

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL IV
PROSEDUR**



Disusun Oleh :

Siti Madina Halim Siregar / 2311102243

S1IF-11-05

Dosen Pengampu :

Arif Amrulloh, S.Kom., M.Kom.

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Prosedur (Procedure) Prosedur merupakan sub program yang melakukan satu atau lebih proses tanpa mengembalikan nilai dari hasil dari proses yang dijalankan. Namun, pada prosedur diperbolehkan mencetak hasil luaran yang dihasilkan dari proses yang dijalankan. Contohnya, mencetak hasil dari proses aritmatika.

Ketika sebuah prosedur dipanggil, maka pada dasarnya dapat melakukan proses pertukaran data antara program utama dan sub program. Proses pertukaran ini dapat dilakukan dengan penggunaan parameter. Pada prosedur terdapat parameter yaitu sebagai berikut:

1. Parameter aktual Parameter aktual merupakan parameter yang disertakan pada saat prosedur dipanggil untuk dijalankan pada program utama, atau dapat sering jenis ini sebagai argument.
2. Parameter formal Parameter formal merupakan parameter yang dituliskan pada saat melakukan pendefinisian sebuah prosedur. Parameter ini ditulis pada bagian kode sub program untuk menerima nilai dari parameter aktual pada program utama. Terdapat tiga jenis parameter formal yang dapat digunakan pada prosedur, diantaranya:
 - a. Parameter masukan (input) Parameter masukan merupakan parameter yang menerima nilai dari parameter aktual untuk dilakukan pemrosesan pada prosedur.
 - b. Parameter keluaran (output) Parameter keluaran merupakan parameter yang digunakan untuk menampung hasil dari pemrosesan dari sebuah prosedur, selanjutnya hasilnya dikirimkan ke program utama program.
 - c. Parameter masukan dan keluaran (input / output) Parameter yang menerima nilai dari parameter aktual untuk diproses dalam prosedur kemudian nilai yang dihasilkan dari pemrosesan tersebut ditampung ke dalam parameter output. parameter output yang dihasilkan oleh prosedur dapat dimanfaatkan pada instruksi kode berikutnya

Pengertian Golang

Golang adalah bahasa pemrograman open-source yang memiliki sintaksis sederhana namun kuat, memungkinkan pengembang untuk menulis kode dengan cepat dan efisien. Bahasa ini menggunakan tipe data statis dan menghasilkan kode biner yang dikompilasi, sehingga dapat berjalan dengan cepat dan efisien.

II. GUIDED

Soal Studi Case

Menghitung Faktorial dan Permutasi menggunakan prosedur

Sourcecode

```
package main

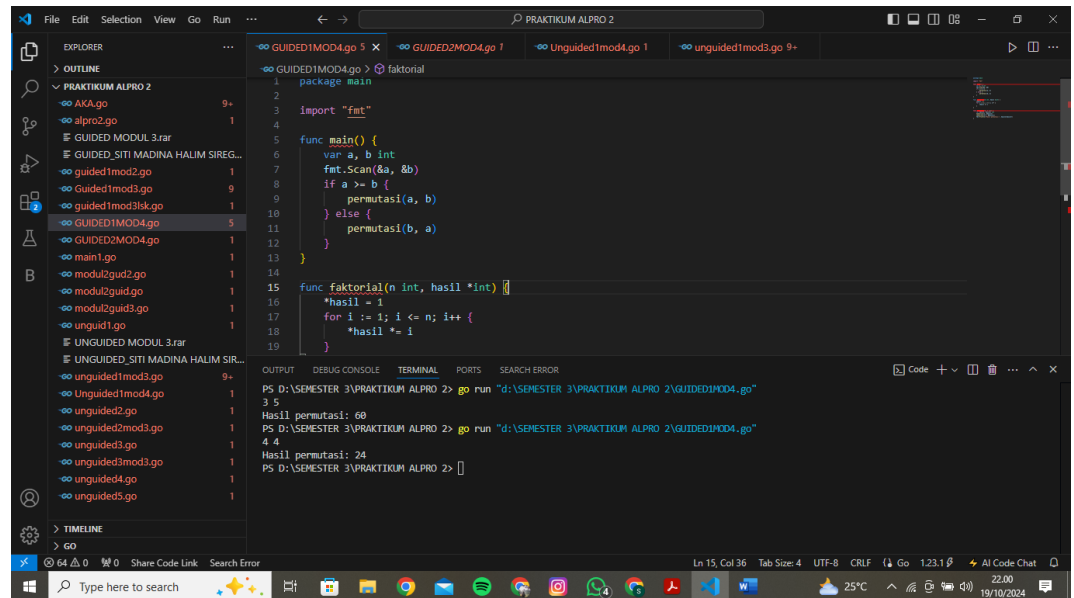
import "fmt"

func main() {
    var a, b int
    fmt.Scan(&a, &b)
    if a >= b {
        permutasi(a, b)
    } else {
        permutasi(b, a)
    }
}

func faktorial(n int, hasil *int) {
    *hasil = 1
    for i := 1; i <= n; i++ {
        *hasil *= i
    }
}

func permutasi(n, r int) {
    var hasiln, hasilnr int
    faktorial(n, &hasiln)
    faktorial(n-r, &hasilnr)
    fmt.Println("Hasil permutasi:", hasiln/hasilnr)
}
```

Screenshoot Output



The screenshot shows a Go IDE with a project named "PRAKTIKUM ALPRO 2". The Explorer panel on the left lists several files, including "GUIDED1MOD4.go" which is selected. The main editor displays the code for "GUIDED1MOD4.go", which implements a factorial function and a permutation function. The code is as follows:

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var a, b int
7     fmt.Scan(&a, &b)
8     if a >= b {
9         permutasi(a, b)
10    } else {
11        permutasi(b, a)
12    }
13 }
14
15 func faktorial(n int, hasil *int) {
16     *hasil = 1
17     for i := 1; i <= n; i++ {
18         *hasil *= i
19     }
20 }
```

The Output panel at the bottom shows the execution results for two test cases:

```
PS D:\SEMESTER 3\PRAKTIKUM ALPRO 2> go run "d:\SEMESTER 3\PRAKTIKUM ALPRO 2\GUIDED1MOD4.go"
3 5
Hasil permutasi: 60
PS D:\SEMESTER 3\PRAKTIKUM ALPRO 2> go run "d:\SEMESTER 3\PRAKTIKUM ALPRO 2\GUIDED1MOD4.go"
4 4
Hasil permutasi: 24
PS D:\SEMESTER 3\PRAKTIKUM ALPRO 2>
```

Deskripsi Program

- Mengambil dua bilangan bulat, a dan b, dari pengguna.
- Memeriksa mana dari a atau b yang lebih besar dan memanggil fungsi permutasi dengan argumen yang sesuai.
- Menghitung faktorial dari bilangan bulat n dan menyimpan hasilnya di variabel hasil.
- Program menampilkan hasil permutasi dalam format: Hasil permutasi: [nilai].
- Jika pengguna memasukkan p(3,5) menampilkan hasil permutasi 60 dengan rumus

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

Soal Studi Case

Menghitung luas dan keliling persegi dengan prosedur

Sourcecode

```
package main

import "fmt"

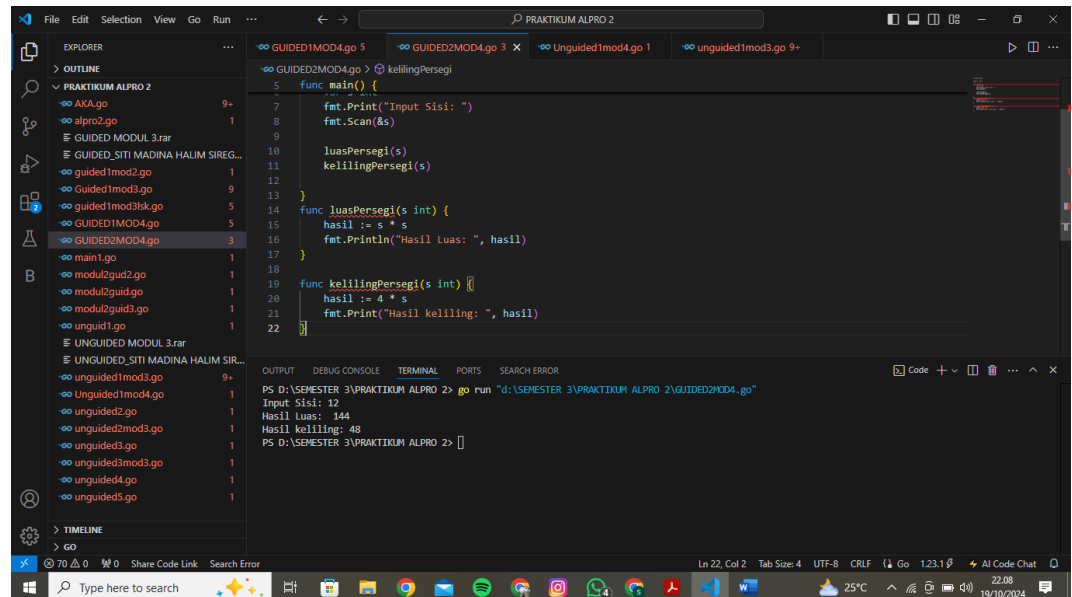
func main() {
    var s int
    fmt.Print("Input Sisi: ")
    fmt.Scan(&s)

    luasPersegi(s)
    kelilingPersegi(s)
}

func luasPersegi(s int) {
    hasil := s * s
    fmt.Println("Hasil Luas: ", hasil)
}

func kelilingPersegi(s int) {
    hasil := 4 * s
    fmt.Print("Hasil keliling: ", hasil)
}
```

Screenshoot Output



The screenshot shows a Go IDE with a project named "PRAKTIKUM ALPRO 2". The Explorer panel on the left lists several files, including "GUIDED2MOD4.go" which is currently selected. The main editor displays the code for "GUIDED2MOD4.go", which defines a `main` function and two helper functions: `luasPersegi` (area) and `kelilingPersegi` (perimeter). The `main` function prompts the user for the side length `s`, calls the helper functions, and prints the results. The Output panel at the bottom shows the execution results: "Input Sisi: 12", "Hasil Luas: 144", and "Hasil keliling: 48".

```
5 func main() {
6     // Prompt user for input
7     fmt.Print("Input Sisi: ")
8     fmt.Scan(&s)
9
10    // Calculate area and perimeter
11    luasPersegi(s)
12    kelilingPersegi(s)
13 }
14
15 func luasPersegi(s int) {
16     hasil := s * s
17     fmt.Println("Hasil Luas: ", hasil)
18 }
19
20 func kelilingPersegi(s int) {
21     hasil := 4 * s
22     fmt.Print("Hasil keliling: ", hasil)
23 }
```

OUTPUT

```
PS D:\SEMESTER 3\PRAKTIKUM ALPRO 2> go run "d:\SEMESTER 3\PRAKTIKUM ALPRO 2\GUIDED2MOD4.go"
Input Sisi: 12
Hasil Luas: 144
Hasil keliling: 48
PS D:\SEMESTER 3\PRAKTIKUM ALPRO 2>
```

Deskripsi Program

- Program meminta pengguna untuk memasukkan panjang sisi persegi (s).
- Menghitung luas persegi dengan rumus $\text{Luas} = s \times s$.
- mencetak hasil luas
- Menghitung keliling persegi dengan rumus $\text{Keliling} = 4 \times s$.
- mencetak hasil keliling

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 ke dalam suatu program. Oleh karena itu bersediakah kalian membantu Jonas? (tidak tentunya ya :p)

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

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!

$P(n,r) = n! / (n-r)!$, sedangkan $C(n,r) = n! / r!(n-r)!$

```
procedure factorial(in n: integer, in/out hasil:integer)
{I.S. terdefinisi bilangan bulat positif n
 F.S. hasil berisi nilai faktorial dari n}

procedure permutation(in n,r : integer, in/out hasil:integer)
{I.S. terdefinisi bilangan bulat positif n dan r, dan n >= r
 F.S. hasil berisi nilai dari n permutasi r}

procedure combination(in n,r : integer, in/out hasil:integer)
{I.S. terdefinisi bilangan bulat positif n dan r, dan n >= r
 F.S. hasil berisi nilai dari n kombinasi r}
```

Sourcecode

```
package main

import "fmt"

func factorial(n int, hasil *int) {
    *hasil = 1
    for i := 1; i <= n; i++ {
        *hasil *= i
    }
}

func permutation(n, r int, hasil *int) {
    var fn, fnr int
    factorial(n, &fn)
    factorial(n-r, &fnr)
    *hasil = fn / fnr
}

func combination(n, r int, hasil *int) {
    var fn, fr, fnr int
    factorial(n, &fn)
    factorial(r, &fr)
    factorial(n-r, &fnr)
    *hasil = fn / (fr * fnr)
}

func main() {
    var a, b, c, d int
    fmt.Scan(&a, &b, &c, &d)

    if a*c >= d && b >= d {
        var permAC, combAC, permBD, combBD int
        permutation(a, c, &permAC)
        combination(a, c, &combAC)
        permutation(b, d, &permBD)
        combination(b, d, &combBD)

        fmt.Println(permAC, combAC)
        fmt.Println(permBD, combBD)
    } else {
        fmt.Println("Syarat tidak terpenuhi.")
    }
}
```


Screenshoot Output

```
func combination(n, r int, hasil *int) {  
    // ...  
}  
  
func main() {  
    var a, b, c, d int  
    fmt.Scan(&a, &b, &c, &d)  
  
    if a*c >= d && b >= d {  
        var permAC, combAC, permBD, combBD int  
        permutation(a, c, &permAC)  
        combination(a, c, &combAC)  
        permutation(b, d, &permBD)  
        combination(b, d, &combBD)  
  
        fmt.Println(permAC, combAC)  
        fmt.Println(permBD, combBD)  
    } else {  
        fmt.Println("Syarat tidak terpenuhi.")  
    }  
}
```

OUTPUT

```
PS D:\SEMESTER 3\PRAKTIKUM ALPRO 2> go run "d:\SEMESTER 3\PRAKTIKUM ALPRO 2\Unguided1mod4.go"  
8 0 2 0  
56 28  
1 1  
2 5 6 9  
Syarat tidak terpenuhi.  
PS D:\SEMESTER 3\PRAKTIKUM ALPRO 2>
```

Deskripsi Program

- Menghitung faktorial dari bilangan bulat n dan menyimpan hasilnya di variabel yang dipassing sebagai pointer (hasil).
- Menghitung permutasi dari n dan r dengan menggunakan fungsi faktorial. Hasilnya disimpan dalam variabel yang dipassing sebagai pointer (hasil).
- Menghitung kombinasi dari n dan r, juga menggunakan fungsi faktorial. Hasilnya disimpan di variabel pointer yang diberikan.
- Membaca empat bilangan bulat dari input (a, b, c, d).
- Memeriksa apakah syarat ($a \cdot c \geq d$ dan $b \geq d$) terpenuhi.
- Jika syarat terpenuhi, menghitung dan mencetak hasil permutasi dan kombinasi untuk pasangan (a, c) dan (b, d).
- Jika syarat tidak terpenuhi, mencetak pesan "Syarat tidak terpenuhi."

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 tau 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.

No	Masukan	Keluaran
1	Astuti 20 50 301 301 61 71 75 10 Bertha 25 47 301 26 50 60 65 21 Selesai	Bertha 7 294

Keterangan:

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"

func hitungSkor(waktu []int, soal *int, skor *int) {
    for _, w := range waktu {
        if w < 301 {
            *soal++
            *skor += w
        }
    }
}

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

    pemenang, maxSoal, minSkor := "", -1, 99999
```

```

    for i := 0; i < peserta; i++ {
        var nama string
        var waktu [8]int
        fmt.Print("Input nama dan waktu: ")
        fmt.Scan(&nama, &waktu[0], &waktu[1], &waktu[2],
&waktu[3], &waktu[4], &waktu[5], &waktu[6], &waktu[7])

        var totalSoal, totalSkor int
        hitungSkor(waktu[:], &totalSoal, &totalSkor)

        if totalSoal > maxSoal || (totalSoal == maxSoal &&
totalSkor < minSkor) {
            pemenang, maxSoal, minSkor = nama, totalSoal,
totalSkor
        }
    }

    fmt.Printf("Pemenang: %s\nJumlah soal: %d\nTotal waktu:
%d\n", pemenang, maxSoal, minSkor)
}

```

Screenshoot Output

The screenshot shows a Go IDE with the following components:

- EXPLORER:** A list of files including `AKA.go`, `alpro2.go`, `GUIDED MODUL 3.rar`, `GUIDED_SITI MADINA HALIM SIREG...`, `guided1mod2.go`, `Guided1mod3.go`, `guided1mod3sk.go`, `GUIDED1MOD4.go`, `GUIDED2MOD4.go`, `main1.go`, `modul2guid2.go`, `modul2guid3.go`, `unguid1.go`, `UNGUIDED MODUL 3.rar`, `UNGUIDED_SITI MADINA HALIM SIR...`, `unguided1mod3.go`, `Unguided1mod4.go`, `unguided2.go`, `unguided2mod3.go`, `unguided2mod4.go`, `unguided3.go`, `unguided3mod3.go`, `unguided4.go`, and `unguided5.go`.
- OUTLINE:** A tree view of the code structure.
- EDITOR:** The main code editor showing the `main` function and `hitungSkor` function. The code is as follows:

```

func hitungSkor(waktu []int, soal *int, skor *int) {
    for _, w := range waktu {
        if w < 301 {
            *soal++
            *skor += w
        }
    }
}

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

    pemenang, maxSoal, minSkor := "", -1, 99999

    for i := 0; i < peserta; i++ {
        var nama string
        var waktu [8]int

```
- OUTPUT:** The terminal output showing the execution of the program:

```

PS D:\SEMESTER 3\PRATIUM ALPRO 2> go run "d:\SEMESTER 3\PRATIUM ALPRO 2\unguided2mod4.go"
Input jumlah peserta: 2
Input nama dan waktu: Astuti 20 50 301 301 61 71 75 10
Input nama dan waktu: Bertha 25 47 301 26 50 60 65 21
Pemenang: Bertha
Jumlah soal: 7
Total waktu: 294
PS D:\SEMESTER 3\PRATIUM ALPRO 2>

```
- STATUS BAR:** Shows the current file is `Ln 18, Col 1`, `Tab Size: 4`, `UTF-8`, `CRLF`, `Go`, `1.23.1`, and `AI Code Chat`.

Deskripsi Program

hitungSkor:

- Menghitung jumlah soal yang diselesaikan dan total waktu untuk seorang peserta.
- Menggunakan parameter pointer untuk mengubah nilai dari variabel soal dan skor yang diberikan.

main:

- Menerima input jumlah peserta.
- Untuk setiap peserta, membaca nama dan waktu penyelesaian 8 soal.
- Menggunakan hitungSkor untuk menghitung total soal dan waktu yang digunakan.
- Menentukan pemenang berdasarkan jumlah soal terbanyak yang diselesaikan dan waktu terendah.

Mencetak nama pemenang, jumlah soal yang diselesaikan, dan total waktu yang dihabiskan.

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 $2n$, 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 skiena 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 cetakberet (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 dipisahkan oleh sebuah spasi

Sourcecode

```
package main

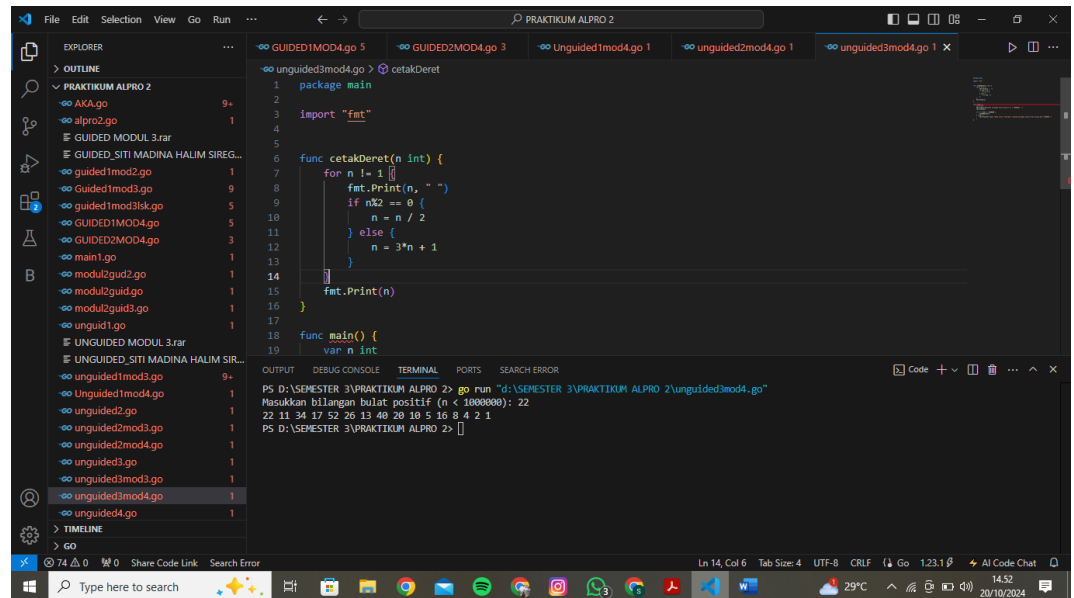
import "fmt"

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

func main() {
    var n int
    fmt.Print("Masukkan bilangan bulat positif (n < 1000000): ")
    fmt.Scan(&n)

    if n > 0 && n < 1000000 {
        cetakDeret(n)
    } else {
        fmt.Println("Input tidak valid. Pastikan n adalah bilangan positif dan kurang dari 1000000.")
    }
}
```

Screenshoot Output



The screenshot shows a Go IDE with the following code in the editor:

```
1 package main
2
3 import "fmt"
4
5
6 func cetakDeret(n int) {
7     for n != 1 {
8         fmt.Print(n, " ")
9         if n%2 == 0 {
10             n = n / 2
11         } else {
12             n = 3*n + 1
13         }
14     }
15     fmt.Print(n)
16 }
17
18 func main() {
19     var n int
```

The output console shows the following execution:

```
PS D:\SEMASTER 3\PRAKTIKUM ALPRO 2> go run "d:\SEMASTER 3\PRAKTIKUM ALPRO 2\unguided3mod4.go"
Masukkan bilangan bulat positif (n < 1000000): 22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS D:\SEMASTER 3\PRAKTIKUM ALPRO 2>
```

Deskripsi Program

cetakDeret:

- Menerima parameter n, yang merupakan nilai awal dari deret.
- Menggunakan loop untuk mencetak setiap suku dari deret sampai mencapai 1.
Memeriksa apakah n genap atau ganjil untuk menentukan suku berikutnya.

main:

- Membaca input dari pengguna dan memvalidasi bahwa n adalah bilangan positif dan kurang dari 1.000.000.
- Memanggil prosedur cetakDeret jika input valid.

Program ini akan terus berjalan dengan baik untuk nilai awal di bawah 1.000.000 sesuai dengan aturan yang ditetapkan.

Kesimpulan

Prosedur (Procedure) Prosedur merupakan sub program yang melakukan satu atau lebih proses tanpa mengembalikan nilai dari hasil dari proses yang dijalankan. Namun, pada prosedur diperbolehkan mencetak hasil luaran yang dihasilkan dari proses yang dijalankan. Contohnya, mencetak hasil dari proses aritmatika.

Ketika sebuah prosedur dipanggil, maka pada dasarnya dapat melakukan proses pertukaran data antara program utama dan sub program. Proses pertukaran ini dapat dilakukan dengan penggunaan parameter.

Daftar Pustaka

https://lmsspada.kemdikbud.go.id/pluginfile.php/679184/mod_resource/content/7/Modul%20PB%2010%20-%20Prosedur.pdf