scan(&skor)
37         if skor == 301 {
38             totalSkor -= 301
39             totalSoal -= 1
40         }
41         totalSkor += skor
42         totalSoal++
43     }
44     return totalSkor, totalSoal
45 }
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDrive\Documents\golang\103112400041_Unguided2.go"
Astuti 20 50 301 301 61 71 75 10
Bertha 25 47 301 26 50 60 65 21
Selesai
Bertha 7 294
PS C:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program ini mencari pemenang dari daftar peserta dan waktu tercepat. Output berupa pemenang, soal terjawab, serta total waktu (skor=waktu).

• UNGUIDED 3

Code:

```
SMT2 > Pertemuan4 > 103112400041_MODUL4 > -go 103112400041_Unguided3.go > cetakDeret
1 //BERTHA ADELA
2 //103112400041
3 package main
4 import "fmt"
5 func main() {
6     var n int
7     fmt.Scan(&n)
8     fmt.Print(cetakDeret(n))
9 }
10 func cetakDeret(n int) int {
11     stop := false
12     for stop == false {
13         fmt.Print(n, " ")
14         if n%2 == 0 {
15             n = n/2
16         } else {
17             n = 3*n+1
18         }
19         if n == 1 {
20             stop = true
21         }
22     }
23     return n
24 }
```

Output:

```
PS C:\Users\levina\OneDrive\Documents\golang> go run "c:\Users\levina\OneDrive\Documents\golang\103112400041_Unguided3.go"
22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS C:\Users\levina\OneDrive\Documents\golang>
```

Penjelasan:

Program ini akan mencetak setiap suku dari deret. 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 $3n+1$.

IV. KESIMPULAN

Parameter pada fungsi sebaiknya adalah pass by value, hal ini dikarenakan fungsi bisa mengembalikan (return) nilai ke pemanggil dan tidak memberikan efek langsung pada program, walaupun tidak menutup kemungkinan menggunakan pass by reference.

Penggunaan pass by reference sebaiknya pada prosedur karena prosedur tidak bisa mengembalikan nilai ke pemanggil. Dengan memanfaatkan pass by reference maka prosedur seolah-olah bisa mengirimkan nilai kepada si pemanggil.

REFERENSI

MODUL 4 PROSEDUR