

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL IV
PROSEDUR**



Disusun Oleh :

Muhammad Hamzah Haifan Ma'ruf

2311102091

IF-11-05

Dosen Pengampu :

Arif Amrulloh, S.Kom., M.Kom

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

A. Definisi Procedur

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.
- 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,...

B. Deklarasi Procedur

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 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.

C. 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 dimintadari suatu prosedur. Sebagai contoh prosedur cetakNFibo di atas dipanggil dengan menuliskan namanya, kemudian sebuah variabel atau nilai integer tertentu sebagai argumen 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 pemanggilan dengan notasi pseudocode dan golang adalah sama. Argumen yang digunakan untuk parameter n berupa integer (sesuai deklarasi) yang terdapat pada suatu variable (carapemanggilan#1) / nilainya secara langsung (carapemanggilan #2).

D. 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
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 cetak Pesan(pesan bilangan) harus sesuai urutan tipe data pada func cetak Pesan (Mstring flag int, yaitu string kemudian integer.

E. 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 }
```

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

- 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 `volume tabung` adalah parameter formal (teks berwarna merah). Artinya ketika memanggil `volume Tabung` maka kita 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 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 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 semua parameter formal pada fysi 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 st pemanggil setelah subprogram tersebut selesai dieksekusi. Pass by reference sebaliknya 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. 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 byreference. 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.

II. GUIDED

1. GUIDED 1

Studi Case : Buatlah sebuah program beserta prosedur yang digunakan untuk menghitung nilai faktorial dan permutasi. Masukan terdiri dari dua buah bilangan positif a dan b . Keluaran berupa sebuah bilangan bulat yang menyatakan nilai a permutasi b apabila $a \geq b$ atau b permutasi a untuk kemungkinan yang lain.

Sourcecode

```
package main

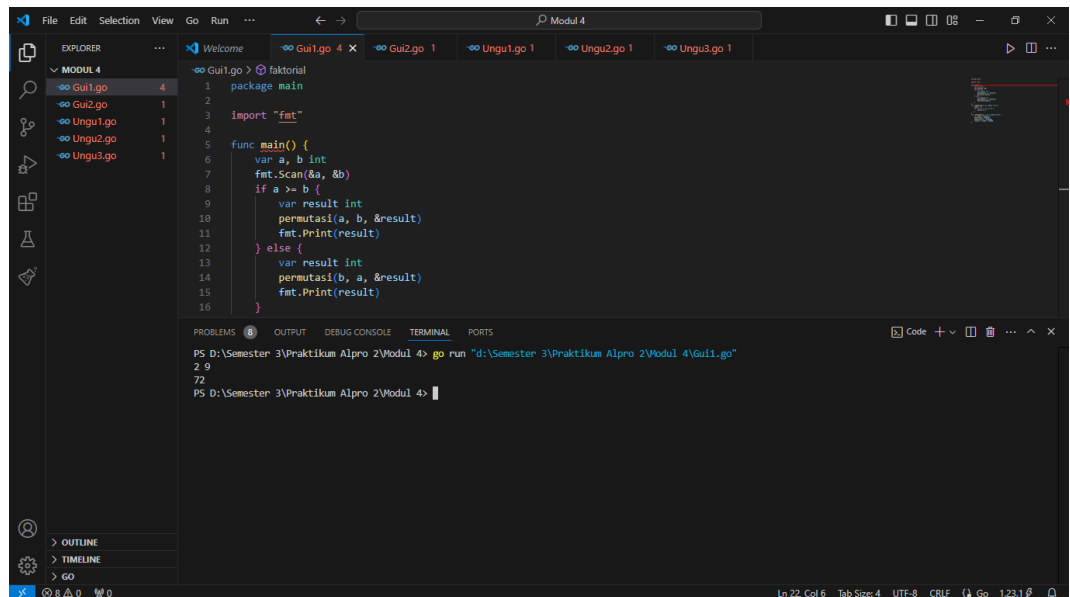
import "fmt"

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

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

func permutasi(n, r int, result *int) {
    var faktN, faktNR int
    faktorial(n, &faktN)
    faktorial(n-r, &faktNR)
    *result = faktN / faktNR
}
```

Screenshoot Output



Deskripsi Program

Program di atas dapat digunakan untuk menghitung permutasi dari dua bilangan bulat, n dan r. Permutasi menghitung kemungkinan susunan elemen yang berbeda dari sekumpulan elemen, di mana urutan elemen diperhitungkan. Program dimulai dengan pengguna memasukkan dua bilangan bulat, a dan b. Kemudian, algoritma memeriksa nilai a dan b. Jika a lebih besar atau sama dengan b, maka program menghitung permutasi a permutasi b ($P(a, b)$), dan jika b lebih besar, maka program menghitung permutasi b permutasi a ($P(b, a)$).

Fungsi faktorial dan permutasi adalah dua fungsi utama program. Fungsi faktorial(n , *hasil) menghitung faktorial dari bilangan n dengan iterasi, di mana faktorial adalah hasil perkalian dari 1 hingga n. Hasil perhitungan disimpan dalam variabel pointer hasil. Selanjutnya, fungsi permutasi(n , r, *result) menghitung permutasi menggunakan rumus $P(n, r) = \frac{n!}{(n-r)!}$. Untuk melakukan ini, fungsi faktorial diminta untuk menghitung faktorial dari n dan n - r, dan kemudian membagi hasilnya. Variabel pointer hasil mengandung hasil perhitungan permutasi. Program menampilkan hasil permutasi di layar setelah proses perhitungan selesai. Sebagai contoh, jika pengguna memasukkan angka 5 pada a dan 3 pada b, program akan menghitung

2. GUIDED 2

Studi Case : Buatlah program beserta prosedur yang digunakan untuk menghitung luas dan keliling persegi. Masukan terdiri dari sisi persegi. Keluaran berupa hasil luas dan keliling persegi.

Sourcecode

```
package main

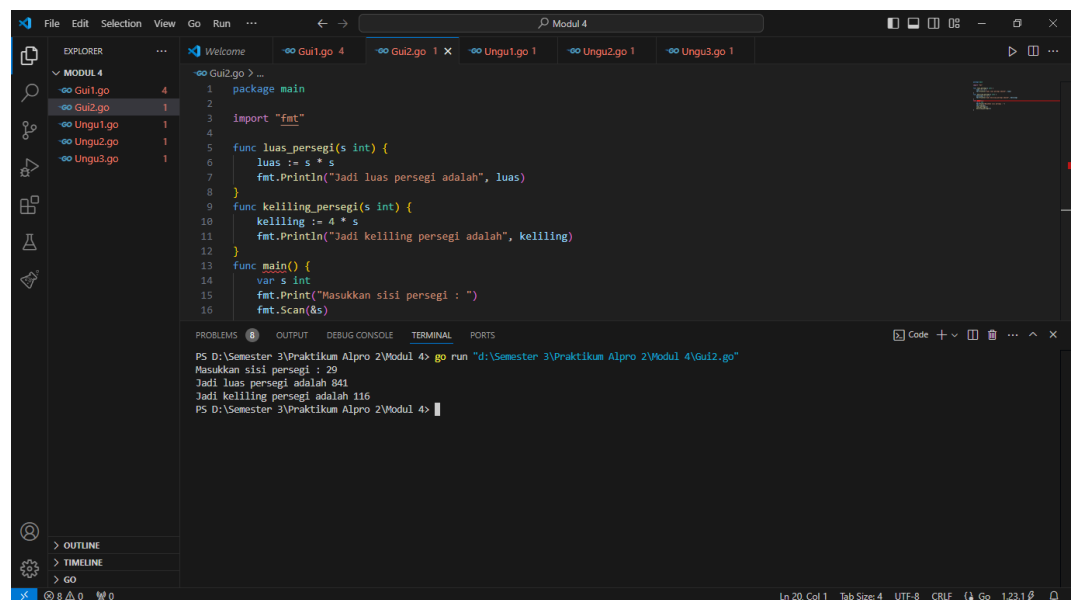
import "fmt"

func luas_persegi(s int) {
    luas := s * s
    fmt.Println("Jadi luas persegi adalah", luas)
}

func keliling_persegi(s int) {
    keliling := 4 * s
    fmt.Println("Jadi keliling persegi adalah", keliling)
}

func main() {
    var s int
    fmt.Print("Masukkan sisi persegi : ")
    fmt.Scan(&s)
    luas_persegi(s)
    keliling_persegi(s)
}
```

Screenshoot Output



The screenshot displays a Go IDE interface. The Explorer panel on the left shows a project structure with files: Gui1.go, Gui2.go, Ungu1.go, Ungu2.go, and Ungu3.go. The main editor window shows the source code for Gui2.go, which is identical to the code provided in the 'Sourcecode' section. Below the editor, the 'TERMINAL' panel shows the execution output. The command 'go run "d:\Semester 3\Praktikum Alpro 2\Modul 4\Gui2.go"' has been executed, resulting in the following output:

```
PS D:\Semester 3\Praktikum Alpro 2\Modul 4> go run "d:\Semester 3\Praktikum Alpro 2\Modul 4\Gui2.go"
Masukkan sisi persegi : 29
Jadi luas persegi adalah 841
Jadi keliling persegi adalah 116
PS D:\Semester 3\Praktikum Alpro 2\Modul 4>
```

Deskripsi Program

Program di atas berfungsi untuk menghitung luas dan keliling sebuah persegi. Ini dimulai dengan meminta pengguna untuk memasukkan nilai

sisi persegi. Kemudian, program melakukan perhitungan dengan dua fungsi berbeda. Setelah menerima input sisi persegi, fungsi pertama, "luas_persegi(s int)," menghitung luas persegi dengan rumus $\text{luas} = s \text{ kali } s$, dan menampilkan hasilnya. Fungsi kedua, "keliling_persegi(s int)" menghitung keliling persegi dengan rumus $\text{keliling} = 4 \text{ kali } s$, dan juga menampilkan hasilnya. Kedua fungsi ini digunakan secara berurutan untuk menghitung dan menampilkan. Misalnya, program akan menghitung luas menjadi 25 dan keliling menjadi 20, jika pengguna memasukkan nilai 5 sebagai panjang sisi. Kemudian, program menampilkan hasil tersebut di layar.

III. UNGUIDED

1. UNGUIDED 1

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 kedalam suatu program. Oleh karena itu bersediakah kalian membantu Jonas? Masukan terdiri dari empat buah bilangan asli a, b, c, dan d yang dipisahkan oleh spasi, dengan syarat $a \geq c$ 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!

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

Sourcecode

```
package main

import (
    "fmt"
)

// menghitung faktorial dan menyimpannya dalam pointer
hasil
func hitungFaktorial(n int, hasil *int) {
    *hasil = 1
    for i := 1; i <= n; i++ {
        *hasil *= i
    }
}
```

```

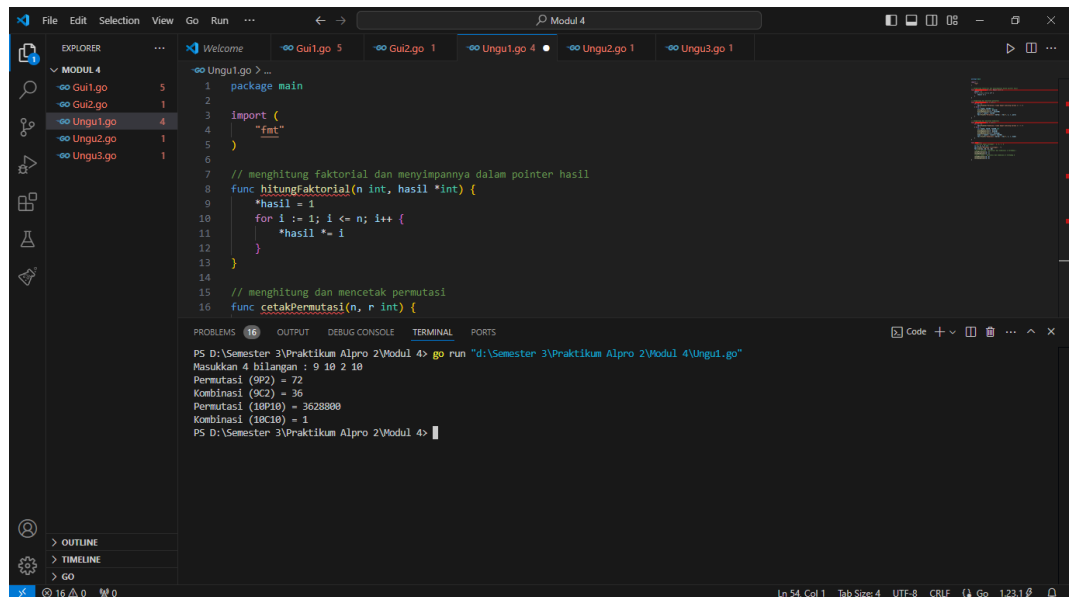
// menghitung dan mencetak permutasi
func cetakPermutasi(n, r int) {
    if n < r {
        fmt.Println("Permutasi tidak dapat dihitung karena
n < r.")
    } else {
        var faktN, faktNR int
        hitungFaktorial(n, &faktN)
        hitungFaktorial(n-r, &faktNR)
        perm := faktN / faktNR
        fmt.Printf("Permutasi (%dP%d) = %d\n", n, r, perm)
    }
}

// menghitung dan mencetak kombinasi
func cetakKombinasi(n, r int) {
    if n < r {
        fmt.Println("Kombinasi tidak dapat dihitung karena
n < r.")
    } else {
        var faktN, faktR, faktNR int
        hitungFaktorial(n, &faktN)
        hitungFaktorial(r, &faktR)
        hitungFaktorial(n-r, &faktNR)
        komb := faktN / (faktR * faktNR)
        fmt.Printf("Kombinasi (%dC%d) = %d\n", n, r, komb)
    }
}

func main() {
    //inputkan empat bilangan : a, b, c, d
    var a, b, c, d int
    fmt.Print("Masukkan 4 bilangan : ")
    fmt.Scan(&a, &b, &c, &d)
    //baris pertama : permutasi dan kombinasi a terhadap
c
    cetakPermutasi(a, c)
    cetakKombinasi(a, c)
    //baris kedua : permutasi dan kombinasi b terhadap d
    cetakPermutasi(b, d)
    cetakKombinasi(b, d)
}

```

Screenshot Output



The screenshot shows a Go IDE with a file explorer on the left containing 'MODUL 4' with files 'Gui1.go', 'Gui2.go', 'Ungu1.go', 'Ungu2.go', and 'Ungu3.go'. The main editor displays 'Ungu1.go' with the following code:

```
1 package main
2
3 import (
4     "fmt"
5 )
6
7 // menghitung faktorial dan menyimpannya dalam pointer hasil
8 func hitungFaktorial(n int, hasil *int) {
9     *hasil = 1
10    for i := 1; i <= n; i++ {
11        *hasil *= i
12    }
13 }
14
15 // menghitung dan mencetak permutasi
16 func cetakPermutasi(n, r int) {
```

The terminal at the bottom shows the execution output:

```
PS D:\Semester 3\Praktikum Alpro 2\Modul 4> go run "d:\Semester 3\Praktikum Alpro 2\Modul 4\Ungu1.go"
Masukkan 4 bilangan : 9 10 2 10
Permutasi (9P2) = 72
Kombinasi (9C2) = 36
Permutasi (10P10) = 3628800
Kombinasi (10C10) = 1
PS D:\Semester 3\Praktikum Alpro 2\Modul 4>
```

Deskripsi Program

Program di atas, yang dibuat dalam bahasa Go, bertujuan untuk menghitung dan mencetak permutasi serta kombinasi dari dua pasang bilangan yang dimasukkan oleh pengguna. Program dimulai dengan meminta pengguna untuk memasukkan empat bilangan bulat, yaitu a, b, c, dan d. Selain itu, ada fungsi `hitungFaktorial` yang digunakan untuk menghitung faktorial dari bilangan n , dengan hasil disimpan dalam pointer `hasil`. Fungsi `cetakPermutasi` juga digunakan untuk menghitung permutasi dari dua bilangan, $P(n, r) = \frac{n!}{(n-r)!}$. Selain itu, fungsi `hitungFaktorial` dihubungkan untuk menghitung faktorial dari n dan $n-r$, dan menampilkan

Fungsi cetak kombinasi adalah untuk menghitung kombinasi dari dua bilangan, seperti $C(n, r) = \frac{n!}{r!(n-r)!}$, dengan cara yang serupa dan menampilkan pesan jika n lebih kecil dari r . Dalam fungsi main, setelah input diterima, program menghitung dan menampilkan permutasi dan kombinasi a terhadap c dan b terhadap d. Jika seseorang memasukkan nilai 5, 4, 3, dan 2, program akan menampilkan hasil permutasi dan kombinasi yang sesuai dengan nilai yang dimasukkan.

2. UNGUIDED 2

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 game yang

mencari pemenang dari daftar peserta yang diberikan. Program harus dibuat modular, yaitu dengan membuat prosedur hitung skor 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. Prosedur 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.

| 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"
    "strings"
)

const maxTime = 301 //waktu max

// menghitung skor (jumlah soal yang diselesaikan dan total waktu)
func hitungSkor(waktu [8]int, soal *int, skor *int) {
    *soal = 0
    *skor = 0
    for _, waktuSoal := range waktu {
        if waktuSoal < maxTime { //jika soal diselesaikan
            *soal++
            *skor += waktuSoal
        }
    }
}
```

```

func main() {
    var pemenang string
    var soalPemenang, skorPemenang int
    soalPemenang = 0
    skorPemenang = maxTime * 8 //nilai awal skor pemenang
    maksimal
    for {
        var nama string
        var waktu [8]int
        //membaca input nama peserta
        fmt.Print("Masukkan nama peserta (atau 'Selesai'
untuk mengakhiri) : ")
        fmt.Scan(&nama)
        nama = strings.TrimSpace(nama)
        //jika input adalah 'Selesai', hentikan loop
        if nama == "Selesai" {
            break
        }
        //membaca waktu penyelesaian 8 soal
        fmt.Print("Masukkan waktu penyelesaian 8 soal
(dalam menit) : ")
        for i := 0; i < 8; i++ {
            fmt.Scan(&waktu[i])
        }
        //hitung soal yang diselesaikan dan total waktu
        var soal, skor int
        hitungSkor(waktu, &soal, &skor)
        //tentukan pemenang berdasarkan jumlah soal dan
total waktu
        if soal > soalPemenang || (soal == soalPemenang
&& skor < skorPemenang) {
            pemenang = nama
            soalPemenang = soal
            skorPemenang = skor
        }
    }
    //print pemenang
    fmt.Printf("Pemenang : %s, Soal yang diselesaikan :
%d, Total waktu : %d menit\n", pemenang, soalPemenang,
skorPemenang)
}

```

Screenshoot Output

```
1 package main
2
3 import (
4     "fmt"
5     "strings"
6 )
7
8 const maxTime = 301 //waktu max
9
10 // menghitung skor (jumlah soal yang diselesaikan dan total waktu)
11 func hitungSkor(waktu [8]int, soal *int, skor *int) {
12     *soal = 0
13     *skor = 0
14     for _, waktuSoal := range waktu {
15         if waktuSoal < maxTime { //jika soal diselesaikan
16             *soal++
17         }
18     }
19 }
20
21 func main() {
22     // ...
23 }
```

PS D:\Semester 3\Praktikum Alpro 2\Modul 4> go run "d:\Semester 3\Praktikum Alpro 2\Modul 4\Ungu2.go"

Masukkan nama peserta (atau 'Selesai' untuk mengakhiri) : boy
Masukkan waktu penyelesaian 8 soal (dalam menit) : 1 2 3 4 5 6 7 8
Masukkan nama peserta (atau 'Selesai' untuk mengakhiri) : girl
Masukkan waktu penyelesaian 8 soal (dalam menit) : 9 8 7 6 5 4 3 2
Masukkan nama peserta (atau 'Selesai' untuk mengakhiri) : Selesai
Pemenang : boy, Soal yang diselesaikan : 8, Total waktu : 36 menit
PS D:\Semester 3\Praktikum Alpro 2\Modul 4>

Deskripsi Program

Program di atas digunakan untuk menentukan pemenang kompetisi berdasarkan jumlah soal yang diselesaikan dan waktu total yang dihabiskan. Program ini dimulai dengan menetapkan waktu maksimum, atau `maxTime`, yaitu 301 menit, yang dianggap sebagai waktu batas untuk menyelesaikan setiap soal. Fungsi `hitungSkor` menghitung jumlah soal yang diselesaikan dan total waktu penyelesaian berdasarkan waktu dari delapan soal, dan mengubah jumlah soal yang diselesaikan dan total skor jika waktu penyelesaian kurang dari `maxTime`. Dalam fungsi `main`, program meminta nama peserta dan waktu penyelesaian setiap soal.

Loop akan dihentikan jika namanya "Selesai". Setelah itu, program menggunakan fungsi `hitung skor` untuk menghitung jumlah soal yang diselesaikan dan total waktu. Kemudian, jika ada kesamaan, pemenang ditentukan berdasarkan jumlah soal yang diselesaikan. Program mencetak jumlah soal yang diselesaikan, jumlah waktu yang dihabiskan, dan nama pemenang setelah semua peserta selesai memasukkan data. Misalnya, jika data untuk dua peserta dimasukkan, program dapat menampilkan hasil seperti "Pemenang: John Doe, Soal yang diselesaikan: 6, Total waktu: 250 menit".

3. UNGUIDED 3

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 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 1 parameter formal, yaitu nilai dari suku awal. prosedur cetakDeret(inn: integer) Masukan berupa satu bilangan integer positif yang lebih kecil dari 1000000. Keluaran terdiri dari satu baris saja. Setiap suku dari deret tersebut dicatati dalam baris yang dan dipisahkan oleh sebuah spasi.

| No | Masukan | Keluaran |
|----|---------|--------------------------------------------|
| 1 | 22 | 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1 |

Sourcecode

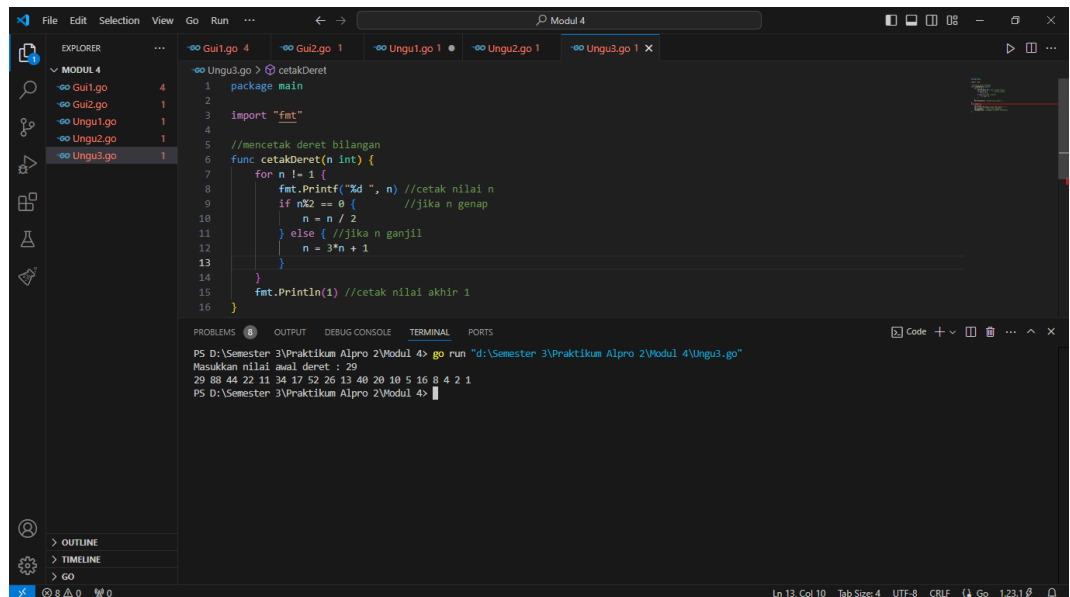
```
package main

import "fmt"

//mencetak deret bilangan
func cetakDeret(n int) {
    for n != 1 {
        fmt.Printf("%d ", n) //cetak nilai n
        if n%2 == 0 {        //jika n genap
            n = n / 2
        } else { //jika n ganjil
            n = 3*n + 1
        }
    }
    fmt.Println(1) //cetak nilai akhir 1
}

func main() {
    var n int
    fmt.Print("Masukkan nilai awal deret : ")
    fmt.Scan(&n) //input dari pengguna
    cetakDeret(n) //panggil prosedur cetakDeret
}
```

Screenshoot Output



```
1 package main
2
3 import "fmt"
4
5 //mencetak deret bilangan
6 func cetakDeret(n int) {
7     for n != 1 {
8         fmt.Printf("%d ", n) //cetak nilai n
9         if n%2 == 0 {        //jika n genap
10             n = n / 2
11         } else { //jika n ganjil
12             n = 3*n + 1
13         }
14     }
15     fmt.Println(1) //cetak nilai akhir 1
16 }
```

```
PS D:\Semester 3\Praktikum Alpro 2\Modul 4> go run "d:\Semester 3\Praktikum Alpro 2\Modul 4\Ungu3.go"
Masukkan nilai awal deret : 29
29 88 44 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS D:\Semester 3\Praktikum Alpro 2\Modul 4>
```

Deskripsi Program

Program diatas mencetak deret bilangan berdasarkan aturan konjektur Collatz suatu bilangan genap dibagi dua, suatu bilangan ganjil ditambah satu, dan seterusnya. Proses ini diulang hingga nilai satu dicapai. Fungsi cetak: Parameter n, yang merupakan nilai awal deret, diberikan kepada deret. Nilai n dicetak oleh program selama n tidak sama dengan 1. Jika n genap, n dibagi dua, dan jika n ganjil, n diperbarui menjadi $3n + 1$. Fungsi akan mencetak nilai 1 untuk menyelesaikan deret setelah mencapai angka 1. Dalam fungsi main, program meminta pengguna untuk memasukkan nilai awal deret yang disimpan dalam variabel n, dan kemudian memanggil fungsi cetakDeret(n) untuk memulai proses pencetakan deret.

IV. KESIMPULAN

Kesimpulan dari seluruh program yang telah dijelaskan di atas adalah bahwa masing-masing program memiliki tujuan dan fungsi spesifik dalam konteks pemrograman menggunakan bahasa Go. Program-program tersebut mencakup perhitungan matematis, pengolahan data, dan logika iteratif.

1. Permutasi dan Kombinasi

Program ini menghitung dan menampilkan permutasi serta kombinasi dari bilangan yang dimasukkan pengguna. Dengan menggunakan konsep faktorial, program menentukan pemenang berdasarkan jumlah soal yang diselesaikan dan total waktu penyelesaian dalam suatu kompetisi.

2. Luas dan Keliling Persegi

Program ini menghitung luas dan keliling dari persegi berdasarkan panjang sisi yang diberikan pengguna. Program ini mengilustrasikan penggunaan fungsi untuk memisahkan logika perhitungan dalam program yang lebih terstruktur.

3. Skor Kompetisi

Program ini menentukan pemenang dari kompetisi dengan menghitung jumlah soal yang diselesaikan dan total waktu penyelesaian. Dengan menggunakan struktur kontrol, program ini mampu mengambil input dari pengguna secara dinamis dan menghasilkan output yang jelas.

4. Deret Bilangan Collatz

Program ini mencetak deret bilangan berdasarkan aturan konjektur Collatz. Dengan proses iteratif yang sederhana, program ini menunjukkan bagaimana angka dapat bertransformasi mengikuti aturan matematis tertentu hingga mencapai nilai akhir.

Secara keseluruhan, program-program ini mencerminkan prinsip dasar pemrograman, seperti penggunaan fungsi, pengendalian alur, dan pemrosesan input/output, yang semuanya berkontribusi pada pembuatan solusi untuk berbagai permasalahan matematis dan logika.

V. REFERENSI

[1] Modul 4 Praktikum Algoritma dan Pemrograman 2

[2] Belajar *golang* - dasar pemrograman *golang*. (n.d.).
<https://dasarpemrogramangolang.novalagung.com/1-berkenalan-dengan-golang.html>

[3] *Golang : Pengertian dan 12 Framework untuk Website Development*. (n.d.).
<https://kelas.work/blogs/mengenal-golang-dan-beberapa-framework-yang-dapat-digunakan>