LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 5

REKURSIF

DISUSUN OLEH:

Keishin Naufa Alfaridzhi

103112400061

S1 IF-12-01

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

A. Bahasa Yang Digunakan

Pada praktikum ini bahasa pemrograman yang digunakan adalah bahasa pemrograman Go, sesuai dengan modul yang menjadi acuan praktikum. Golang (atau) Go adalah bahasa pemrograman baru, yang mulai dilirik oleh para developer karena kelebihan-kelebihan yang dimilikinya. Sudah banyak Perusahaan besar yang menggunakan bahasa ini untuk produk-produk mereka hingga di level production.

B. Komentar

Komentar biasa dimanfaatkan untuk menyisipkan catatan pada kode program, menulis penjelasan atau deskripsi mengenai suatu blok kode, atau bisa juga digunakan untuk me-remark kode (men-non-aktifkan kode yang tidak digunakan). Komentar akan diabaikan Ketika kompilasi maupun eksekusi program.

Ada 2 jenis komentar di Golang, yaitu inline dan multiline.

1. Komentar Inline

Penulisan komentar jenis ini diawali dengan tanda *double slash* (//) lalu diikuti pesan komentarnya. Komentar inline hanya berlaku untuk satu baris pesan saja. Jika pesan komentar lebih dari satu baris, maka tanda *double slash* harus ditulis lagi di baris selanjutnya.

2. Komentar Multiline

Komentar yang cukup panjang akan lebih rapi jika ditulis menggunakan teknik komentar multiline. Ciri dari komentar jenis ini adalah penulisannya diawali dengan tanda (/*) dan diakhiri (*/).

C. Variabel

Golang mengadopsi 2 jenis penulisan variabel, yang dituliskan tipe data-nya dan yang tidak. Kedua cara tersebut intinya adalah sama, pembedanya hanyalah cara penulisannya saja. Untuk penulisan variabel dengan tipe data, keyword *var* digunakan untuk deklarasi variabel kemudian diakhiri dengan tipe data misalnya *string*. Kemudian untuk penulisan variabel tanpa tipe data, variabel dideklarasikan dengan menggunakan metode type inference. Penandanya tipe data tidak dituliskan pada saat deklarasi. Pada penggunaan metode ini, operand (=) harus diganti dengan (:=) dan keyword *var* dihilangkan.

Golang memiliki aturan unik yang tidak dimiliki bahasa lain, yaitu tidak boleh ada satupun variabel yang menganggur. Artinya, semua variabel yang dideklarasikan harus digunakan. Jika terdapat variabel yang tidak digunakan tapi dideklarasikan, program akan gagal dikompilasi. Untuk mengatasi itu, golang memiliki variabel yaitu underscore. Underscore (_) adalah predefined variabel yang bisa dimanfaatkan untuk menampung nilai yang tidak dipakai.

D. Tipe Data

Golang mengenal beberapa jenis tipe data, diantaranya adalah tipe data numerik (decimal dan non-desimal), string, dan boolean.

1. Tipe Data Numerik Non-Desimal (uint, int)
2. Tipe Data Numerik Desimal (float64, float32)
3. Tipe Data Bool (true, false)
4. Tipe Data String (string, " ")

E. Operator Aritmatika

Operator aritmatika merupakan operator yang digunakan untuk operasi yang sifatnya perhitungan. Golang mendukung beberapa operator aritmatika standar, yaitu:

1. Penjumlahan (+)
2. Pengurangan (-)
3. Perkalian (*)
4. Pembagian (/)
5. Modulus atau sisa hasil pembagian (%)

F. Seleksi Kondisi

Seleksi kondisi pada program berguna untuk mengontrol sebuah blok kode yang akan dieksekusi. Yang dijadikan acuan oleh selksi kondisi adalah nilai bertipe bool, bisa berasal dari variabel, ataupun hasil operasi perbandingan. Nilai tersebut menentukan blok kode mana yang akan dieksekusi. Go memiliki 2 macam keyword untuk selesksi kondisi, yaitu if else dan switch.

1. **If Expression**

   If adalah salah satu kata kunci yang digunakan dalam percabangan. Percabangan artinya kitabisa mengeksekusi kode program tertentu ketika suatu kondisi terpenuhi. Hampir semua bahasa pemrograman mendukung if expression.

2. **Else if expression**

   Terkadang kita butuh membuat beberapa kondisi. Kasus seperti ini dapat menggunakan else if expression. If mendukung short statement sebelum kondisi.

Hal ini sangat cocok untuk membuat statement yang sederhana sebelum melakukan pengecekan terhadap kondisi.

3. **Switch-Case**

   Switch merupakan seleksi kondisi yang sifatnya fokus pada satu variabel, lalu kemudian di-cek nilainya. Contoh sederhananya seperti penentuan apakah nilai variabel x adalah: 1, 2, 3, atau lainnya. Perlu diketahui, switch pada pemrograman Go memiliki perbedaan dibanding bahasa lain. Di Go, ketika sebuah case terpenuhi, tidak akan dilanjutkan ke pengecekan case selanjutnya, meskipun tidak ada keyword "break" di situ. Konsep ini berkebalikan dengan switch pada umumnya pemrograman lain (yang ketika sebuah case terpenuhi, maka akan tetap dilanjut mengecek case selanjutnya kecuali ada keyword "break").

G. Perulangan

Perulangan merupakan proses mengulang dan mengeksekusi blok kode tanpa henti sesuai dengan kondisi yang dijadikan acuan. Biasanya disiapkan variabel untuk iterasi atau penanda kapan perulangan akan dihentikan.

a. For Loop

For loop merupakan statement perulangan dasar dan cukup sering ditemui. Format for loop yaitu sebagai berikut.

- *Init Statement*: bagian ini akan dieksekusi sebelum perulangan dimulai. Biasanya diisi dengan mendeklarasi variabel iterasi.
- *Condition Expression*: bagian ini akan dicek dan dieksekusi setiap perulangan yang dilakukan, jika true maka perulangan akan terus berjalan hingga kondisi bernilai false.
- *Post Statement*: statement ini akan dieksekusi pada akhir iterasi. Jika terdapat range, maka perulangan akan dieksekusi untuk setiap item pada range.

b. While Loop

While loop merupakan perulangan yang akan terus berjalan hingga suatu kondisi terpenuhi. Penulisan while loop adalah dengan menuliskan kondisi setelah keyword for (hanya kondisi). Deklarasi dan iterasi variabel counter tidak dituliskan setelah keyword, hanya kondisi perulangan saja. Konsepnya mirip seperti while milik bahasa pemrograman lain.

c. Repeat Until

Untuk Repeat Until ini mirip seperti for loop biasa namun hanya menggunakan inisiasi dan kondisi saja.

H. Fungsi

Dalam konteks pemrograman, fungsi adalah sekumpulan blok kode yang dibungkus dengan nama tertentu. Penerapan fungsi yang tepat akan menjadikan kode lebih modular dan juga *dry* (singkatan dari *don't repeat yourself*) yang artinya kita tidak perlu menuliskan banyak kode untuk kegunaan yang sama berulang kali. Cukup deklarasikan sekali saja blok kode sebagai suatu fungsi, lalu panggil sesuai kebutuhan.

1. Penerapan Fungsi

Sebenarnya kita sudah mengimplementasikan fungsi pada banyak praktek sebelumnya, yaitu fungsi `main()`. Fungsi `main()` sendiri merupakan fungsi utama pada program Go, yang akan dieksekusi ketika program dijalankan.

Selain fungsi `main()`, kita juga bisa membuat fungsi lainnya. Dan caranya cukup mudah, yaitu dengan menuliskan keyword `func` kemudian diikuti nama fungsi, lalu kurung `()` (yang bisa diisi parameter), dan diakhiri dengan kurung kurawal untuk membungkus blok kode.

Parameter merupakan variabel yang menempel di fungsi yang nilainya ditentukan saat pemanggilan fungsi tersebut. Parameter sifatnya opsional, suatu fungsi bisa tidak memiliki parameter, atau bisa saja memeliki satu atau banyak parameter (tergantung kebutuhan).

2. Fungsi dengan Nilai Balik / Return Value

Selain parameter, fungsi bisa memiliki attribute **return value** atau nilai balik. Fungsi yang memiliki return value, saat deklarasinya harus ditentukan terlebih dahulu tipe data dari nilai baliknya.

3. Fungsi tanpa Nilai Balik / Return Value

Fungsi juga dapat tidak memiliki nilai balik yang dapat disebut juga sebagai Prosedur. 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.

II. GUIDED

1. **Source Code:**

```
package main
import "fmt"

func main() {
   var base, exp, n int

   // Input pangkat
   fmt.Print("Masukkan bilangan: ")
   fmt.Scanln(&base)
   fmt.Print("Masukkan pangkat: ")
   fmt.Scanln(&exp)

   fmt.Printf("%d^%d = %d\n", base, exp, pangkatIteratif(base, exp))

   // Input faktorial
   fmt.Print("Masukkan angka untuk faktorial: ")
   fmt.Scanln(&n)

   fmt.Printf("%d! = %d\n", n, faktorialIteratif(n))
}

func pangkatIteratif(base, exp int) int {
   hasil := 1
   for i := 0; i < exp; i++ {
      hasil *= base
   }
   return hasil
}

func faktorialIteratif(n int) int {
   hasil := 1
   for i := 2; i <= n; i++ {
      hasil *= i
   }
   return hasil
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum>go run "d:\Tugas\SEM2\Alpro\Praktikum\modul5\103112400061_
Masukkan bilangan: 4
Masukkan pangkat: 2
4^2 = 16
Masukkan angka untuk faktorial: 5
5! = 120
```

Penjelasan:
Program untuk menghitung bilangan pangkat dan faktorial dengan metode iterasi.

2. **Source Code:**

```
package main
import "fmt"

func main() {
   var base, exp, n int

   // Input pangkat
   fmt.Print("Masukkan bilangan: ")
   fmt.Scanln(&base)
   fmt.Print("Masukkan pangkat: ")
   fmt.Scanln(&exp)

   fmt.Printf("%d^%d = %d\n", base, exp, pangkatRekursif(base, exp))

   // Input faktorial
   fmt.Print("Masukkan angka untuk faktorial: ")
   fmt.Scanln(&n)

   fmt.Printf("%d! = %d\n", n, faktorialRekursif(n))
}

func pangkatRekursif(base, exp int) int {
   if exp == 0 {
      return 1
   }
   return base * pangkatRekursif(base, exp-1)
}

func faktorialRekursif(n int) int {
   if n == 0 || n == 1 {
      return 1
   }
   return n*faktorialRekursif(n-1)
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum>go run "d:\Tugas\SEM2\Alpro\Praktikum\modul5\103112400061_
Masukkan bilangan: 4
Masukkan pangkat: 2
4^2 = 16
Masukkan angka untuk faktorial: 5
5! = 120
```

Penjelasan:
Program untuk menghitung bilangan pangkat dan faktorial dengan metode rekursif.

III. UNGUIDED

1. Latihan no. 1

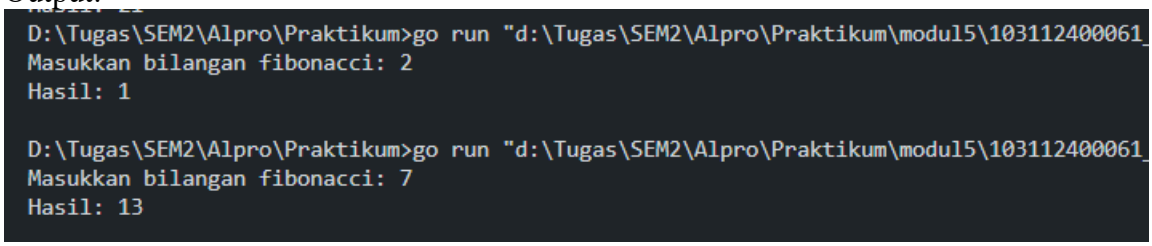
Source Code:

```
// KEISHIN NAUFA ALFARIDZHI
// 103112400061
package main
import "fmt"

func main() {
    var n int
    fmt.Scan(&n)
    fmt.Println(fibonacci(n))
}

func fibonacci(n int) int {
    if n == 0 {
        return 0
    } else if n == 1 || n == 2 {
        return 1
    }
    return fibonacci(n-1) + fibonacci(n-2)
}
```

Output:



Deskripsi Program:

Mencari nilai suatu suku dalam deret fibonacci. Terdapat function dengan nama fibonacci() untuk menghitung suku n-1 dan n-2 dengan metode rekursif yang mana akan memanggil parameter n untuk kemudian mencari nilai suku n-1 + n-2.

2. Latihan no. 2

Source Code:

```
// KEISHIN NAUFA ALFARIDZHI
// 103112400061
package main
import "fmt"

func main() {
    var n int
```

```
    fmt.Scan(&n)
    bintang(n)
}

func bintang(n int) {
    if n <= 0 {
        return
    }
    bintang(n-1)
    baris(n)
}

func baris(n int) {
    if n <= 0 {
        fmt.Println()
        return
    }
    fmt.Print("*")
    baris(n-1)
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum>go run "d:\Tugas\SEM2\Alpro\Praktikum\modul5\103112400061_
1
*

D:\Tugas\SEM2\Alpro\Praktikum>go run "d:\Tugas\SEM2\Alpro\Praktikum\modul5\103112400061_
5
*
**
***
****
*****
```

Deskripsi Program:

Program mencetak pattern bintang berdasarkan bilangan yang diinputkan. Terdapat dua function yaitu bintang() dan baris(). Function baris() digunakan untuk mencetak banyaknya bintang dalam 1 baris, sedangkan function bintang() digunakan untuk memanggil dan mengatur banyak nya bintang yang dicetak secara rekursif. Terdapat langkah algoritma penting dalam function ini yaitu pada saat melakukan rekursif pemanggilan function bintang(n-1) sebelum baris(n) yang mana akan membuat program ini mencetak baris dengan angka yang lebih kecil terlebih dahulu, 1 ke n bintang. Jika semisal kita balik pemanggilannya seperti function baris(n) sebelum bintang(n-1) maka pattern bintang akan terbalik upside-down, n ke 1 bintang.

3. Latihan no. 3

Source Code:

```
// KEISHIN NAUFA ALFARIDZHI
// 103112400061
package main
import "fmt"

func main() {
    var n int
    fmt.Scan(&n)
    faktor(n, 1)
}

func faktor(n int, i int) {
    if i <= n {
        if n%i == 0 {
            fmt.Print(i, " ")
        }
        faktor(n, i + 1)
    }
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum>go run "d:\Tugas\SEM2\Alpro\Praktikum\modul5\103112400061_
Masukkan bilangan: 5
Faktor dari bilangan: 1 5

D:\Tugas\SEM2\Alpro\Praktikum>go run "d:\Tugas\SEM2\Alpro\Praktikum\modul5\103112400061_
Masukkan bilangan: 12
Faktor dari bilangan: 1 2 3 4 6 12
```

Deskripsi Program:

Program untuk mencari deret faktor dari suatu bilangan n. Terdapat satu function bernama faktor() dengan 2 parameter yaitu n dan i yang memiliki base case jika i <= n, maka program akan eksekusi cek bilangan faktor dengan cek setiap n dimodulo dengan i, jika kondisi terpenuhi maka print deret faktor. Kemudian dilakukan rekursi faktor(n, i+1).

IV. KESIMPULAN

Pada praktikum ini telah dibahas perihal cara melakukan metode rekursif function dalam bahasa Go dan perbedaannya dengan iterasi. Perbedaan rekursi dengan iterasi yaitu rekursi memiliki base case sebagai kondisi untuk terminasi programnya. Perbedaan lainnya yaitu meliputi penulisan kode yang lebih pendek dan dapat digunakan untuk memecahkan masalah yang lebih kompleks.

V. DAFTAR PUSTAKA

Noval Agung Prayogo. *Dasar Pemrograman Golang.* Diakses pada 01 Oktober 2024. https://dasarpemrogramangolang.novalagung.com

Annisa Nur Isnaeni. *Golang — Seleksi Kondisi.* Diakses pada 01 Oktober 2024.

https://medium.com/@annisaisna/golang-seleksi-kondisi-f988ead004b4

Parvez Alam, *Golang for loop example | Golang Loops Tutorial – Phpflow.com* https://medium.com/@parvez1487/golang-for-loop-example-golang-loops-tutorial-phpflow-com-f4b2b0e57944