

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL IV
PROSEDUR**



Disusun Oleh :

Deshan Rafif Alfarisi / 2311102326

S1-IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi S.Kom., M.Kom.

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

4.1 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.

Kedua, keduanya prosedur sama seperti instruksi dasar yang sudah ada sebelumnya (**assignment**) dan/atau instruksi yang berasal dari paket (**fmt**), seperti **fmt.Scan** dan **fmt.Println**. Karena itu selalu pilih nama prosedur yang berbentuk **Kata Kerja** atau sesuatu yang merepresentasikan proses sebagai nama dari prosedur. Contoh: **cetak**, **hitungRata**, **carilahNilai**, **belajar**, **mulai**, ...

4.2 Deklarasi Procedure

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

| Notasi Algoritma | |
|------------------------|--|
| 1 | procedure <nama procedure> (<params>) kamus |
| 2 | { <i>deklarasi variabel lokal dari procedure</i> } |
| 3 | |
| 4 | algoritma |
| 5 | { <i>badan algoritma procedure</i> } |
| 6 | |
| 7 | endprocedure |
| 8 | |
| Notasi dalam bahasa Go | |
| 9 | func <nama procedure> (<params>) { |
| 10 | /* <i>deklarasi variabel lokal dari procedure</i> */ |
| 11 | |
| 12 | /* <i>badan algoritma procedure</i> */ |
| 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 program 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 ← 0 |
| 6 | f3 ← 1 |
| 7 | for i ← 1 to n do |
| 8 | output(f3) |
| 9 | f1 ← f2 |
| 10 | f2 ← f3 |
| 11 | f3 ← 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 | } |

Catatan: Kata kunci **in** pada contoh di atas akan dijelaskan pada materi parameter di modul 5 ini.

4.3 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 yang lain.

Pemanggilan suatu prosedur cukup mudah, yaitu dengan hanya menuliskan nama beserta parameter atau argumen yang diterima dari suatu prosedur. Sebagai contoh prosedur cetakNFibo

di atas dipanggil dengan menulis `nama` namanya, kemudian sebuah variabel atau nilai integer tertentu sebagai argumen untuk parameter `n`. Contoh:

| | Notasi Algoritma |
|----|---|
| 1 | program contohprosedur |
| 2 | kamus |
| 3 | <code>x : integer algoritma</code> |
| 4 | <code>x ← 5</code> |
| 5 | <code>cetakNFibo(x)</code> |
| 6 | <code>cetakNFibo(100)</code> <i>{cara pemanggilan #1}</i> |
| 7 | <i>{cara pemanggilan #2}</i> |
| 8 | endprogram |
| | Notasi dalam bahasa Go |
| 9 | <code>func main() {</code> |
| 10 | <code>var x int</code> |
| 11 | <code>x = 5</code> |
| 12 | <code>cetakNFibo(x)</code> <i>{cara pemanggilan #1}</i> |
| 13 | <code>cetakNFibo(700)</code> <i>{cara pemanggilan #2}</i> |
| 14 | <code>}</code> |

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

4.4 Contoh Program dengan Procedure

Berikut ini adalah contoh penulisan prosedur pada suatu program lengkap.

Buatlah sebuah program beserta prosedur 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
  import "fmt"
2
3 func main(){
4     var bilangan int
5     var pesan string
6
7     fmt.Scan(&bilangan, &pesan)
8     cetakPesan(pesan,bilangan)
9 }
10
11 func cetakPesan(M string, flag int){
12     var jenis string= ""
13
14     if flag -- 0 {
15         jenis = "error"
16     }else if flag -- 1 {
17         jenis = "warning"
18     }else if flag== 2 {
19         jenis = "informasi"
20     }
21
22     fmt.Println(M,jenis)
23 }

```

D:\DEV\DEMO>gobuildcontoh.go

D:\DEV\DEMO>contoh.exe

1 hello_world

hello_word error

Penulisan argumen pada parameter cetakPesan(pesan,bilangan) **harus sesuai urutan tipe data** pada **func**

cetakPesan(M **string**, flag **int**), yaitu string kemudian integer.

4.1 Parameter

Suatu subprogram yang dipanggil dapat berinteraksi dengan pemanggilnya melalui argumen yang diberikan melalui parameter yang didefinisikan 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) volume=
5     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 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 Rita 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 Rita 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 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 umum 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

asterisk (*) sebelum tipe data di parameter formal yang menjadi pass by reference.

Catatan:

- 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.

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

$$4. f(x,y) = 2x - \frac{y}{2} + 3$$

5.

| 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(){ var a,b int; var c float64 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

1. Soal Studi Case

Buatlah sebuah program beserta prosedur 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.

Sourcecode

```
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)
}
```

Screenshot Output

```
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3> go run "
c:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3\guided1.go"
3 1
1
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3> go run "
c:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3\guided1.go"
1 2
2 warning
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3> |
```

Deskripsi Program

Program ini untuk menerima dua input: sebuah bilangan bulat (flag) dan sebuah pesan teks. Bilangan bulat ini berfungsi sebagai penanda jenis pesan, yaitu apakah pesan tersebut merupakan pesan error (jika flag=0), warning (jika flag=1), atau informasi (jika flag=2). Setelah menerima input, program akan mencetak pesan tersebut beserta jenisnya. Jadi, program ini pada dasarnya adalah sebuah fungsi sederhana untuk menampilkan pesan dengan kategori yang berbeda-beda berdasarkan nilai flag yang diberikan.

2. Soal Studi Case

Mengirim Email ke pengguna yang haru terdaftar

Sourcecode

```
package main

import "fmt"

// Procedure untuk mengirim email pemberitahuan
func sendEmailNotification(email string) {
    fmt.Printf("Mengirim email ke %s: Pendaftaran berhasil.\n", email)
}

func main() {
    // Daftar email pengguna baru
    emails := []string{"user1@example.com",
"user2@example.com", "user3@example.com"}

    // Mengirim email pemberitahuan ke setiap pengguna
    fmt.Println("Mengirim email ke pengguna yang baru
terdaftar")
    for _, email := range emails {
        sendEmailNotification(email)
    }
}
```

Screenshoot Output

```
Mengirim email ke user@example.com: Pendaftaran berhasil.
PS C:\Users\Lenovo\Documents\file kuliah\Laparak Alpro 2\Laparak Alpro, Modul 3> go run
c:\Users\Lenovo\Documents\file kuliah\Laparak Alpro 2\Laparak Alpro, Modul 3\guided2.go
Mengirim email ke pengguna yang baru terdaftar
Mengirim email ke user1@example.com: Pendaftaran berhasil.
Mengirim email ke user2@example.com: Pendaftaran berhasil.
Mengirim email ke user3@example.com: Pendaftaran berhasil.
PS C:\Users\Lenovo\Documents\file kuliah\Laparak Alpro 2\Laparak Alpro, Modul 3> |
```

Deskripsi Program

Program ini mengirimkan email pemberitahuan kepada pengguna baru yang telah mendaftar. Program ini memiliki daftar email pengguna baru yang disimpan dalam sebuah array. Kemudian, program akan melakukan perulangan untuk setiap alamat email dalam daftar tersebut dan memanggil fungsi `sendEmailNotification` untuk mengirimkan email pemberitahuan pendaftaran berhasil. Fungsi `sendEmailNotification` sendiri sebenarnya hanya menampilkan pesan di layar untuk mensimulasikan pengiriman email, namun dalam aplikasi nyata, fungsi ini akan berisi kode untuk benar-benar mengirimkan email menggunakan layanan email seperti SMTP. Jadi, program ini secara sederhana mendemonstrasikan proses pengiriman email massal kepada sejumlah pengguna berdasarkan daftar email yang telah ditentukan.

3. Soal Studi Case

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

$$f(x,y) = 2x - 2^y + 3$$

Sourcecode

```
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) { //pass by reference
    *hasil = float64(2*x) - 0.5*float64(y) + 3.0 //pass by value
}

func main() {
    var a, b int
    var c float64

    //take input for a and b
    fmt.Print("Enter two integers: ")
    fmt.Scan(&a, &b)

    //call f2 to calculate and store the result in c
```

```

    f2(a, b, &c)

    //Print the result from f2
    fmt.Println("Result from f2 (stored in c): ", c)

    //call f1 and print the result
    resultF1 := f1(b, a) //(6 4)
    fmt.Println("Result from f1:", resultF1)
}

```

Screenshot Output

```

PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3> go run "
c:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3\guided3.go"
Enter two integers: 2 3
Result from f2 (stored in c):  5.5
Result from f1: 8
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3>

```

Deskripsi Program

Program ini untuk menghitung hasil dari dua fungsi matematika yang berbeda: f1 dan f2. Fungsi f1 menerima dua bilangan bulat sebagai input dan mengembalikan hasil perhitungan sebagai bilangan floating-point. Fungsi f2 juga menerima dua bilangan bulat sebagai input, tetapi menerima hasil perhitungan sebagai pointer ke bilangan floating-point. Hal ini memungkinkan f2 untuk memodifikasi nilai variabel yang dilewatkan sebagai parameter. Program kemudian meminta pengguna untuk memasukkan dua bilangan bulat, memanggil fungsi f2 untuk menghitung hasilnya dan menyimpannya dalam variabel c, lalu mencetak hasilnya. Selanjutnya, program memanggil fungsi f1 dengan parameter yang terbalik dan mencetak hasilnya.

III. UNGUIDED

1. Soal Studi Case

Minggu ini, mahasiswa Fakultas Informatika mendapatkan tugas dari mata Kuliah matematika diskrit untuk mempelajari kombinasi dan permutasi. Jonas salah seorang mahasiswa, iseng untuk mengimplementasikannya Re dalam suatu program. Oleh karena itu bersedia Ralian membantu Jonas? (tidak tentunya ya :p)

Masukan terdiri dari empat buah bilangan asli a, b, c , dan d yang dipisahkan oleh spasi, dengan syarat $a \geq c$ dan $b \geq d$.

Keluaran terdiri dari dua baris. Baris pertama adalah hasil permutasi dan kombinasi a terhadap c , sedangkan baris kedua adalah hasil permutasi dan kombinasi b terhadap d .

Catatan: permutasi (P) dan kombinasi (C) dari n terhadap r ($n \geq r$) dapat dihitung dengan menggunakan persamaan berikut!

$$P(n, r) = \frac{n!}{(n-r)!}, \text{ sedangkan } C(n, r) = \frac{n!}{r!(n-r)!}$$

Sourcecode

```
package main

import (
    "fmt"
)

// Fungsi untuk menghitung faktorial
func faktorial(n int) int {
    if n == 0 {
        return 1
    }
    result := 1
    for i := 1; i <= n; i++ {
        result *= i
    }
    return result
}

// Fungsi untuk menghitung permutasi P(n, r)
func permutasi(n, r int) int {
    return faktorial(n) / faktorial(n-r)
}
```

```
// Fungsi untuk menghitung kombinasi C(n, r)
func kombinasi(n, r int) int {
    return faktorial(n) / (faktorial(r) * faktorial(n-r))
}

func main() {
    // Input
    var a, b, c, d int
    fmt.Println("Masukkan 4 bilangan (a b c d):")
    fmt.Scan(&a, &b, &c, &d)

    // Mengecek syarat a >= c dan b >= d
    if a >= c && b >= d {
        // Menghitung permutasi dan kombinasi a terhadap c
        perm_a_c := permutasi(a, c)
        comb_a_c := kombinasi(a, c)

        // Menghitung permutasi dan kombinasi b terhadap d
        perm_b_d := permutasi(b, d)
        comb_b_d := kombinasi(b, d)

        // Output
        fmt.Printf("Permutasi(a, c): %d, Kombinasi(a, c): %d\n", perm_a_c, comb_a_c)
        fmt.Printf("Permutasi(b, d): %d, Kombinasi(b, d): %d\n", perm_b_d, comb_b_d)
    } else {
        fmt.Println("Input tidak valid. Pastikan a >= c dan b >= d.")
    }
}
```

Screenshoot Output

```
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3> go run "
c:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3\unguided2.go"
"
Masukkan 4 bilangan (a b c d):
5 6
2 3
Permutasi(a, c): 20, Kombinasi(a, c): 10
Permutasi(b, d): 120, Kombinasi(b, d): 20
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3> █
```

Deskripsi Program

Program ini menghitung permutasi dan kombinasi dari dua pasang bilangan bulat yang diberikan oleh pengguna. Program ini dimulai dengan meminta pengguna untuk memasukkan empat bilangan bulat (a, b, c, dan d). Kemudian, program akan memeriksa apakah a lebih besar sama dengan c dan b lebih besar sama dengan d. Jika syarat ini terpenuhi, maka program akan menghitung permutasi dan kombinasi untuk kedua pasang bilangan tersebut menggunakan fungsi-fungsi yang telah didefinisikan, yaitu fungsi faktorial, permutasi, dan kombinasi. Hasil perhitungan kemudian akan ditampilkan ke layar. Fungsi faktorial digunakan untuk menghitung faktorial dari sebuah bilangan, sedangkan fungsi permutasi dan kombinasi menggunakan hasil dari fungsi faktorial untuk menghitung permutasi dan kombinasi. Jika syarat awal tidak terpenuhi, maka program akan menampilkan pesan kesalahan.

2. Soal Studi Case

Kompetisi pemrograman tingkat nasional berlangsung ketat. Setiap peserta diberikan 8 soal yang harus dapat diselesaikan dalam waktu 5 jam saja. Peserta yang berhasil menyelesaikan soal paling banyak dalam waktu paling singkat adalah pemenangnya.

Buat program gema yang mencari pemenang dari daftar peserta yang diberikan. Program harus dibuat modular, yaitu dengan membuat prosedur hitungSkor yang mengembalikan total soal dan total skor yang dikerjakan oleh seorang peserta, melalui parameter formal. Pembacaan nama peserta dilakukan di program utama, sedangkan waktu pengerjaan dibaca di dalam prosedur.

prosedure hitungSkor (in/out soal, skor integer)

Setiap baris masukan dimulai dengan satu string nama peserta tersebut diikuti dengan adalah 8 integer yang menyatakan berapa lama (dalam menit) peserta tersebut menyelesaikan soal. Jika tidak berhasil atau tidak mengirimkan jawaban maka otomatis dianggap menyelesaikan. dalam waktu 5 jam 1 menit (301 menit).

Satu baris keluaran berisi nama pemenang, jumlah soal yang diselesaikan, dan nilai yang diperoleh. Nilai adalah total waktu yang dibutuhkan untuk menyelesaikan soal yang berhasil diselesaikan. Astuti menyelesaikan 6 soal dalam waktu 287 menit, sedangkan Bertha 7 soal dalam waktu 294 menit. Karena Bertha menyelesaikan lebih banyak, maka Bertha menang. Jika keduanya menyelesaikan sama banyak, maka pemenang adalah yang menyelesaikan dengan total waktu paling kecil.

Sourcecode

```
package main

import (
    "fmt"
)

// Prosedur untuk menghitung jumlah soal yang diselesaikan
// dan total skor
func hitungSkor(waktu [8]int, soal *int, skor *int) {
    *soal = 0
    *skor = 0
    for i := 0; i < 8; i++ {
        if waktu[i] <= 300 { // Jika soal diselesaikan dalam
            waktu <= 300 menit
            *soal++
            *skor += waktu[i]
        } else {
            *skor += 301 // Jika soal tidak diselesaikan,
            // tambahkan 301 menit
        }
    }
}

func main() {
    var jumlahPeserta int
    fmt.Print("Masukkan jumlah peserta: ")
    fmt.Scan(&jumlahPeserta)

    // Variabel untuk menyimpan informasi pemenang
    var pemenangNama string
    var pemenangSoal, pemenangSkor int

    // Loop untuk setiap peserta
    for i := 0; i < jumlahPeserta; i++ {
        // Membaca nama peserta
        var namaPeserta string
        fmt.Print("Masukkan nama peserta: ")
        fmt.Scan(&namaPeserta)

        // Membaca waktu pengerjaan 8 soal
        var waktu [8]int
        fmt.Print("Masukkan waktu pengerjaan 8 soal (dalam
        menit): ")
    }
```

```

        for j := 0; j < 8; j++ {
            fmt.Scan(&waktu[j])
        }

        // Variabel untuk menyimpan hasil skor peserta saat
        ini
        var soal, skor int

        // Menghitung jumlah soal yang diselesaikan dan
        total skor
        hitungSkor(waktu, &soal, &skor)

        // Menentukan pemenang
        if soal > pemenangSoal || (soal == pemenangSoal &&
        skor < pemenangSkor) {
            pemenangNama = namaPeserta
            pemenangSoal = soal
            pemenangSkor = skor
        }
    }

    // Output pemenang
    fmt.Printf("Pemenangnya adalah %s yang menyelesaikan %d
    soal dengan total waktu %d menit.\n", pemenangNama,
    pemenangSoal, pemenangSkor)
}

```

Screenshoot Output

```

PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3> go run "
c:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3\unguided2.go"
Masukkan jumlah peserta: 2
Masukkan nama peserta: tuti
Masukkan waktu pengerjaan 8 soal (dalam menit): 20 50 60 20 20 20 20 20
Masukkan nama peserta: eto
Masukkan waktu pengerjaan 8 soal (dalam menit): 60 80 60 40 30 20 10 50
Pemenangnya adalah tuti yang menyelesaikan 8 soal dengan total waktu 230 menit.
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3> █

```

Deskripsi Program

Program ini untuk menentukan pemenang dalam sebuah kompetisi pemrograman. Program ini akan meminta pengguna untuk memasukkan jumlah peserta, lalu untuk setiap peserta, akan meminta nama dan waktu yang dibutuhkan untuk menyelesaikan masing-masing dari 8 soal. Program kemudian akan menghitung jumlah soal yang berhasil diselesaikan oleh setiap peserta dan total waktu yang digunakan. Peserta dengan jumlah soal terbanyak atau dengan jumlah soal sama

tetapi waktu tercepat akan dinyatakan sebagai pemenang. Program ini menggunakan fungsi `hitungSkor` untuk menghitung jumlah soal yang diselesaikan dan total skor untuk setiap peserta, serta menggunakan perbandingan untuk menentukan pemenang.

3. Soal Studi Case

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.

Sourcecode

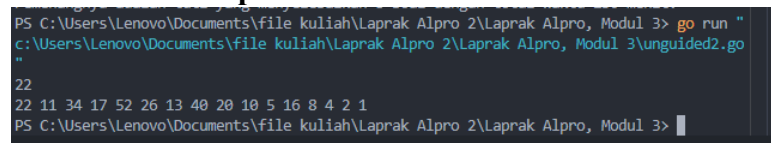
```
package main

import "fmt"

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

```
func main() {  
    var n int  
    fmt.Scan(&n)  
    cetakDeret(n)  
}
```

Screenshoot Output



```
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3> go run "  
c:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3\unguided2.go  
"  
22  
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1  
PS C:\Users\Lenovo\Documents\file kuliah\Laprak Alpro 2\Laprak Alpro, Modul 3>
```

Deskripsi Program

Program ini menghasilkan deret bilangan yang mengikuti aturan tertentu, yang dikenal sebagai dugaan Collatz. Aturannya sederhana: jika bilangan saat ini genap, bagi dengan 2; jika ganjil, kalikan dengan 3 lalu tambahkan 1. Proses ini diulang terus-menerus sampai bilangannya menjadi 1. Program ini akan meminta pengguna untuk memasukkan bilangan awal (n), lalu mencetak deret bilangan yang dihasilkan mulai dari bilangan awal tersebut hingga mencapai 1.