LAPORAN PRAKTIKUM ALGORITMA

DAN PEMROGRAMAN 1

MODUL 5 - 6

MODUL 5 – 6 FOR-LOOP

Disusun oleh:

MUHAMMAD FIRDAUS ARDIANSYAH

109082500126

S1IF-13-07

Asisten Praktikum

Adithana dharma putra

Apri pandu wicaksono

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

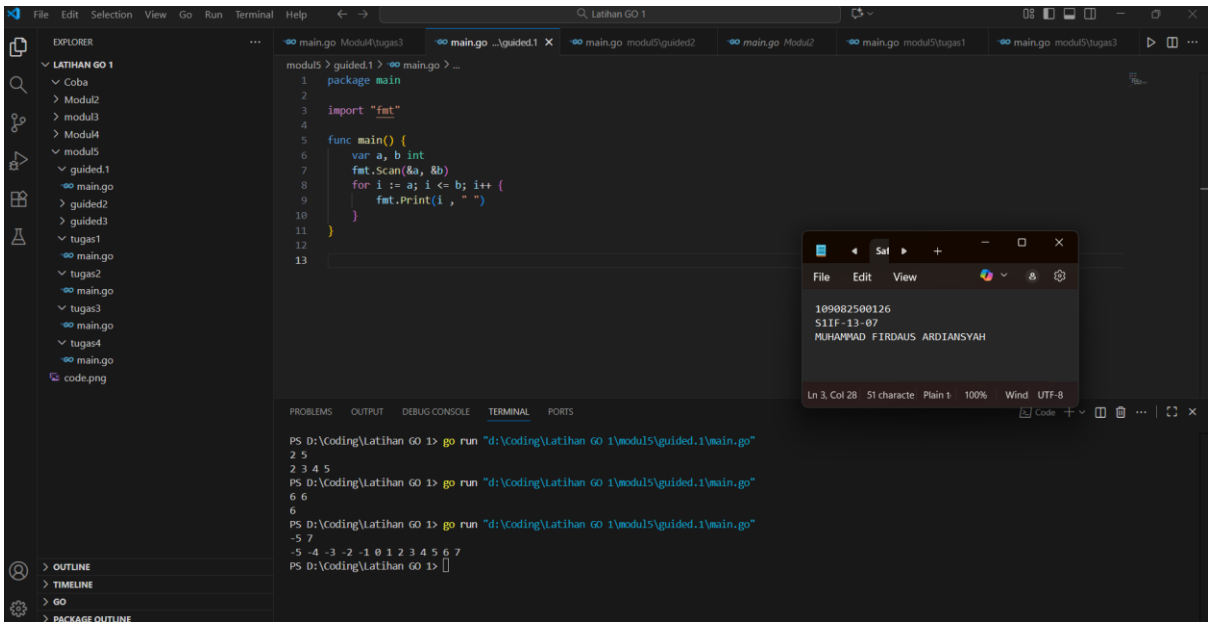
LATIHAN KELAS – GUIDED

1. Guided 1

Source Code

```
package main


import "fmt"


func main() {
    var a, b int
    fmt.Scan(&a, &b)
    for i := a; i <= b; i++ {
        fmt.Print(i , " ")
    }
}
```

Screenshoot program



Deskripsi program

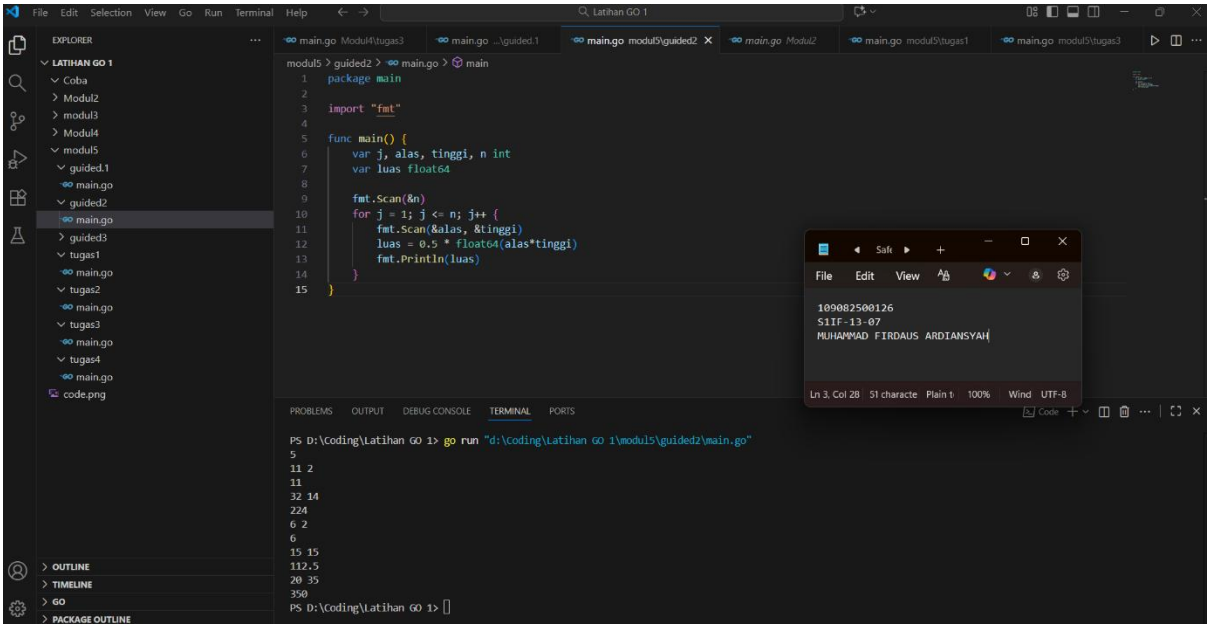
Program ini berada dalam paket main dan menggunakan fungsi main() sebagai titik awal eksekusi. Fungsi ini mendeklarasikan dua variabel integer, a dan b. Program kemudian menggunakan fmt.Scan(&a, &b) untuk membaca dua bilangan bulat yang dimasukkan oleh pengguna dari konsol dan menyimpannya dalam variabel a dan b. Setelah input diterima, program menjalankan perulangan for yang dimulai dari nilai a dan berlanjut hingga nilai i mencapai b (inklusif). Di dalam loop, fmt.Print(i, " ") digunakan untuk mencetak nilai saat ini dari i diikuti dengan spasi, yang secara efektif menampilkan semua bilangan bulat dari a hingga b secara berurutan pada satu baris. Contoh eksekusi pada terminal menunjukkan bahwa ketika inputnya adalah 2 5, outputnya adalah 2,3, 4, 5; dan ketika inputnya adalah -5 7, outputnya adalah -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7.

2. **Guided 2**

Source Code

```
package main


import "fmt"


func main() {
    var j, alas, tinggi, n int
    var luas float64


    fmt.Scan(&n)
    for j = 1; j <= n; j++ {
        fmt.Scan(&alas, &tinggi)
        luas = 0.5 * float64(alas*tinggi)
        fmt.Println(luas)
    }
}
```

Screenshoot program



Deskripsi program

Program di atas berfungsi untuk menghitung dan mencetak luas segitiga secara berulang. Program ini membaca input dari pengguna: pertama, jumlah iterasi (n) yang menentukan berapa kali perhitungan akan dilakukan; kemudian, di setiap iterasi, program membaca nilai alas (a) dan tinggi (t) dari segitiga. Perhitungan luas menggunakan rumus Luas = 0.5 x alas x tinggi, di mana alas dan tinggi dikonversi menjadi tipe data *float64* untuk hasil yang akurat, dan hasilnya dicetak ke konsol.

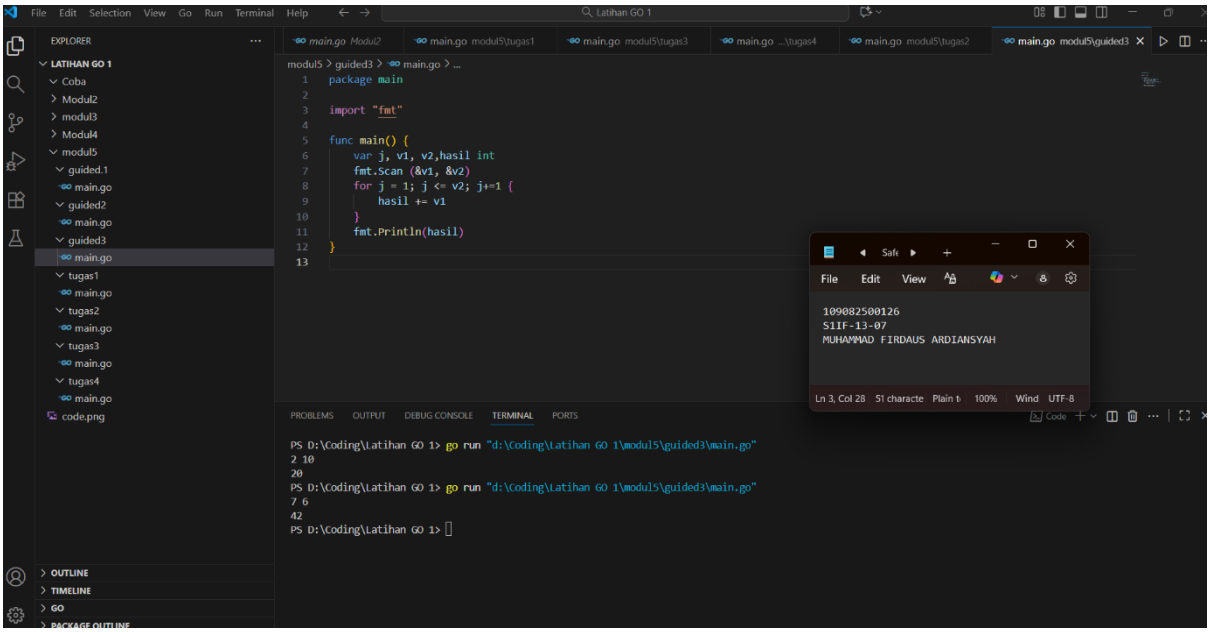
Tampak bahwa program tersebut diimplementasikan dalam fungsi main() di sebuah *package* main, menggunakan *package* fmt untuk operasi input/output. Variabel seperti j, alas, tinggi, n, dan luas dideklarasikan, dengan alas, tinggi, dan luas bertipe float64. Perulangan for dieksekusi sebanyak n kali (j dari 0 hingga n-1). Hasil akhir dari setiap perhitungan luas segitiga langsung dicetak.

Source Code

```
package main


import "fmt"


func main() {
    var j, v1, v2,hasil int
    fmt.Scan (&v1, &v2)
    for j = 1; j <= v2; j+=1 {
        hasil += v1
```

```
        }

        fmt.Println(hasil)

    }
```

Screenshoot program



Deskripsi program

Program ini berfungsi untuk menghitung hasil perkalian dua bilangan bulat secara berulang berdasarkan jumlah iterasi yang ditentukan pengguna. Awalnya, program membaca jumlah perulangan (i). Di dalam perulangan, program akan membaca dua bilangan bulat (v1 dan v2) pada setiap langkah, kemudian mengalikan keduanya (hasil = v1 x v2). Setelah semua iterasi selesai, program akan mencetak nilai hasil terakhir yang tersimpan ke konsol. Terdapat ketidaksesuaian sintaks pada kondisi perulangan for yang menggunakan variabel input (v2) sebagai batas perulangan dan memiliki cara *update* variabel yang tidak standar, meskipun secara fungsional program tetap melakukan perkalian berulang.

TUGAS

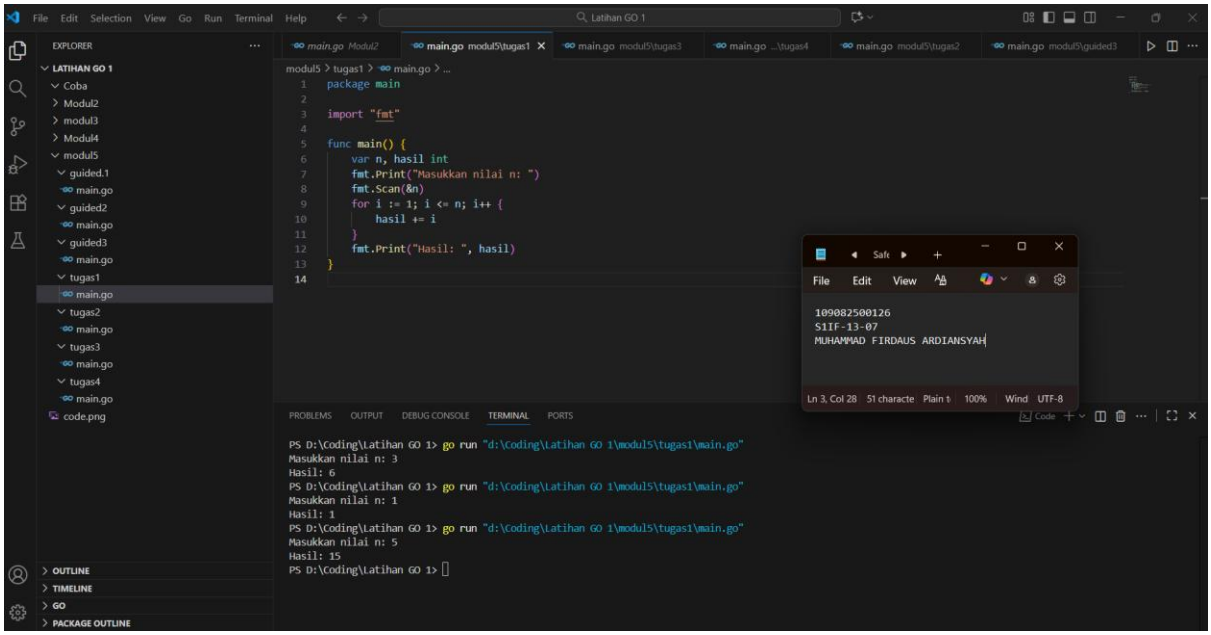
1. **Tugas 1**

Source code

```
package main
```

```
import "fmt"


func main() {
    var n, hasil int
    fmt.Print("Masukkan nilai n: ")
    fmt.Scan(&n)
    for i := 1; i <= n; i++ {
        hasil += i
    }
    fmt.Print("Hasil: ", hasil)
}
```

Screenshoot program



Deskripsi program

Fungsi dari program ini adalah untuk menghitung jumlah deret bilangan berurutan mulai dari 1 sampai dengan angka n yang diinput pengguna. Prosesnya dimulai dengan meminta kita memasukkan nilai n sebagai batas atas. Setelah itu,

program menggunakan putaran for untuk secara bertahap menambahkan (akumulasi) setiap angka bulat i ke variabel hasil, dimulai dari i=1 hingga n. Ini pada dasarnya adalah cara manual untuk menghitung rumus n(n+1)/2.

Setelah semua angka dalam deret tersebut selesai dijumlahkan, total akumulasi yang tersimpan dalam variabel hasil kemudian dicetak ke konsol. Output yang terlihat (misalnya n=3 menghasilkan 6, n=5 menghasilkan 15) menegaskan bahwa program bekerja dengan benar, berhasil menjumlahkan deret 1+2+…+n.

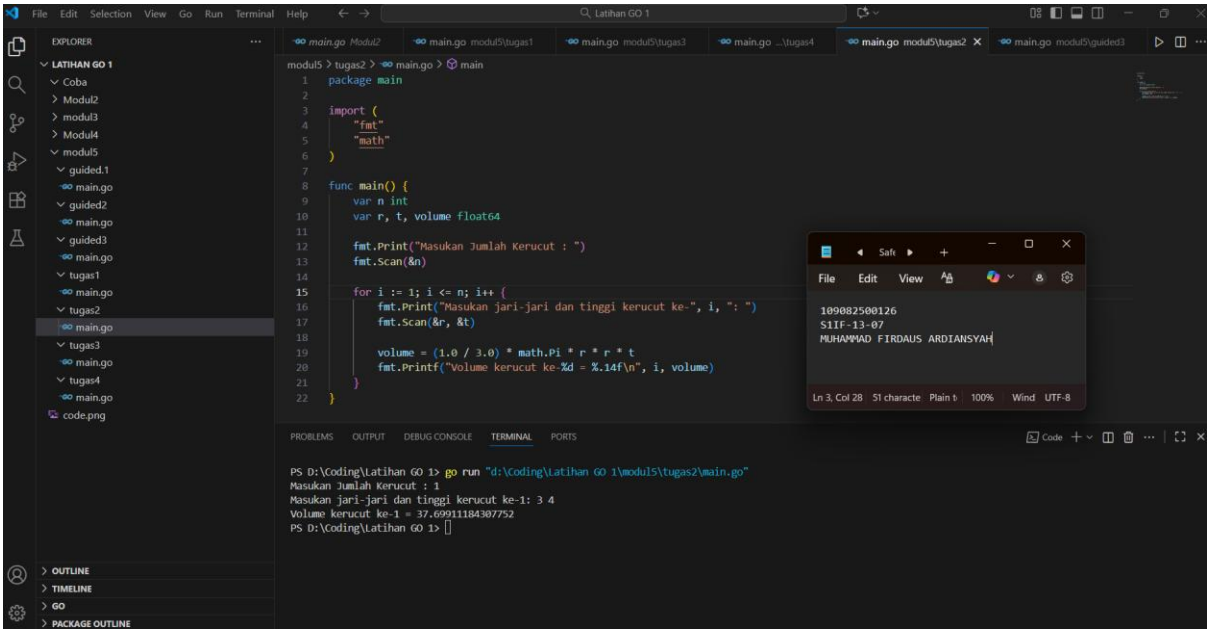
2. Tugas 2

Source code

```
package main


import (
    "fmt"
    "math"
)


func main() {
    var n int
    var r, t, volume float64


    fmt.Print("Masukan Jumlah Kerucut : ")
    fmt.Scan(&n)


    for i := 1; i <= n; i++ {
        fmt.Print("Masukan jari-jari dan tinggi kerucut
ke-", i, ": ")
        fmt.Scan(&r, &t)
```

```
            volume = (1.0 / 3.0) * math.Pi * r * r * t

            fmt.Printf("Volume kerucut ke-%d = %.14f\n", i,
    volume)

        }

    }
```

Screenshoot program



Deskripsi program

Program Go ini berfungsi untuk menghitung volume kerucut secara berulang. Awalnya, program meminta pengguna memasukkan jumlah kerucut (n) yang volumenya akan dihitung. Kemudian, program menggunakan loop for yang akan berulang sebanyak n kali. Dalam setiap iterasi, pengguna diminta memasukkan jari-jari (r) dan tinggi (t) kerucut. Program kemudian menghitung volume menggunakan rumus Volume = 1/3pi r^2 t, memanfaatkan konstanta pi dari package math dan memastikan pembagian 1/3 dilakukan sebagai operasi *float* (1.0 / 3.0) untuk hasil yang akurat.

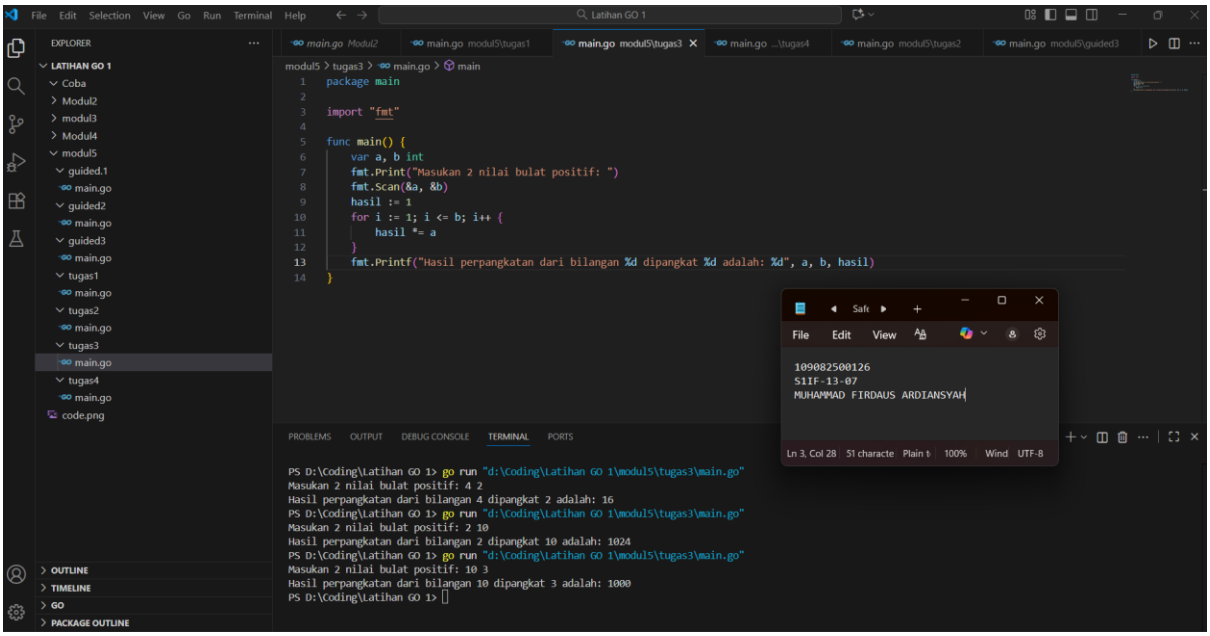
Setelah perhitungan di setiap iterasi, program mencetak volume kerucut tersebut. Variabel r, t, dan volume dideklarasikan sebagai float64 untuk menangani nilai desimal, sedangkan n (jumlah kerucut) adalah int. Output volume dicetak dengan format yang membatasi jumlah angka di belakang koma (%.14f), memastikan tampilan hasil perhitungan matematis yang presisi.

3. Tugas 3

Source code

```
package main

import "fmt"

func main() {
    var a, b int
    fmt.Print("Masukan 2 nilai bulat positif: ")
    fmt.Scan(&a, &b)
    hasil := 1
    for i := 1; i <= b; i++ {
        hasil *= a
    }
    fmt.Printf("Hasil perpangkatan dari bilangan %d
dipangkat %d adalah: %d", a, b, hasil)
}
```

Screenshoot program



Deskripsi program

Program ini dirancang untuk menghitung hasil perpangkatan dari dua bilangan bulat positif yang diinput pengguna. Program akan meminta *user* memasukkan dua nilai, yaitu basis (a) dan eksponen (b). Setelah menerima input, program

menggunakan perulangan for yang beroperasi sebanyak b kali untuk melakukan perkalian berulang basis (a). Hasil perkalian diakumulasikan dalam variabel hasil, yang diinisialisasi menjadi 1 sebelum perulangan dimulai.

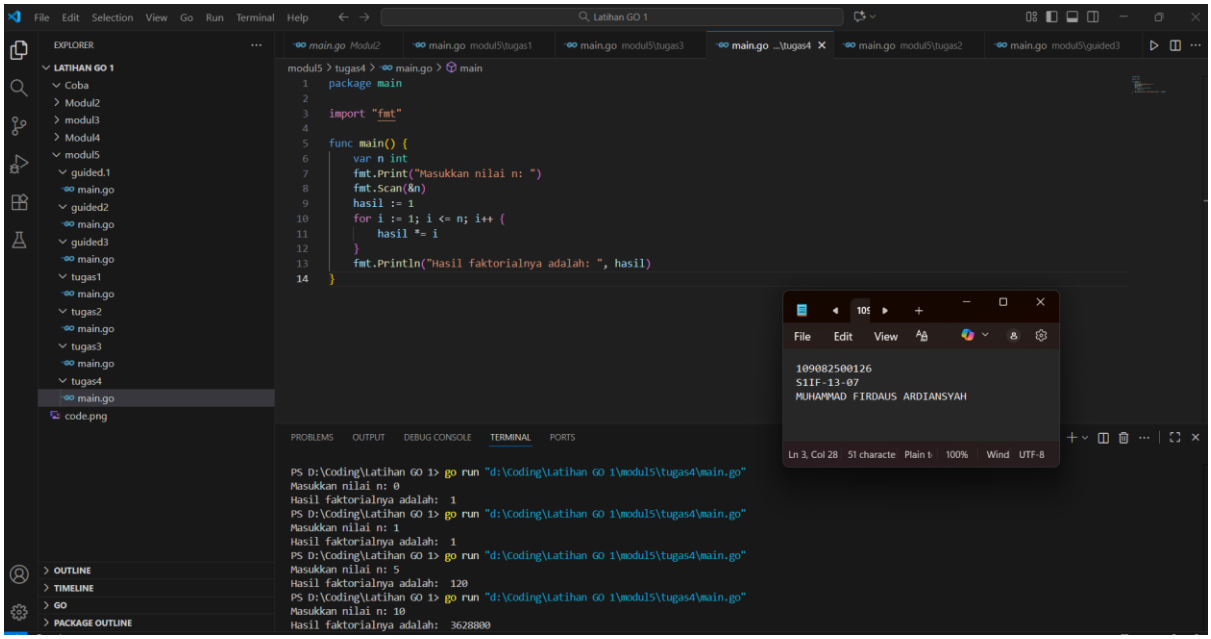
Secara teknis, variabel a, b, dan hasil semuanya dideklarasikan sebagai integer (int). Operasi inti terletak pada baris hasil *= a, yang merupakan singkatan dari hasil = hasil * a, memastikan bahwa a dikalikan dengan dirinya sendiri sebanyak b kali. Setelah perulangan selesai, program mencetak hasilnya dalam format yang jelas, menunjukkan bilangan basis, eksponen, dan nilai hasil perpangkatannya, seperti yang terlihat pada *output* konsol (2^4=16 atau 10^3=1000).

4. Tugas 4

Source code

```
package main

import "fmt"

func main() {
    var n int
    fmt.Print("Masukkan nilai n: ")
    fmt.Scan(&n)
    hasil := 1
    for i := 1; i <= n; i++ {
        hasil *= i
    }
    fmt.Println("Hasil faktorialnya adalah: ", hasil)
}
```

Screenshoot program



Deskripsi program

Fungsi utama dari program Go ini adalah untuk menghitung nilai faktorial (n!) dari bilangan bulat non-negatif n yang dimasukkan oleh pengguna. Program akan meminta kita memasukkan nilai n. Setelah input diterima, program menggunakan perulangan for yang beroperasi dari i=1 hingga n. Di dalam loop, variabel hasil yang diinisialisasi menjadi 1 akan terus dikalikan dengan nilai i saat ini (hasil *= i). Proses perkalian berulang ini adalah definisi dari faktorial: n! = 1 x 2 x 3 x … x n.

Variabel n dan hasil dideklarasikan sebagai integer (int). Program ini menangani kasus dasar faktorial dengan benar, di mana faktorial dari 0 dan 1 sama dengan 1, sesuai dengan inisialisasi hasil = 1 sebelum loop dimulai. Setelah loop selesai, program akan mencetak nilai faktorial yang telah dihitung ke konsol, seperti terlihat pada *output* (misalnya, 5! = 120 dan 10! = 362880).