

LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL IV
PROSEDUR



Oleh:

NAMA : ARVAN MURBIYANTO

NIM : 2311102074

KELAS : IF 11 02

S1 TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024

I. DASAR TEORI

Definisi Prosedur

Prosedur adalah sekumpulan instruksi program yang dikemas menjadi satu perintah baru untuk memudahkan pengelolaan kode dalam program yang kompleks dan besar. Prosedur akan berdampak langsung pada program saat dipanggil dari program utama. Sebuah subprogram dikategorikan sebagai prosedur jika:

1. Tidak mendefinisikan tipe nilai yang akan dikembalikan, dan
2. Tidak menggunakan kata kunci `return` di dalam tubuh subprogram.

Prosedur berperan seperti instruksi dasar (seperti assignment) atau perintah dari pustaka (seperti `fmt.Scan` dan `fmt.Print`). Oleh karena itu, nama prosedur idealnya berupa kata kerja atau istilah yang menggambarkan suatu tindakan, seperti: cetak, hitungRerata, cariNilai, belok, mulai, dan sebagainya.

- Deklarasi prosedur

Berikut ini notasi pada pseudocode dan notasi pada golang

	Notasi algoritma
1	procedure procedure> (<params>)
2	kamus
3	{ <i>deklarasi</i>
4	...
5	algoritma
6	{ <i>badan algoritma procedure</i> }
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	}

- Cara Pemanggilan Prosedur

Memanggil sebuah prosedur sangat sederhana, yaitu hanya dengan menuliskan nama prosedur beserta parameter atau argumen yang diperlukan. Sebagai contoh, prosedur `cetakNFibo` di atas dapat dipanggil dengan menyebutkan namanya, kemudian memberikan sebuah variabel atau nilai integer tertentu sebagai argumen untuk parameter `n`.

	Notasi Algoritma
1	program contohprosedur
2	kamus
3	integer
4	algoritma
5	$x \leftarrow 5$
6	cetakNFibo(x)
7	cetakNFibo(100)
8	endprogram
	Notasi dalam bahasa Go
9	func main() {
10	var x int
11	5
12	cetakNFibo(x)
13	cetakNFibo(100)
14	}

Dari contoh di atas, dapat dilihat bahwa metode pemanggilan dengan notasi pseudocode dan GoLang memiliki kesamaan. Argumen yang digunakan untuk parameter `(n)` bertipe integer (sesuai dengan deklarasi) dapat berupa variabel (seperti pada cara pemanggilan #1) atau dapat juga diisi dengan nilainya secara langsung (seperti pada cara pemanggilan #2).

Parameter

Parameter adalah cara subprogram berkomunikasi dengan pemanggilnya melalui argumen yang disampaikan. Parameter dibagi menjadi dua jenis berdasarkan posisinya dalam program:

parameter formal dan *parameter aktual*.

1. Parameter Formal

Parameter formal adalah yang dideklarasikan saat subprogram dibuat. Mereka menentukan argumen yang diperlukan saat pemanggilan subprogram. Misalnya, pada fungsi `volumeTabung`, parameter `Jari_Jari` dan `Tinggi` adalah parameter formal, yang berarti kita harus menyediakan dua nilai (misalnya, jari-jari dan tinggi) saat memanggil fungsi tersebut.

2. Parameter Aktual

Parameter aktual adalah argumen yang diberikan saat memanggil subprogram. Jumlah dan tipe data dari parameter aktual harus sesuai dengan parameter formal. Sebagai contoh, dalam pemanggilan fungsi `volumeTabung`, argumen seperti `r`, `t`, atau angka seperti 15, 14, dan 100 adalah parameter aktual yang mewakili nilai jari-jari dan tinggi.

Catatan

Parameter dalam fungsi sebaiknya menggunakan *pass by value*, karena fungsi dapat mengembalikan nilai tanpa mempengaruhi program utama. Namun, *pass by reference* lebih cocok digunakan pada prosedur, karena prosedur tidak dapat mengembalikan nilai secara langsung. Dengan *pass by reference*, prosedur dapat mengubah nilai di pemanggilnya, seolah-olah mengembalikan hasil.

Untuk lebih jelas perhatikan contoh sebuah subprogram yang digunakan untuk menghitung persamaan berikut ini:

$$f(x,y) = 2x - \frac{y}{2} + 3$$

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>x dan y pada fungsi f1 dan prosedur f2 adalah pass by value,</p> <p>sedangkan variabel hasil pada prosedur f2 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(){ float64 var a,b int; var c fmt.Scan(&a,&b) f2(a,b,&c) output(c, f1(b,a)) endprogram </pre>	<p>x dan y pada fungsi f1 dan prosedur f2 adalah pass by value,</p> <p>sedangkan variabel hasil pada prosedur f2 adalah pass by reference.</p> <p>Karena variabel hasil adalah pointer to float64, maka untuk mengaksesnya menggunakan simbol bintang (*) pada variabelnya.</p> <p>Pada bahasa Go saat pemanggilan prosedur f2, maka parameter aktual untuk pass by reference harus diberi ampersand "&", contohnya &c</p>

II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

SOAL

NO 1.

Source code:

```
package main

import "fmt"

func main() {
    var bilangan int
    var pesan string

    fmt.Scan(&bilangan, &pesan)

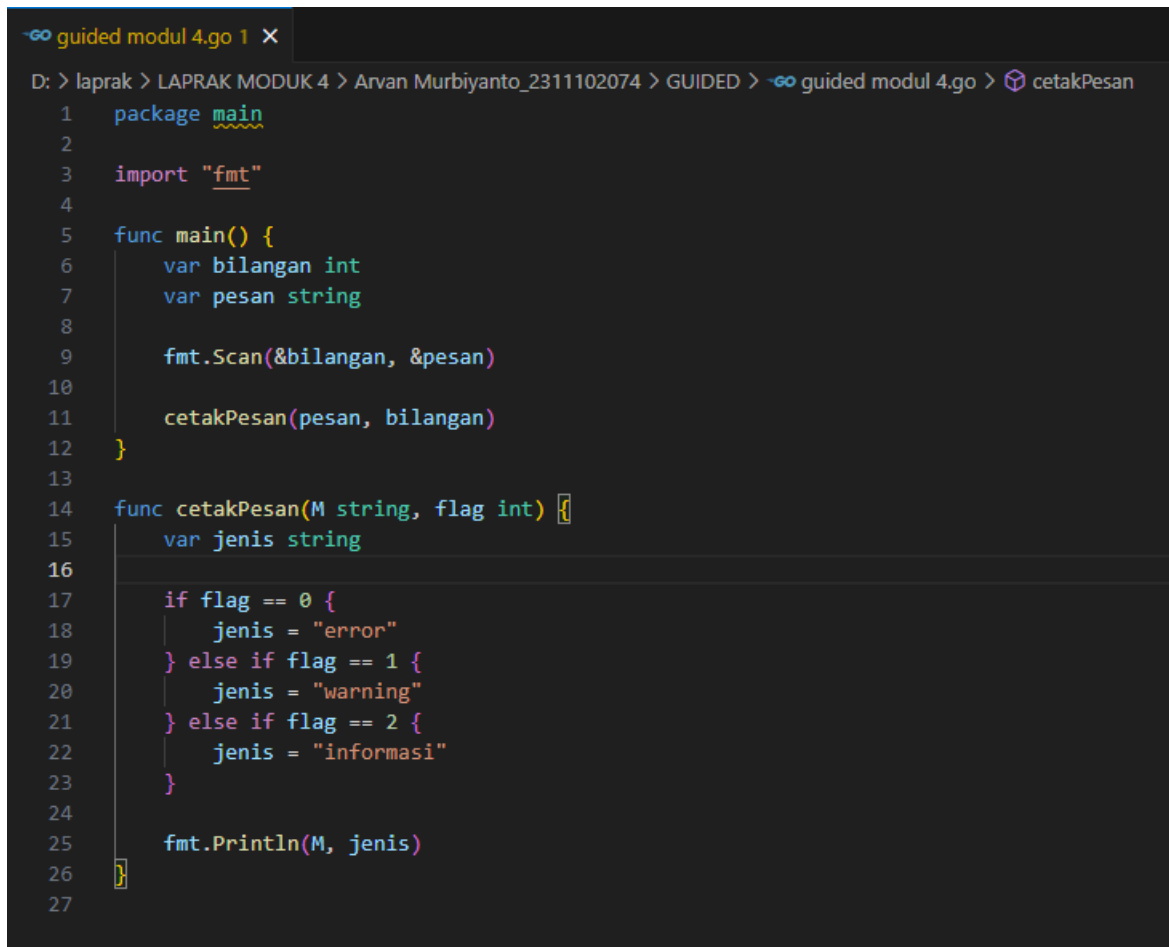
    cetakPesan(pesan, bilangan)
}

func cetakPesan(M string, flag int) {
    var jenis string

    if flag == 0 {
        jenis = "error"
    } else if flag == 1 {
        jenis = "warning"
    } else if flag == 2 {
        jenis = "informasi"
    }

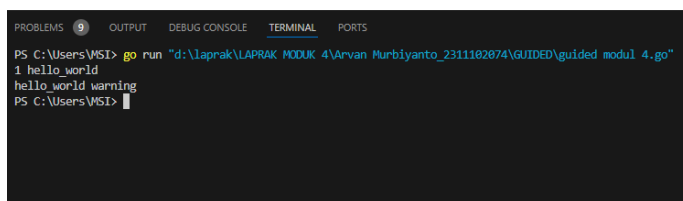
    fmt.Println(M, jenis)
}
```

Screenshoot :



```
guided modul 4.go 1 X
D: > laprak > LAPRAK MODUK 4 > Arvan Murbianto_2311102074 > GUIDED > -guided modul 4.go > cetakPesan
1 package main
2
3 import "fmt"
4
5 func main() {
6     var bilangan int
7     var pesan string
8
9     fmt.Scan(&bilangan, &pesan)
10
11     cetakPesan(pesan, bilangan)
12 }
13
14 func cetakPesan(M string, flag int) {
15     var jenis string
16
17     if flag == 0 {
18         jenis = "error"
19     } else if flag == 1 {
20         jenis = "warning"
21     } else if flag == 2 {
22         jenis = "informasi"
23     }
24
25     fmt.Println(M, jenis)
26 }
27
```

Output :



```
PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\WSI> go run "d:\laprak\LAPRAK MODUK 4\Arvan Murbianto_2311102074\GUIDED\guided modul 4.go"
1 hello_world
hello_world warning
PS C:\Users\WSI>
```

Deskripsi :

Program ini menerima dua input dari pengguna: sebuah bilangan integer dan pesan dalam bentuk string. Berdasarkan nilai integer yang dimasukkan, program akan menampilkan pesan yang digabungkan dengan jenis status yang sesuai. Fungsi cetakPesan digunakan untuk menentukan jenis status: jika bilangan bernilai 0, statusnya adalah "error"; jika 1, statusnya "warning"; dan jika 2, statusnya "informasi". Setelah itu, program mencetak pesan beserta status yang sesuai berdasarkan nilai integer yang dimasukkan oleh pengguna.

III. UNGUIDED

Source Code + Screenshot hasil program beserta penjelasan

SOAL 3

Skiena dan Revilla 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

Untuk suku awal sampai dengan 1000000, diketahui deret selalu mencapai suku dengan nilai 1.

Buat program sklena yang akan mencetak setiap suku dari deret yang dijelaskan di atas untuk nilai suku awal yang diberikan. Pencetakan deret harus dibuat dalam prosedur cetak Deret yang mempunyai 1 parameter formal, yaitu nilai dari suku awal.

prosedure 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 dan dipisahkan oleh sebuah spasi.

Source code :

```
guided modul 4 No packages found for open file D:\laprak\LAPRAK MODUK 4\Arvan Murbiyanto_2311102074\UNGUIDED\unguided modul 4.go. go list
D: > laprak > LAPRAK View Problem (Alt+F8) No quick fixes available
1 package main
2
3 import "fmt"
4
5 func cetakDeret(n int) {
6     for n != 1 {
7         fmt.Printf("%d ", n)
8         if n%2 == 0 {
9             n = n / 2
10        } else {
11            n = 3*n + 1
12        }
13    }
14    fmt.Println(n)
15 }
16
17 func main() {
18     var n int
19     fmt.Println("Masukkan nilai suku awal (integer positif kurang dari 1000000): ")
20     fmt.Scan(&n)
21
22     if n > 0 && n < 1000000 {
23         cetakDeret(n)
24     } else {
25         fmt.Println("Masukan tidak valid. Masukkan angka positif kurang dari 1000000.")
26     }
27 }
28
```

Output :

```
PROBLEMS 10 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\MSI> go run "d:\laprak\LAPRAK MODUK 4\Arvan Murbiyanto_2311102074\UNGUIDED\unguided modul 4.go"
Masukkan nilai suku awal (integer positif kurang dari 1000000):
22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS C:\Users\MSI>
```

Penjelasan :

Program ini menghasilkan deret angka berdasarkan aturan Skiena. Deret dimulai dari bilangan bulat positif yang dimasukkan oleh pengguna. Jika bilangan tersebut genap, maka dibagi dua, sedangkan jika ganjil, dikalikan tiga lalu ditambah satu. Proses berlanjut hingga angka dalam deret mencapai 1. Fungsi utama, "cetakderet", menjalankan algoritma tersebut dan mencetak setiap angka dalam deret secara berurutan. Program juga memeriksa bahwa input merupakan bilangan positif kurang dari 1.000.000 untuk memastikan validitas sebelum proses dijalankan.