

**LAPORAN PRAKTIKUM  
PEMROGRAMAN BERORIENTASI OBJEK**

**MODUL 4  
PROSEDUR**



Disusun Oleh:

NAMA : MARIA DWI A

NIM : 2311102228

KELAS : S1- 1F – 11 - 02

**S1 TEKNIK INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

# I. DASAR TEORI

## 1. 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 :

1. **Tidak ada** deklarasi tipe nilai yang dikembalikan, dan
2. **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, mulai, ...**

## 2. Deklarasi Procedure

Berikut ini adalah cara penulisan deklarasi prosedur pada notasi Pseudocode dan Golang.

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	}

Penulisan deklarasi ini berada di luar blok yang dari program utama atau **func main()** pada suatu program Go, dan bisa ditulis sebelum atau setelah dari blok utama tersebut.

Contoh deklarasi prosedur mencetak  $n$  nilai pertama dari deret Fibonacci.

	Notasi Algoritma
1	procedure cetakNFibo(in n : integer)
2	kamus
3	f1, f2, f3, i : integer
4	algoritma
5	f2 $\leftarrow$ 0
6	f3 $\leftarrow$ 1
7	for i $\leftarrow$ 1 to n do
8	output(f3)
9	f1 $\leftarrow$ f2
10	f2 $\leftarrow$ f3
11	f3 $\leftarrow$ 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	}

### 3. Cara Pemanggilan 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 lain.

Pemanggilan suatu prosedur cukup mudah, yaitu dengan hanya **menuliskan nama beserta parameter atau argument 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.

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	}

Dari contoh di atas terlihat bahwa cara oemanggilan dengan notasi pseudocode dan Golang adalah sama. Argumen yang digunakan untuk parameter n berupa integer (sesuai deklarasi) yang terdapat pada suatu varibael (cara pemanggilan #1) atau nilainya secara langsung (cara pemanggilan #2).

#### 4. Contoh Program dengan Procedur

Berikut ini adalah contoh prosedur pada suatu program lengkap.

Buatlah sebuah program beserta peosedur yang digunakan untuk menampilkan suatu pesan error, warning atau informasi berdasarkan masukan dari user.

**Masukan** terdiri dari sebuah bilangan bulat **flag** (0 s.d 2) dan sebuah string pesan **M**.

**Keluaran** berupa string pesan **M** beserta jenis pesannya, yaitu error, warning atau informasi berdasarkan nilai flag 0, 1 dan 2 secara berturut- turut.

```

1 package main
2 import "fmt"
3
4 func main(){
5     var bilangan int
6     var pesan string
7     fmt.Scan(&bilangan, &pesan)
8     cetakPesan(pesan,bilangan)
9 }
10
11 func cetakPesan(M string, flag int){
12     var jenis string = ""
13     if flag == 0 {
14         jenis = "error"
15     }else if flag == 1 {
16         jenis = "warning"
17     }else if flag == 2 {
18         jenis = "informasi"
19     }
20     fmt.Println(M,jenis)
21 }

```

Penulisan argumen pada parameter cetakPesan(pesan, bilangan) harus sesuai urutan tipe data pada func cetakPesan(M string, flag int), yaitu string kemudian integer.

## 5. 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 aktual.

```

1 func volumeTabung(jari_jari,tinggi int) float64 {
2     var luasAlas,volume float64
3
4     luasAlas = 3.14 * float64(jari_jari * jari_jari)
5     volume = luasAlas * tinggi
6     return volume
7 }
8
9 func main() {
10     var r,t int
11     var v1,v2 float64
12     r = 5; t = 10
13     v1 = volumeTabung(r,t)
14     v2 = volumeTabung(r,t) + volumeTabung(15,t)
15     fmt.Println(volumeTabung(14,100))
16 }

```

## 1. Parameter Formal

Parameter formal merupakan parameter yang ditulis pada saat deklarasi suatu subprogram, parameter ini berfungsi sebagai petunjuk bahwa argumen apa saja yang diperlukan pada saat pemanggilan subprogram.

Misalnya parameter **jari\_jari, tinggi** pada deklarasi **fungsi volumeTabung** adalah parameter formal(**jari\_jari, tinggi**).

Artinya ketika memanggil volumeTabung maka harus mempersiapkan dua integer (berapapun nilainya) sebagai jari\_jari dan tinggi.

## 2. Parameter Aktual

Sedangkan parameter actual adalah argument yang digunakan pada bagian parameter saat pemanggilan suatu subprogram.

Banyaknya argument dan tipe data yang terdapat pada parameter aktua;, yang menyatakan nilai yang kita berikan sebagai jari-jari dan tinggi.

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

### 1. Pass by Value

Nilai pada parameter actual disalin ke variabel lokal (parameter formal) pada subprogram. Artinya parameter aktual dan formal dialokasikan di dalam memori komputer dengan alamat memori yang berbeda-beda. 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.

## **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

Source Code + Screenshot hasil program beserta penjelasan

1. Buatlah sebuah program beserta fungsi yang digunakan untuk menghitung nilai factorial dan permutasi.

*Jawab :*

Source Code :

```
package main

import "fmt"

func faktorial(n int) int{
    if n == 0 || n == 1{
        return 1
    }

    result := 1
    for i := 2; i <= n; i++){
        result *= i
    }
    return result
}

// fungsi untuk menghitung permutasi
func permutation(n, r int) int {
    return faktorial(n) / faktorial(n-r)
}

func combination(n, r int) int {
    return faktorial(n)/(faktorial(r) * (n-r))
}

func main(){
```



```
// input 4 bilangan
var a, b, c, d int

fmt.Println("Masukkan bilangan a, b, c, d(dengan
spasi): ")

fmt.Scanf("%d %d %d %d", &a, &b, &c, &d)

// cek syarat a >= c dan b >= d
if a >= c && b >= d{
    //menghitung permutasi dan kombinasi a dan c
    permutasiAC := permutation(a, c)
    kombinasiAC := combination(a, c)

    // menghitung permutasi dan kombinasi b dan d

    permutasiBD := permutation(b, d)
    kombinasiBD := combination(b, d)

    // output hasil :
    fmt.Println("Permutasi (a,c) dan Kombinasi
(a,c), : ", permutasiAC, kombinasiAC)
    fmt.Println("Permutasi (b, d) dan Kombinasi
(b,d): ", permutasiBD, kombinasiBD)
} else {
    fmt.Println("Syarat a >= c dan b >= d tidak
terpenuhi.")
}
}
```

### Screenshoot Program

```
NG\PRAKTIKUM ALPRO MODUL 3\UNGUIDED\soal1_alpro.go"

Masukkan bilangan a, b, c, d (dengan spasi) : 8 0 2 0

Permutasi (a,c) dan kombinasi (a,c) : 56 28
Permutasi (b,d) dan kombinasi (b,d) : 1 1
```

### Deskripsi Program

Program diatas merupakan implemnetasi penggunaan function/fungsi yang digunakan untuk menghitung nilai factorial dan permutasi. Program akan meminta user untuk menginputkan dua bilangan bulat yaitu (a, dan b). Lalu program akan memeriksa apakah  $a \geq b$  jika kondisi benar maka program akan menghitung permutasi (a, b) dan jika kondisi ini tidak terpenuhi maka program akan menghitung permutasi (b,a). Kemudian program akan mencetak pada layar hasil perhitung dari permutasi kedua bilangan tersebut.

### III. UNGUIDED Soal Modul 4

1. Buatlah 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 parameter formal, yaitu nilai dari suku awal.

*Jawab :*

```
Source code
package main

import "fmt"

func cetakDeret (n int){
    fmt.Print(n)

    // loop hingga n != 1
    for n != 1 {
        if n%2 == 0 {
            // jika n genap, maka n dibagi dengan 2
            n = n/2
        } else {
            // jika n ganjil, n = 3n + 1
            n = 3*n + 1
        }
        fmt.Print(" ", n )
    }
}

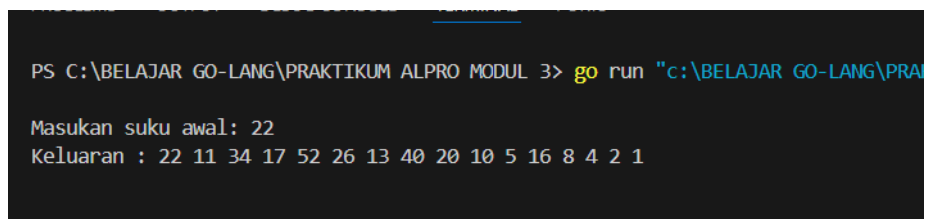
func main(){

    var n int
    fmt.Printf("\nMasukan suku awal: ")
    fmt.Scan(&n)
```

```
    fmt.Print("Keluaran : ")
    cetakDeret(n)

    fmt.Println("\n\n")
}
```

### Screenshoot Program



```
PS C:\BELAJAR GO-LANG\PRAKTIKUM ALPRO MODUL 3> go run "c:\BELAJAR GO-LANG\PRAKTIKUM ALPRO MODUL 3\main.go"
Masukan suku awal: 22
Keluaran : 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
```

### Deskripsi Program

Program tersebut merupakan program untuk mencetak deret bilangan dari sebuah bilangan yang diinputkan user dengan syarat : Jika bilangan  $n$  genap maka suku berikutnya adalah  $\frac{1}{2}n$ , tetapi jika ganjil maka suku berikutnya bernilai  $3n+1$ . Rumus yang sama digunakan terus menerus untuk mencari suku berikutnya. Dan deret akan berakhir ketika suku terakhir bernilai 1.