LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK

MODUL IV

PROSEDUR

Oleh:

NAMA : ARNANDA SETYA NOSA PUTRA

NIM : 2311102180

KELAS : IF 11 02

S1 TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

I. DASAR TEORI

Definisi Prosedur

Prosedur adalah sekelompok instruksi program yang dikemas menjadi sebuah instruksi baru untuk menyederhanakan kode pada program yang kompleks dan besar. Prosedur akan memberikan efek langsung pada program saat dipanggil dari program utama. Suatu subprogram disebut prosedur jika:

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

Prosedur memiliki fungsi yang sama dengan instruksi dasar (seperti assignment) atau instruksi dari paket (seperti `fmt.Scan` dan `fmt.Print`). Oleh karena itu, nama prosedur sebaiknya berupa kata kerja atau sesuatu yang menunjukkan proses, misalnya: cetak, hitungRerata, cariNilai, belok, mulai, dll.

- Deklarasi prosedur

Berikut ini notasi pada pseudocode dan notasi pada golang

	Notasi algoritma
1	`procedure  procedure> (<params>)`
2	`kamus`
3	`    {deklarasi`
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	`}`

- 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
2
3
4
5
6
7
8
	`program contohprosedur`
	`kamus`
	`  integer`
	`algoritma`
	`  x ← 5`
	`  cetakNFibo(x)`
	`  cetakNFibo(100)`
	`endprogram`

	Notasi dalam bahasa Go

9
10
11
12
13
14
	`func main() {`
	`  var x int`
	`  5`
	`  cetakNFibo(x)`
	`  cetakNFibo(100)`
	`}`

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

adalah cara bagi suatu subprogram untuk berkomunikasi dengan pemanggilnya melalui argumen yang disampaikan melalui parameter yang telah dideklarasikan di dalam subprogram. Parameter dapat dikelompokkan menjadi dua jenis berdasarkan posisinya dalam program, yaitu *parameter formal* dan *parameter aktual*.

1. Parameter Formal

Parameter formal adalah parameter yang dituliskan saat mendeklarasikan suatu subprogram. Parameter ini berfungsi sebagai petunjuk mengenai argumen yang dibutuhkan saat pemanggilan subprogram. Contohnya, parameter Jari_Jari dan Tinggi dalam deklarasi fungsi volumeTabung adalah parameter formal (ditandai dengan teks berwarna merah). Ini berarti bahwa saat memanggil volumeTabung, kita harus menyiapkan dua integer (dengan nilai apapun) sebagai jari-jari dan tinggi.

2. Parameter Aktual

Parameter aktual, di sisi lain, adalah argumen yang digunakan dalam parameter saat pemanggilan subprogram. Jumlah dan tipe data pada parameter aktual harus sesuai dengan parameter formal. Sebagai contoh, argumen r, t, 15, 14, dan 100 dalam contoh kode di atas (ditandai dengan teks berwarna biru) adalah parameter aktual yang menyatakan nilai yang diberikan sebagai jari-jari dan tinggi.

Catatan:

Parameter dalam fungsi sebaiknya menggunakan *pass by value, karena fungsi dapat mengembalikan nilai ke pemanggil tanpa memberikan efek langsung pada program, meskipun tidak menutup kemungkinan untuk menggunakan **pass by reference. Penggunaan **pass by reference* lebih disarankan pada prosedur, karena prosedur tidak dapat mengembalikan nilai kepada pemanggil. Dengan menggunakan *pass by reference*, prosedur seolah-olah dapat mengirimkan nilai kepada pemanggil

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

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

Notasi Algoritma

```
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
```

x dan y pada fungsi f1 dan prosedur f2 adalah **pass by value**,

sedangkan variabel **hasil** pada prosedur f2 adalah **pass by reference**.

Untuk pemanggilan dengan notasi pseudocode masih sama dengan materi yang sudah dipelajari sebelumnya

Notasi dalam bahasa Go

```
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
}

                        float64
func main(){
    var a,b int; var c
    fmt.Scan(&a,&b)
    f2(a,b,&c)
    output(c, f1(b,a))
endprogram
```

x dan y pada fungsi f1 dan prosedur f2 adalah **pass by value**,

sedangkan variabel **hasil** pada prosedur f2 adalah **pass by reference**.

Karena variabel **hasil adalah pointer to float64**, maka untuk mengaksesnya menggunakan **simbol bintang** (*) pada variabelnya.

Pada bahasa Go saat pemanggilan prosedur f2, maka **parameter aktual** untuk **pass by reference** harus diberi **ampersand** "&", contohnya &c

II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

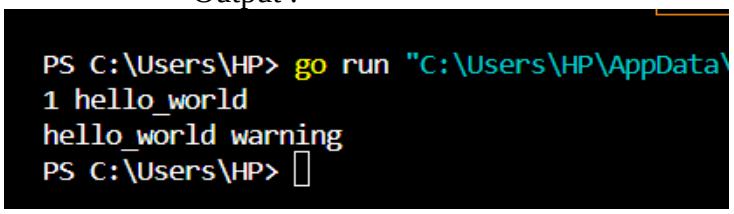
SOAL

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)
}
```

Output :



Penjelasan :

Program di atas membaca dua input dari pengguna, yaitu sebuah bilangan integer dan sebuah pesan string. Berdasarkan nilai integer yang dimasukkan, program kemudian menampilkan pesan yang dikombinasikan dengan jenis status yang sesuai. Fungsi `cetakPesan` digunakan untuk menentukan jenis status: jika bilangan 0, jenisnya adalah "error"; jika 1, jenisnya "warning"; dan jika 2, jenisnya "informasi". Program kemudian mencetak pesan yang diberikan bersama dengan jenis status yang sesuai, tergantung pada nilai integer yang diinputkan.

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 :

```go
package main

import "fmt"

func cetakDeret(n int) {
    for n != 1 {
        fmt.Printf("%d ", n)
        if n%2 == 0 {
            n = n / 2
        } else {
            n = 3*n + 1
        }
    }
    fmt.Println(n)
}

func main() {
    var n int
    fmt.Println("Masukkan nilai suku awal (integer positif kurang dari 1000000): ")
    fmt.Scan(&n)

    if n > 0 && n < 1000000 {
        cetakDeret(n)
    } else {
        fmt.Println("Masukan tidak valid. Masukkan angka positif kurang dari 1000000.")
    }
}
```

Output :

```
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\tempCodeRunnerFile.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\HP>
```

Penjelasan :

Program di atas mencetak deret angka sesuai dengan aturan Skiena. Deret dimulai dari bilangan bulat positif yang diberikan oleh pengguna. Jika bilangan tersebut genap, ia dibagi dua, sedangkan jika ganjil, dikalikan tiga dan ditambah satu. Proses ini berlanjut hingga nilai deret mencapai 1. Fungsi utama, `cetakDeret`, menjalankan logika ini dan mencetak setiap suku dari deret secara berurutan. Program juga memverifikasi bahwa input adalah bilangan positif yang lebih kecil dari 1.000.000, memastikan validitas sebelum menjalankan prosedur.