LAPORAN PRAKTIKUM

Algoritma Pemrograman

MODUL 12

WHILE-LOOP

Disusun Oleh:

MUHAMAD RAFI ALFIANSYAH

109082500191

S1IF-13-04

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

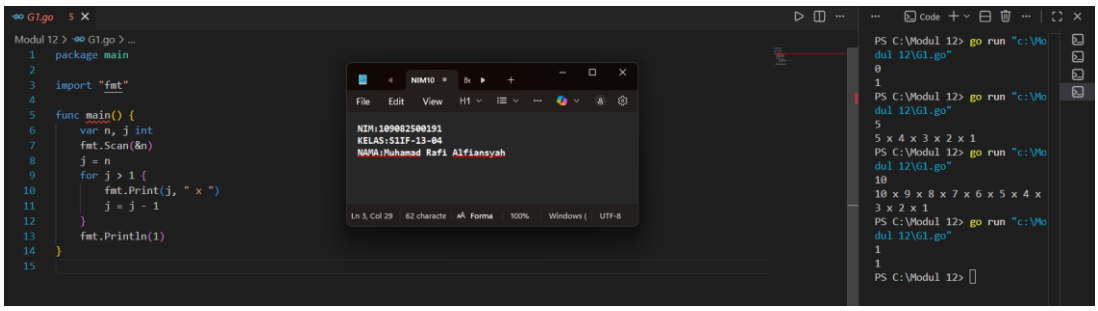
LATIHAN KELAS – GUIDED

1. Guided 1

Source Code

```
package main
import "fmt"
func main() {
	var n, j int
	fmt.Scan(&n)
	j = n
	for j > 1 {
		fmt.Print(j, " x ")
		j = j - 1
	}
	fmt.Println(1)
}
```

Screenshoot Program



Deskripsi Program

Program ini dibuat untuk menampilkan deret bilangan faktorial dari suatu bilangan bulat non-negatif n. Program dimulai dengan mendeklarasikan variabel n yang berfungsi untuk menyimpan nilai masukan dari pengguna. Setelah itu, nilai n disalin ke variabel j yang berfungsi sebagai penghitung selama proses perulangan. Penyalinan ini penting agar nilai asli n tetap terjaga dan tidak berubah selama perhitungan dilakukan. Setelah nilai input dibaca menggunakan fmt.Scan(&n), program mulai membentuk deret faktorial dalam bentuk ekspresi perkalian, bukan hasil perhitungannya. Deret faktorial yang dimaksud adalah susunan angka dari n menurun hingga 1, misalnya “5 x 4 x 3 x 2 x 1”.

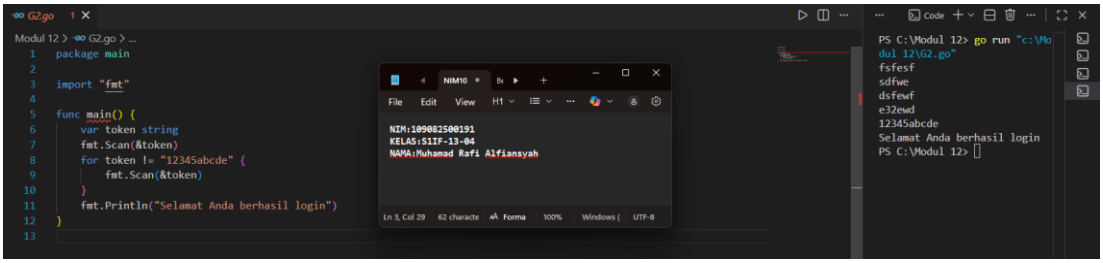
Untuk menghasilkan deret tersebut, program menggunakan perulangan dengan kondisi j > 1. Selama kondisi ini terpenuhi, program akan mencetak nilai j diikuti dengan tanda “ x ” sebagai pemisah antar bilangan. Setelah dicetak, nilai j dikurangi satu agar proses berlanjut ke angka berikutnya yang lebih kecil. Ketika j akhirnya mencapai nilai 1, perulangan berhenti dan angka 1 dicetak sebagai penutup deret tanpa tanda “ x ” di belakangnya. Dengan cara ini, program mampu membentuk deret faktorial lengkap sesuai nilai input yang diberikan. Program ini tidak menghitung nilai faktorial secara matematis, tetapi hanya menampilkan bentuk ekspresi perkalian yang menggambarkan proses faktorial secara visual dan berurutan.

2. Guided 2

Source Code

```
package main
import "fmt"
func main() {
	var token string
	fmt.Scan(&token)
	for token != "12345abcde" {
		fmt.Scan(&token)
	}
	fmt.Println("Selamat Anda berhasil login")
}
```

Screenshoot Program



Deskripsi Program

Program ini dibuat untuk mensimulasikan proses login sederhana menggunakan token. Pada program ini, token yang dianggap valid telah ditentukan, yaitu “12345abcde”. Program diawali dengan deklarasi variabel token bertipe string untuk menampung masukan dari pengguna. Nilai token pertama kali dibaca menggunakan fmt.Scan(&token). Setelah itu, program melakukan pemeriksaan apakah token yang diberikan sudah sesuai dengan token valid.

Jika token yang dimasukkan pengguna tidak sama dengan “12345abcde”, maka program akan terus meminta masukan ulang dari pengguna. Proses pengulangan ini dilakukan menggunakan perulangan dengan kondisi token != "12345abcde". Selama kondisi tersebut bernilai benar, artinya token masih salah, maka program akan kembali menjalankan fmt.Scan(&token) untuk meminta input token berikutnya. Mekanisme ini memastikan bahwa program tidak akan berhenti dan tidak akan memberikan akses sebelum pengguna memasukkan token yang benar.

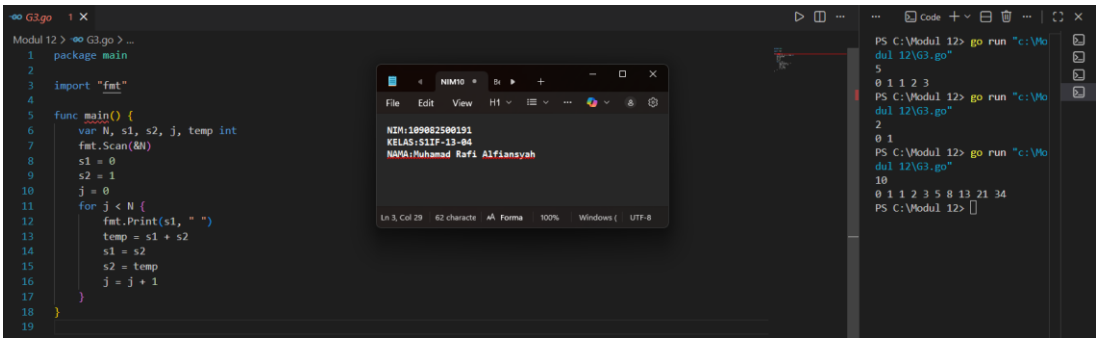
Ketika akhirnya pengguna memasukkan token yang sesuai dengan token valid, kondisi perulangan menjadi salah sehingga loop berhenti. Setelah keluar dari perulangan, program menampilkan pesan “Selamat Anda berhasil login” sebagai tanda bahwa proses autentikasi berhasil dilakukan. Dengan logika ini, program mampu memastikan bahwa hanya token yang benar yang dapat melewati proses login, sementara token yang salah akan selalu diminta ulang. Program ini merupakan contoh sederhana dari konsep validasi input dan autentikasi berbasis token.

3. Guided 3

Source Code

```
package main

import "fmt"

func main() {

    var N, s1, s2, j, temp int

    fmt.Scan(&N)

    s1 = 0

    s2 = 1

    j = 0

    for j < N {

        fmt.Print(s1, " ")

        temp = s1 + s2

        s1 = s2

        s2 = temp

        j = j + 1

    }

}
```

Screenshoot Program



Deskripsi Program

Program ini dibuat untuk menghasilkan dan menampilkan N bilangan pertama dari deret Fibonacci. Deret Fibonacci merupakan deret bilangan di mana setiap suku merupakan hasil penjumlahan dua suku sebelumnya, dengan dua nilai awal yaitu 0 dan 1. Program dimulai dengan deklarasi beberapa variabel, yaitu N untuk menyimpan jumlah suku yang ingin ditampilkan, s1 dan s2 sebagai dua suku awal deret Fibonacci, j sebagai penghitung iterasi, serta temp sebagai variabel sementara untuk menyimpan hasil penjumlahan dua suku sebelumnya. Nilai awal s1 diisi 0 dan s2 diisi 1 sesuai definisi deret Fibonacci.

Setelah variabel dideklarasikan, program membaca input nilai N dari pengguna menggunakan fmt.Scan(&N). Nilai N diasumsikan sebagai bilangan bulat positif dengan nilai minimal 2 sesuai instruksi soal. Selanjutnya, program memulai proses perulangan untuk mencetak N bilangan Fibonacci pertama. Perulangan menggunakan kondisi j < N, artinya program akan mencetak satu bilangan Fibonacci di setiap iterasi hingga jumlah yang dicetak mencapai N buah.

Di dalam perulangan, langkah pertama yang dilakukan adalah mencetak nilai s1 sebagai bilangan Fibonacci berikutnya. Setelah itu, nilai temp dihitung sebagai hasil penjumlahan s1 dan s2. Nilai s1 kemudian digeser menjadi nilai s2, dan s2 digeser menjadi nilai temp. Pergeseran nilai ini bertujuan agar pada iterasi berikutnya, perhitungan Fibonacci tetap mengikuti aturan bahwa setiap bilangan merupakan hasil penjumlahan dua bilangan sebelumnya. Nilai j kemudian ditambah satu sebagai penanda bahwa satu bilangan Fibonacci telah berhasil dicetak.

Proses ini berlangsung terus hingga jumlah bilangan yang dicetak mencapai N. Dengan demikian, program dapat menghasilkan deret Fibonacci sesuai jumlah yang diminta pengguna, dimulai dari bilangan 0 dan 1, lalu berlanjut sesuai pola Fibonacci. Program ini merupakan implementasi langsung dari logika pembentukan deret Fibonacci menggunakan perulangan dan variabel sementara untuk penggeseran nilai.

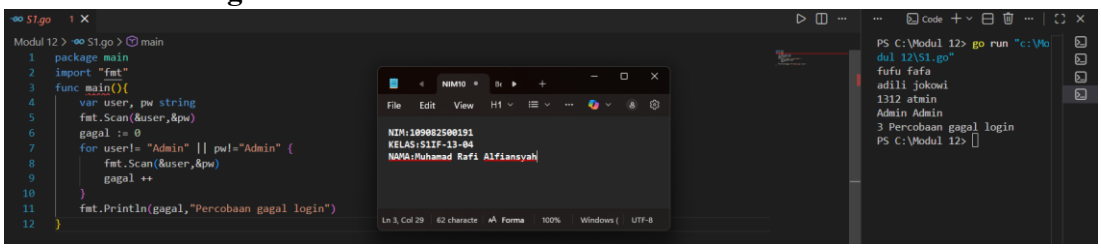
TUGAS

1. Tugas 1

Source Code

```
package main
import "fmt"
func main(){
	var user, pw string
	fmt.Scan(&user,&pw)
	gagal := 0
	for user!= "Admin" || pw!="Admin" {
		fmt.Scan(&user,&pw)
		gagal ++
	}
	fmt.Println(gagal,"Percobaan gagal login")
}
```

Screenshoot Program



Deskripsi Program

Program ini dibuat untuk menghitung berapa kali seorang pengguna gagal melakukan login karena salah memasukkan username maupun password. Pada program ini, dua variabel string yaitu user dan pw digunakan untuk menampung input username dan password dari pengguna. Nilai tersebut dibaca melalui perintah fmt.Scan(&user, &pw). Untuk mencatat jumlah kegagalan, program menggunakan variabel gagal yang bertipe integer dan pada awalnya bernilai 0.

Setelah menerima input pertama, program melakukan pengecekan apakah username dan password yang diberikan pengguna sudah sesuai dengan yang dianggap valid, yaitu “Admin” untuk username dan “Admin” untuk password. Pemeriksaan dilakukan menggunakan kondisi user != "Admin" || pw != "Admin". Kondisi ini berarti

bahwa jika salah satu atau kedua nilai yang dimasukkan masih salah, maka pengguna dianggap gagal login. Selama kondisi tersebut masih benar, program akan meminta input ulang username dan password. Setiap kali pengguna memasukkan kombinasi yang salah, variabel gagal akan ditambah satu sebagai penanda adanya percobaan login yang gagal.

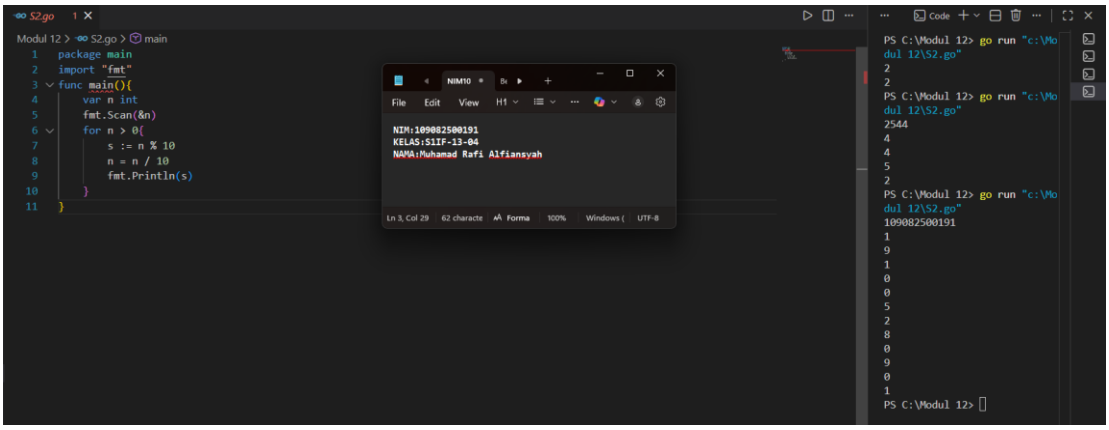
Perulangan akan terus berlanjut sampai pengguna memasukkan username dan password yang benar secara bersamaan. Ketika input sudah sesuai, kondisi pada perulangan menjadi salah sehingga proses perulangan berhenti. Setelah itu, program mencetak nilai variabel gagal untuk menunjukkan berapa kali percobaan login yang salah dilakukan sebelum akhirnya login berhasil. Dengan demikian, program ini tidak hanya melakukan validasi input, tetapi juga mencatat jumlah kesalahan yang terjadi selama proses login berlangsung. Program ini menggambarkan mekanisme kontrol login sederhana menggunakan perulangan dan logika kondisi.

2. Tugas 2

Source Code

```
package main
import "fmt"
func main(){
	var n int
	fmt.Scan(&n)
	for n > 0{
		s := n % 10
		n = n / 10
		fmt.Println(s)
	}
}
```

Screenshoot Program



Deskripsi Program

Program ini dibuat untuk mencacah setiap digit yang terdapat pada suatu bilangan bulat positif, kemudian menampilkan digit-digit tersebut mulai dari digit paling kanan hingga digit paling kiri. Program dimulai dengan deklarasi variabel n yang bertipe integer sebagai penampung nilai input dari pengguna. Nilai n ini dibaca menggunakan perintah fmt.Scan(&n) dan harus berupa bilangan bulat positif sesuai ketentuan soal.

Setelah input diterima, program memasuki proses pencacahan digit menggunakan perulangan yang berjalan selama nilai n lebih besar dari 0. Pada setiap iterasi, program mengambil digit paling belakang dari bilangan dengan menggunakan operasi modulo, yaitu s := n % 10. Operasi ini menghasilkan s sebagai digit satuan dari nilai n. Digit tersebut kemudian langsung dicetak ke layar sebagai salah satu hasil keluaran.

Setelah digit terakhir berhasil diambil, nilai n diperbarui menjadi n / 10. Operasi pembagian bilangan bulat ini menghapus digit paling belakang dari n, sehingga pada iterasi berikutnya digit sebelumnya dapat diambil. Proses ini terus berlangsung hingga seluruh digit berhasil dicacah dan ditampilkan.

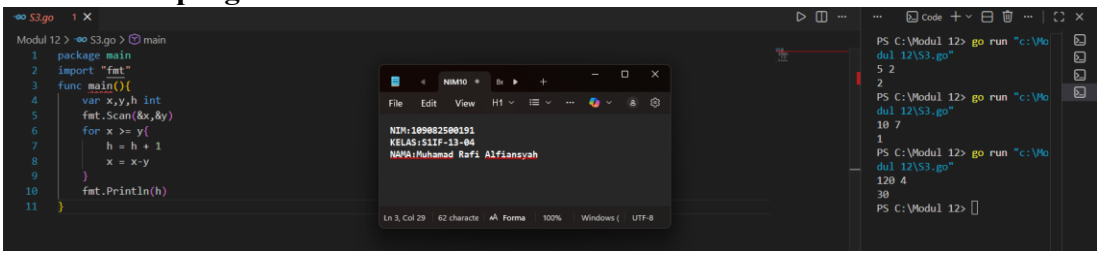
Dengan cara ini, program mampu menampilkan seluruh digit dari suatu bilangan positif satu per satu, dimulai dari digit terakhir hingga digit pertama. Program bekerja berdasarkan konsep dasar operasi modulo dan pembagian bilangan bulat untuk memisahkan digit-digit suatu bilangan secara berurutan dari kanan ke kiri.

3. Tugas 3

Source code

```
package main
import "fmt"
func main(){
	var x,y,h int
	fmt.Scan(&x,&y)
	for x >= y{
		h = h + 1
		x = x-y
	}
	fmt.Println(h)
}
```

Screenshoot program



Deskripsi Program

Program ini dibuat untuk menentukan hasil integer division (pembagian bilangan bulat tanpa sisa) dari dua buah bilangan bulat positif, yaitu x sebagai bilangan yang akan dibagi dan y sebagai bilangan pembagi. Pada program ini, proses pembagian tidak dilakukan menggunakan operator pembagian (/), melainkan harus disimulasikan menggunakan perulangan dan operasi pengurangan berulang.

Pertama-tama, program membaca dua buah masukan berupa bilangan bulat positif x dan y melalui perintah fmt.Scan. Sesuai ketentuan soal, diasumsikan bahwa nilai x selalu lebih besar atau sama dengan y, sehingga operasi pembagian dapat dilakukan setidaknya satu kali.Setelah itu, program menyediakan sebuah variabel penghitung bernama h yang berfungsi untuk menyimpan berapa kali nilai x dapat dikurangi oleh y. Pada awal program, nilai h diinisialisasi dengan 0, karena belum ada proses pengurangan yang dilakukan.

Program kemudian memasuki sebuah perulangan for dengan kondisi x >= y. Artinya, selama nilai x masih cukup besar untuk dikurangi dengan y, perulangan akan terus berjalan. Pada setiap iterasi perulangan, program melakukan dua langkah utama:

1. Mengurangi nilai x dengan y, yaitu x = x - y. Pengurangan ini mensimulasikan satu kali proses pembagian, karena setiap pengurangan menunjukkan bahwa y “dapat masuk” ke dalam x satu kali.
2. Menambah nilai penghitung h, yaitu h = h + 1. Penambahan ini mencatat jumlah total pengurangan yang berhasil dilakukan, yang sekaligus mewakili jumlah kali pembagi y dapat dimasukkan ke bilangan semula x.

Proses pengurangan ini berlangsung terus-menerus sampai kondisi x >= y tidak terpenuhi lagi. Ketika nilai x sudah menjadi lebih kecil dari y, program menghentikan perulangan karena y sudah tidak dapat dikurangkan lebih lanjut dari x.Setelah perulangan selesai, program menampilkan nilai h. Nilai inilah yang menjadi hasil dari operasi integer division atau hasil pembagian bulat dari x dibagi dengan y. Karena pembagian dilakukan hanya menggunakan pengurangan berulang, nilai h sepenuhnya menggambarkan berapa kali y dapat dikurangkan dari x secara utuh.

Dengan demikian, program ini bekerja sebagai simulasi proses pembagian bilangan bulat secara manual tanpa menggunakan operator pembagian, dan menghasilkan quotient yang akurat berdasarkan jumlah pengurangan yang dilakukan.