LAPORAN PRAKTIKUM ALGORITMA

DAN PEMROGRAMAN 1

MODUL 5 & 6

FOR-LOOP

Disusun oleh:

Anindya Rahadita Yumnaa

109082500138

S1IF-13-07

Asisten Praktikum

Adithana dharma putra

Apri pandu wicaksono

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

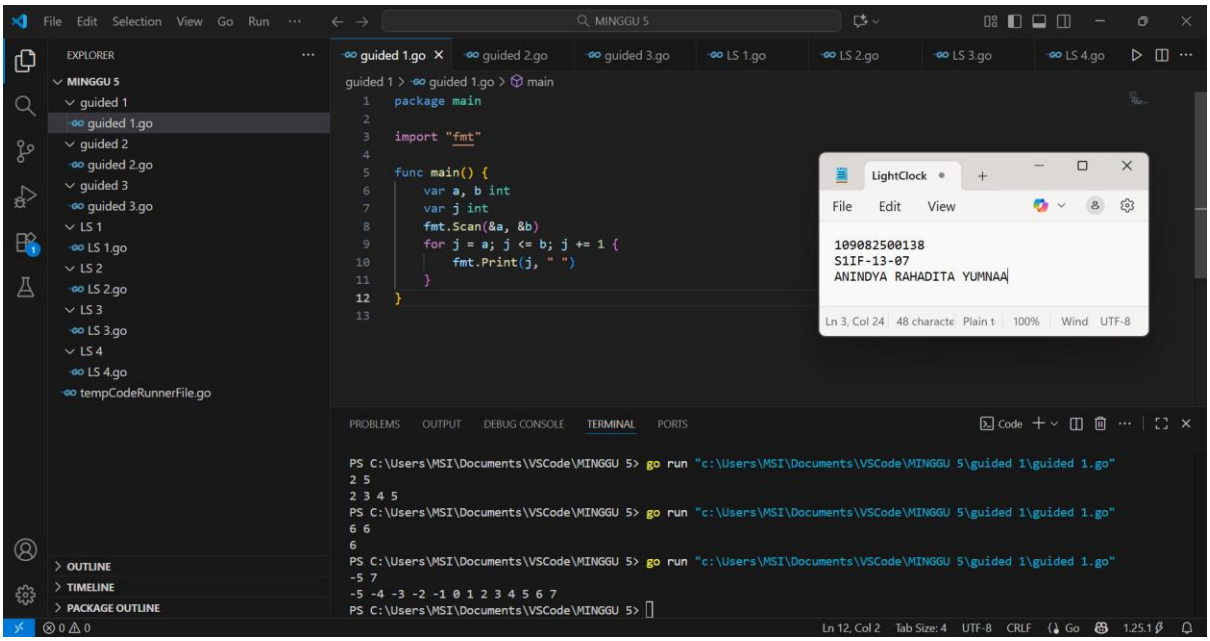
LATIHAN KELAS – GUIDED

1. Guided 1

Source Code

```
package main


import "fmt"


func main() {
    var a, b int
    var j int
    fmt.Scan(&a, &b)
    for j = a; j <= b; j += 1 {
        fmt.Print(j, " ")
    }
}
```

Screenshoot program



Deskripsi program

1. package main

Program ini menandakan bahwa file tersebut adalah **program utama** yang bisa dijalankan (bukan library). Library sendiri itu adalah kode bantu yang dapat berjalan hanya jika dipanggil oleh kode atau program utama, yaitu **package main**. Jika sebuah paket diberi nama main, tool Go akan tahu jika tujuan akhirnya adalah membuat program yang bisa dijalankan.

2. import "fmt"

import sendiri berfungsi untuk memanggil kode lain agar fungsinya bisa dipakai di file tersebut. Kemudian, fmt adalah paket standar Go untuk **formatting** dan operasi input atau output sederhana (untuk menulis ke layar dan membaca dari input).

3. func main() { ... }

func main() adalah **titik masuk** program, kode di dalam func main() dijalankan pertama kali setelah kode atau program diberi nama. Bentuknya harus **"func main()"** tanpa parameter dan tanpa nilai balik.

4. var a, b int

Deklarasi Variabel a dan b. Mendeklarasikan dua variabel: a dan b, dengan tipe data **integer** (int). Kedua variabel ini akan menampung batas awal dan batas akhir deret.

5. var j int

Deklarasi Variabel j. Mendeklarasikan variabel j (biasanya digunakan sebagai *iterator* atau pencacah), dengan tipe data **integer** (int).

6. fmt.Scan(&a, &b)

Input Variabel a dan b. Membaca dua angka dari masukan pengguna. Angka pertama disimpan ke variabel a (batas awal), dan angka kedua disimpan ke variabel b (batas akhir). Simbol & menunjukkan bahwa data disimpan langsung ke lokasi memori variabel

7. for j = a; j <= b; j += 1 {

Perulangan (*For Loop*). Struktur kontrol perulangan yang akan mengulang kode di dalamnya. Perulangan dimulai dengan j sama dengan nilai a, terus berlanjut selama j **kurang dari atau sama dengan** nilai b, dan setiap iterasi j akan **bertambah 1** (j += 1).

8. fmt.Print(j, " ")

Cetak Bilangan. Mencetak nilai variabel j saat ini (yaitu bilangan dalam deret) diikuti dengan satu **spasi** (" "). Karena menggunakan fmt.Print, semua bilangan akan dicetak dalam satu baris.

9. }

Akhir Perulangan. Menutup blok kode perulangan for.

10. }

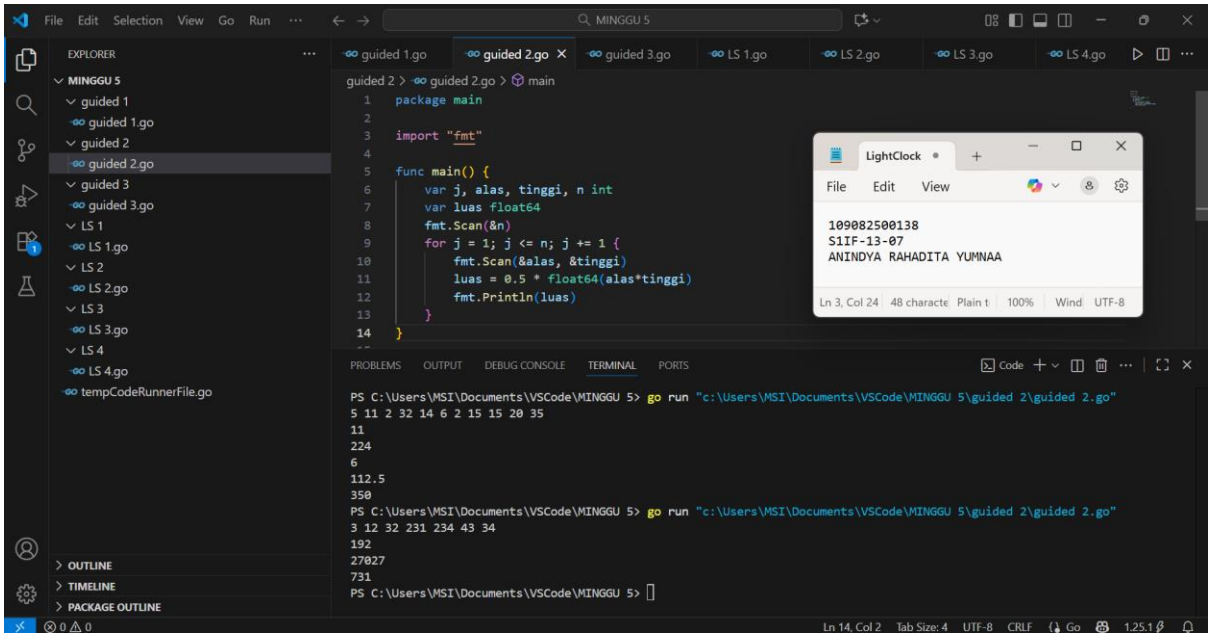
Akhir Fungsi. Menutup blok kode fungsi main()

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

Screenshoot program



Deskripsi program

1. package main

Program ini menandakan bahwa file tersebut adalah **program utama** yang bisa dijalankan (bukan library). Library sendiri itu adalah kode bantu yang dapat berjalan hanya jika dipanggil oleh kode atau program utama, yaitu **package main**. Jika sebuah paket diberi nama main, tool Go akan tahu jika tujuan akhirnya adalah membuat program yang bisa dijalankan.

2. import "fmt"

import sendiri berfungsi untuk memanggil kode lain agar fungsinya bisa dipakai di file tersebut. Kemudian, fmt adalah paket standar Go untuk **formatting** dan operasi input atau output sederhana (untuk menulis ke layar dan membaca dari input).

3. func main() { ... }

func main() adalah **titik masuk** program, kode di dalam func main() dijalankan pertama kali setelah kode atau program diberi nama. Bentuknya harus **"func main()"** tanpa parameter dan tanpa nilai balik.

4. var j, alas, tinggi, n int

Deklarasi Variabel Integer. Mendeklarasikan empat variabel: j (pencacah), alas, tinggi, dan n (jumlah iterasi), semuanya bertipe **integer** (int).

5. var luas float64

Deklarasi Variabel *float*. Mendeklarasikan variabel luas yang akan menyimpan hasil perhitungan volume. Menggunakan tipe float64 untuk hasil desimal.

6. fmt.Scan(&n)

Input Jumlah Iterasi. Membaca satu angka dari masukan pengguna, yang disimpan ke variabel n. Angka ini menentukan berapa kali perhitungan luas segitiga akan dilakukan.

7. for j = 1; j <= n; j += 1 {
Perulangan (*For Loop*). Struktur perulangan yang akan berjalan sebanyak n kali. Dimulai dari j sama dengan 1 dan berakhir saat j mencapai n.

8. fmt.Scan(&alas, &tinggi)
Input Alas dan Tinggi. Di setiap iterasi, program membaca dua angka dari masukan: angka pertama disimpan ke alas dan angka kedua disimpan ke tinggi.

9. luas = 0.5 * float64(alas*tinggi)
Perhitungan Luas Segitiga. Menghitung luas segitiga dengan ***rumus: 0.5 x alas x tinggi.*** Hasil perkalian alas*tinggi dikonversi ke float64 sebelum dikalikan dengan 0.5 agar hasil akhir berupa bilangan desimal.

10. fmt.Println(luas)
Cetak Luas. Mencetak hasil perhitungan luas ke layar, diikuti dengan baris baru.

11. }
Akhir Perulangan. Menutup blok kode perulangan for.

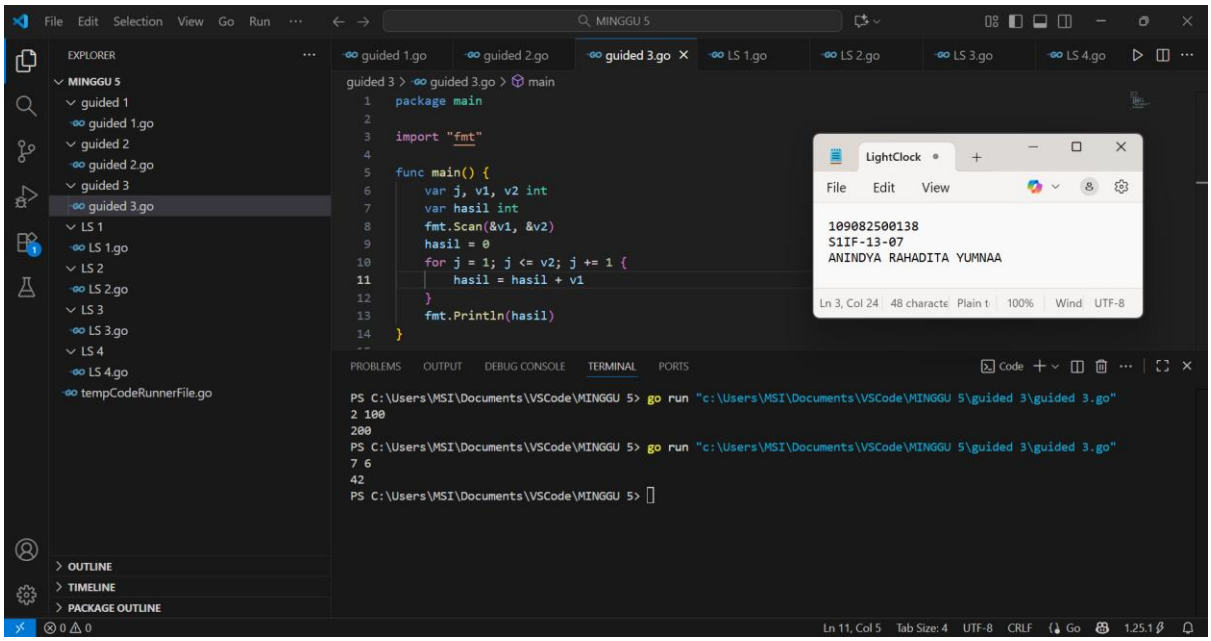
12. }
Akhir Fungsi. Menutup blok kode fungsi main().

3. **Guided 3**
Source Code

```
package main


import "fmt"


func main() {
    var j, v1, v2 int
    var hasil int
    fmt.Scan(&v1, &v2)
```

```
        hasil = 0

        for j = 1; j <= v2; j += 1 {

            hasil = hasil + v1

        }

        fmt.Println(hasil)

    }
```

Screenshoot program



Deskripsi program

1. package main

Program ini menandakan bahwa file tersebut adalah **program utama** yang bisa dijalankan (bukan library). Library sendiri itu adalah kode bantu yang dapat berjalan hanya jika dipanggil oleh kode atau program utama, yaitu **package main**. Jika sebuah paket diberi nama main, tool Go akan tahu jika tujuan akhirnya adalah membuat program yang bisa dijalankan.

2. import "fmt"

import sendiri berfungsi untuk memanggil kode lain agar fungsinya bisa dipakai di file tersebut. Kemudian, fmt adalah paket standar Go untuk **formatting** dan operasi input atau output sederhana (untuk menulis ke layar dan membaca dari input).

3. func main() { ... }
func main() adalah **titik masuk** program, kode di dalam func main() dijalankan pertama kali setelah kode atau program diberi nama. Bentuknya harus **"func main()"** tanpa parameter dan tanpa nilai balik.

4. var j, v1, v2 int
Deklarasi Variabel Integer. Mendeklarasikan tiga variabel: j (pencacah), v1 (bilangan pertama/basis), dan v2 (bilangan kedua/pengali), semuanya bertipe **integer** (int).

5. var hasil int
Deklarasi Variabel Hasil. Mendeklarasikan variabel hasil yang akan menyimpan hasil akhir perhitungan, bertipe **integer** (int).

6. fmt.Scan(&v1, &v2)
Input Dua Bilangan. Membaca dua angka dari masukan pengguna: angka pertama disimpan ke v1, dan angka kedua disimpan ke v2.

7. hasil = 0
Inisialisasi Hasil. Menyetel nilai awal variabel hasil menjadi nol (0), sebelum memulai penjumlahan.

8. for j = 1; j <= v2; j += 1 {
Perulangan (*For Loop*). Perulangan akan berjalan sebanyak v2 kali. Dimulai dari j=1 dan berlanjut selama j kurang dari atau sama dengan v2.

9. hasil = hasil + v1
Operasi Penjumlahan Berulang. Di setiap iterasi, nilai v1 (basis) ditambahkan ke variabel hasil. Ini secara efektif mengimplementasikan perkalian ***v1 x v2.***

10. }
Akhir Perulangan. Menutup blok kode perulangan for.

11. fmt.Println(hasil)
Cetak Hasil. Mencetak nilai akhir dari variabel hasil ke layar, diikuti dengan baris baru.

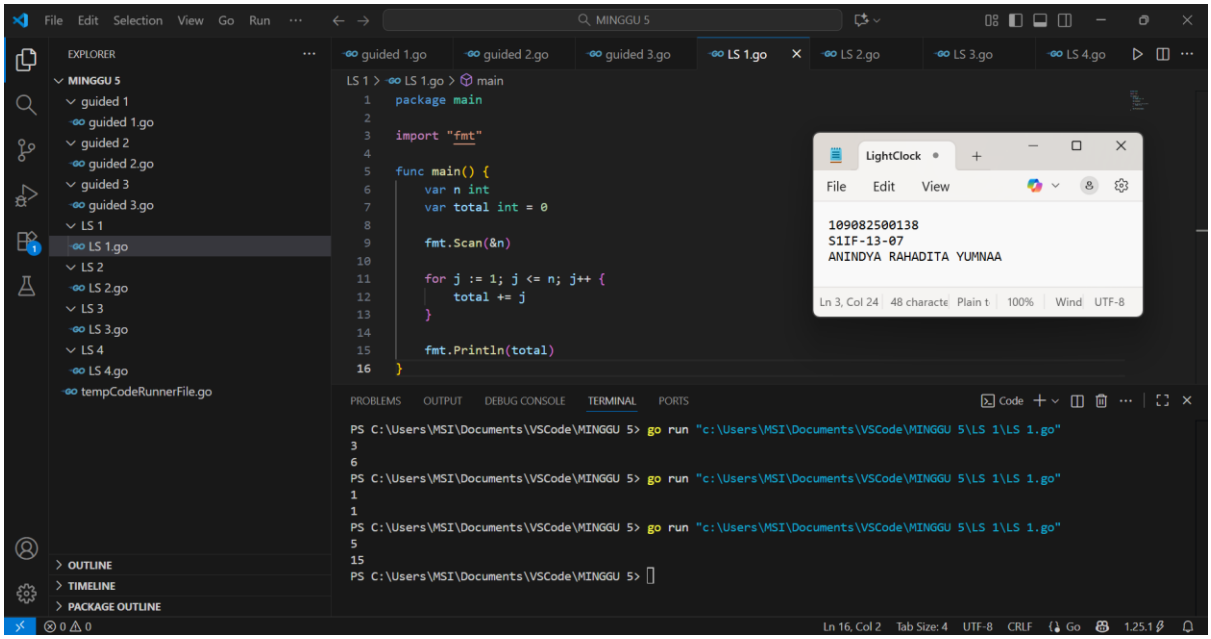
12. }
Akhir Fungsi. Menutup blok kode fungsi main().

TUGAS

1. **Tugas 1**
   Source code

```go
package main


import "fmt"


func main() {
    var n int
    var total int = 0


    fmt.Scan(&n)


    for j := 1; j <= n; j++ {
        total += j
    }


    fmt.Println(total)
}
```

Screenshoot program



Deskripsi program

1. package main

Program ini menandakan bahwa file tersebut adalah **program utama** yang bisa dijalankan (bukan library). Library sendiri itu adalah kode bantu yang dapat berjalan hanya jika dipanggil oleh kode atau program utama, yaitu **package main**. Jika sebuah paket diberi nama main, tool Go akan tahu jika tujuan akhirnya adalah membuat program yang bisa dijalankan.

2. import "fmt"

import sendiri berfungsi untuk memanggil kode lain agar fungsinya bisa dipakai di file tersebut. Kemudian, fmt adalah paket standar Go untuk **formatting** dan operasi input atau output sederhana (untuk menulis ke layar dan membaca dari input).

3. func main() { ... }

func main() adalah **titik masuk** program, kode di dalam func main() dijalankan pertama kali setelah kode atau program diberi nama. Bentuknya harus **"func main()"** tanpa parameter dan tanpa nilai balik.

4. var n int

Deklarasi Variabel n. Mendeklarasikan variabel n (batas akhir deret) bertipe **integer** (int).

5. var total int = 0

Deklarasi dan Inisialisasi Total. Mendeklarasikan variabel total yang akan menyimpan hasil penjumlahan, bertipe **integer** (int), dan langsung diinisialisasi dengan nilai nol (0).

6. fmt.Scan(&n)

Input Batas Akhir (N). Membaca satu angka dari masukan pengguna dan menyimpannya ke variabel n.

7. for j := 1; j <= n; j++ {

Perulangan dimulai. Perintah ini memberitahu program untuk **mengulang** langkah berikutnya. Perulangan dimulai dengan j bernilai 1, dan akan terus berjalan **sampai** j melebihi nilai n. Setiap kali selesai satu putaran, nilai j akan **bertambah satu**.

8. total += j

Operasi Penjumlahan. Menambahkan nilai pencacah j saat ini ke variabel total. Ini setara dengan total = total + j.

9. }

Akhir Perulangan. Menutup blok kode perulangan for.

10. fmt.Println(total)

Cetak Hasil. Mencetak nilai akhir dari variabel total ke layar, diikuti dengan baris baru.

11. }
Akhir Fungsi. Menutup blok kode fungsi main().

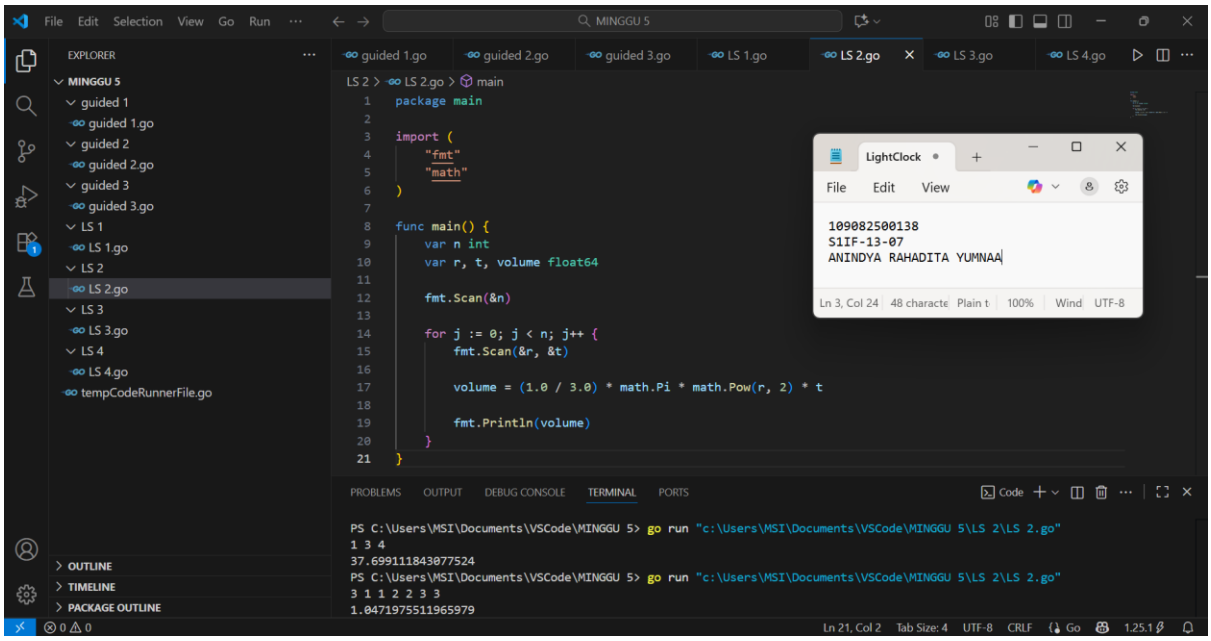
2. Tugas 2

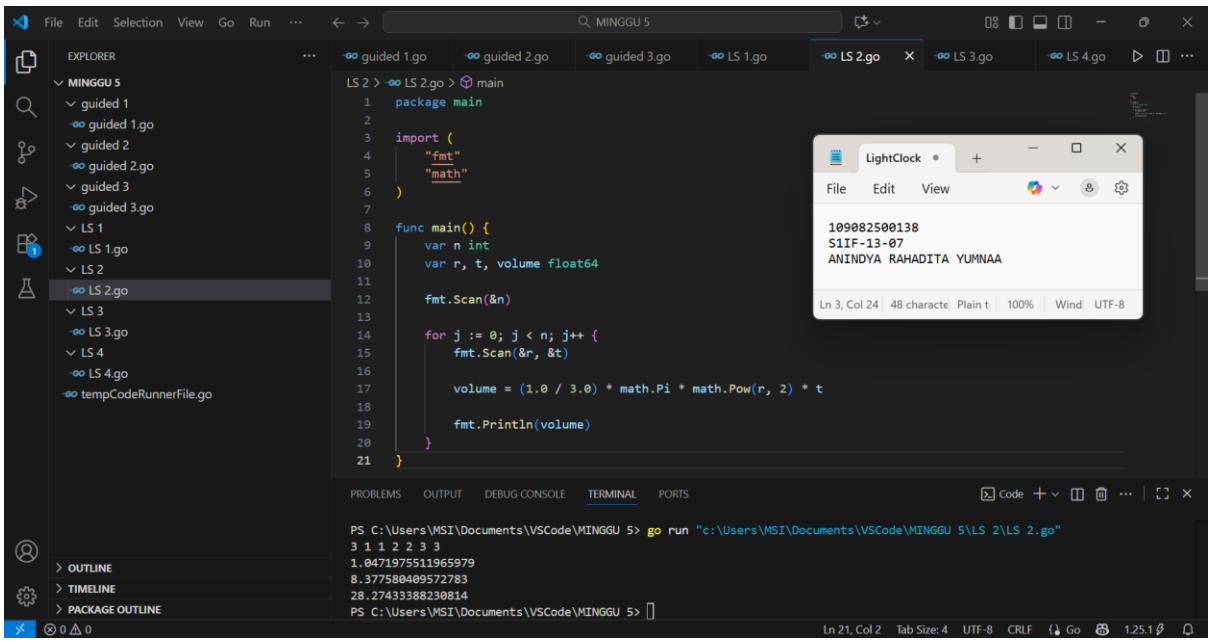
Source code

```
package main


import (
    "fmt"
    "math"
)


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


    fmt.Scan(&n)


    for j := 0; j < n; j++ {
        fmt.Scan(&r, &t)


        volume = (1.0 / 3.0) * math.Pi * math.Pow(r, 2)
* t


        fmt.Println(volume)
    }
```

```
}
```

Screenshoot program





Deskripsi program

1. package main

Program ini menandakan bahwa file tersebut adalah **program utama** yang bisa dijalankan (bukan library). Library sendiri itu adalah kode bantu yang dapat berjalan hanya jika dipanggil oleh kode atau program utama, yaitu **package main**. Jika sebuah paket diberi nama main, tool Go akan tahu jika tujuan akhirnya adalah membuat program yang bisa dijalankan.

2. import (
Awal Blok Impor. Memulai blok untuk mengimpor lebih dari satu paket.

3. "fmt"
Impor Paket *fmt*. Mengimpor paket standar Go untuk operasi *formatting*, input, dan output (Scan, Println).

4. "math"
Impor Paket *math*. Mengimpor paket standar Go yang berisi fungsi dan konstanta matematika, seperti nilai ***pi*** (math.Pi) dan fungsi pangkat (math.Pow).

5.)
Akhir Blok Impor. Menutup blok impor paket.

6. func main() {
Fungsi Utama (*main*). Mendefinisikan titik masuk program. Kode di dalamnya dijalankan pertama kali.

7. var n int
Deklarasi Variabel n. Mendeklarasikan variabel n (jumlah kerucut yang akan dihitung) bertipe **integer** (int).

8. var r, t, volume float64
Deklarasi Variabel *Float*. Mendeklarasikan variabel r (jari-jari), t (tinggi), dan volume. Semua bertipe float64 karena perhitungan volume melibatkan bilangan desimal (pi).

9. fmt.Scan(&n)
Input Jumlah Kerucut (N). Membaca satu angka dari masukan pengguna dan menyimpannya ke variabel n.

10. for j := 0; j < n; j++ {
Perulangan (*For Loop*). Perulangan akan berjalan sebanyak n kali (dari j=0 hingga j=n-1). j adalah pencacah yang bertambah 1 di setiap iterasi.

11. fmt.Scan(&r, &t)
Input Jari-jari (r) dan Tinggi (t). Di setiap iterasi, program membaca dua angka dari masukan: angka pertama disimpan ke r dan angka kedua disimpan ke t.

12. volume = (1.0 / 3.0) * math.Pi * math.Pow(r, 2) * t
Baris ini **menghitung volume kerucut** berdasarkan rumusnya. Variabel volume diisi dengan hasil dari: **1/3 dikali nilai pi** (math.Pi), dikali jari-jari (r) yang dipangkatkan dua (math.Pow(r, 2)), dan dikali tinggi (t).

13. fmt.Println(volume)
Cetak Volume. Mencetak hasil perhitungan volume ke layar, diikuti dengan baris baru.

14. }
Akhir Perulangan. Menutup blok kode perulangan for.

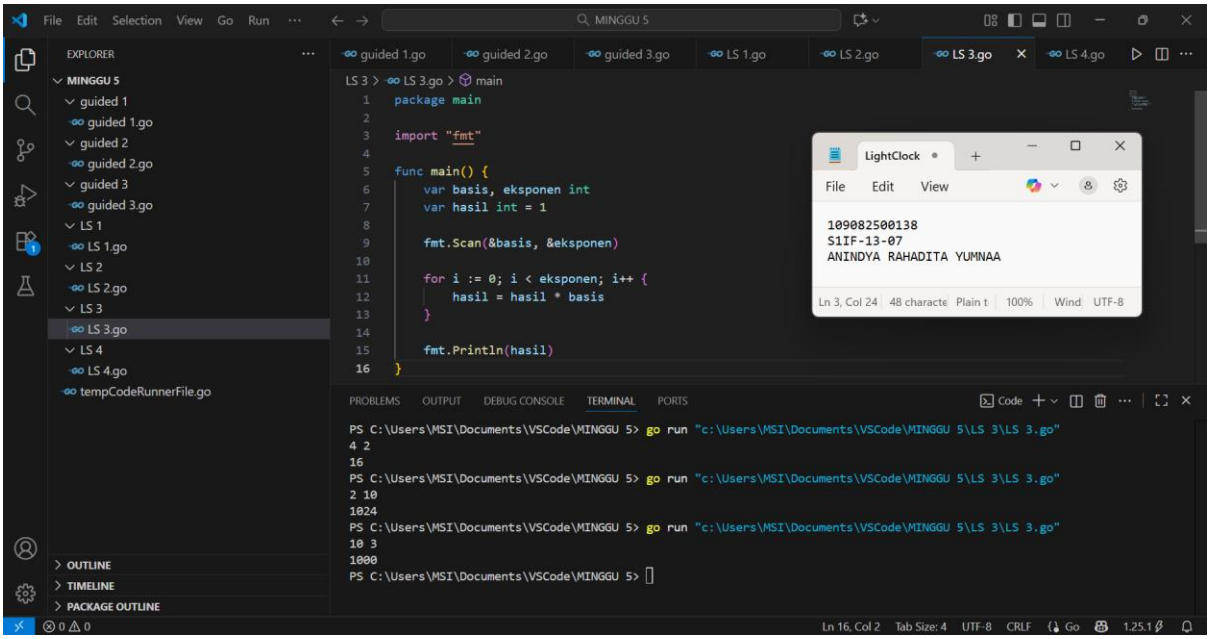
15. }
Akhir Fungsi. Menutup blok kode fungsi main().

3. Tugas 3

Source code

```
package main

import "fmt"

func main() {
    var basis, eksponen int
    var hasil int = 1

    fmt.Scan(&basis, &eksponen)

    for i := 0; i < eksponen; i++ {
        hasil = hasil * basis
    }

    fmt.Println(hasil)
}
```

Screenshoot program



Deskripsi program

1. package main

Program ini menandakan bahwa file tersebut adalah **program utama** yang bisa dijalankan (bukan library). Library sendiri itu adalah kode bantu yang dapat berjalan hanya jika dipanggil oleh kode atau program utama, yaitu **package main**. Jika sebuah paket diberi nama main, tool Go akan tahu jika tujuan akhirnya adalah membuat program yang bisa dijalankan.

2. import "fmt"

import sendiri berfungsi untuk memanggil kode lain agar fungsinya bisa dipakai di file tersebut. Kemudian, fmt adalah paket standar Go untuk **formatting** dan operasi input atau output sederhana (untuk menulis ke layar dan membaca dari input).

3. func main() { ... }

func main() adalah **titik masuk** program, kode di dalam func main() dijalankan pertama kali setelah kode atau program diberi nama. Bentuknya harus **"func main()"** tanpa parameter dan tanpa nilai balik.

4. var basis, eksponen int

Deklarasi Variabel Integer. Mendeklarasikan dua variabel: basis (bilangan yang akan dipangkatkan) dan eksponen (pangkat), keduanya bertipe **integer** (int).

5. var hasil int = 1

Deklarasi dan Inisialisasi Hasil. Mendeklarasikan variabel hasil dan langsung disetel nilainya menjadi **1**. Ini adalah nilai awal yang benar karena bilangan apa pun dipangkatkan 0 adalah 1, dan perkalian harus dimulai dari 1.

6. fmt.Scan(&basis, &eksponen)

Input Dua Bilangan. Membaca dua angka dari masukan pengguna: angka pertama disimpan ke basis, dan angka kedua disimpan ke eksponen.

7. for i := 0; i < eksponen; i++ {
Perulangan (*For Loop*). Struktur perulangan untuk melakukan perkalian berulang. Perulangan akan berjalan sebanyak **nilai dari eksponen** kali (dari i=0 hingga i=eksponen-1).

8. hasil = hasil * basis
Operasi Perkalian Berulang. Di setiap iterasi, nilai hasil yang ada dikalikan dengan basis. Ini secara efektif menghitung pemangkatan.

9. }
Akhir Perulangan. Menutup blok kode perulangan for.

10. fmt.Println(hasil)
Cetak Hasil. Mencetak nilai akhir dari variabel hasil (hasil pemangkatan) ke layar.

11. }
Akhir Fungsi. Menutup blok kode fungsi main().

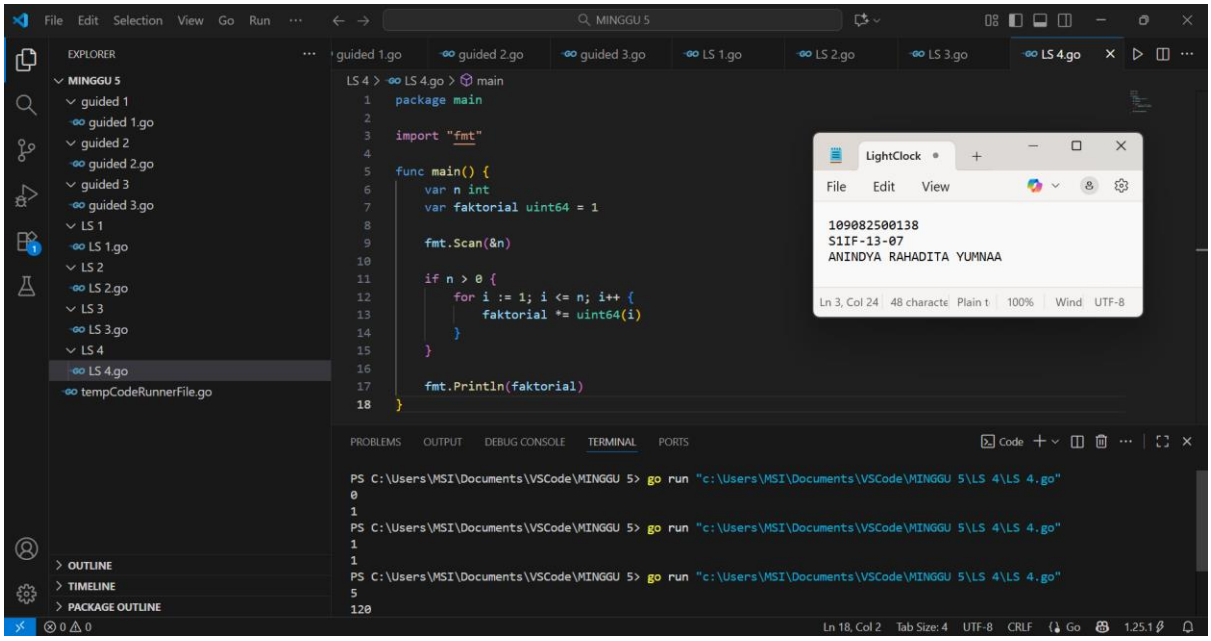
4. Tugas 4

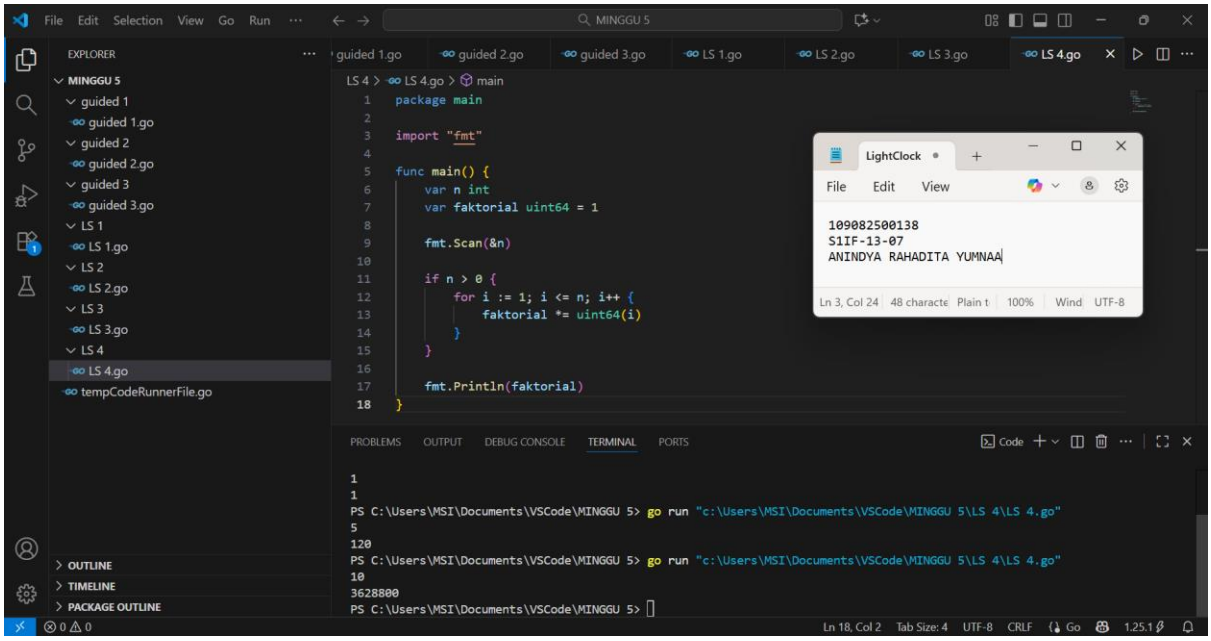
Source code

```
package main

import "fmt"

func main() {
    var n int
    var faktorial uint64 = 1

    fmt.Scan(&n)

    if n > 0 {
        for i := 1; i <= n; i++ {
            faktorial *= uint64(i)
        }
    }
```

```
            fmt.Println(faktorial)
        }
```

Screenshoot program





Deskripsi program

1. package main

Program ini menandakan bahwa file tersebut adalah **program utama** yang bisa dijalankan (bukan library). Library sendiri itu adalah kode bantu yang dapat berjalan hanya jika dipanggil oleh kode atau program utama, yaitu **package main**. Jika sebuah

paket diberi nama main, tool Go akan tahu jika tujuan akhirnya adalah membuat program yang bisa dijalankan.

2. import "fmt"

import sendiri berfungsi untuk memanggil kode lain agar fungsinya bisa dipakai di file tersebut. Kemudian, fmt adalah paket standar Go untuk **formatting** dan operasi input atau output sederhana (untuk menulis ke layar dan membaca dari input).

3. func main() { ... }

func main() adalah **titik masuk** program, kode di dalam func main() dijalankan pertama kali setelah kode atau program diberi nama. Bentuknya harus **"func main()"** tanpa parameter dan tanpa nilai balik.

4. var n int

Deklarasi Variabel n. Mendeklarasikan variabel n (bilangan yang akan dicari faktorialnya) bertipe **integer** (int).

5. var faktorial uint64 = 1

Deklarasi dan Inisialisasi Faktorial. Mendeklarasikan variabel faktorial dengan tipe **unsigned integer 64-bit** (uint64), yang dapat menampung nilai faktorial yang sangat besar. Variabel ini diinisialisasi ke **1**, karena $0! = 1$ dan $1! = 1$.

6. fmt.Scan(&n)

Input Bilangan (N). Membaca satu angka dari masukan pengguna dan menyimpannya ke variabel n.

7. if n > 0 {

Pengecekan Kondisi. Memulai blok kondisional (*if*) yang hanya akan menjalankan perhitungan faktorial **jika** nilai n lebih besar dari 0. (Jika n adalah 0, hasil akan tetap 1 sesuai inisialisasi di baris 7).

8. for i := 1; i <= n; i++ {

Perulangan (*For Loop*). Struktur perulangan untuk mengalikan bilangan dari 1 hingga ***n.***

9. faktorial *= uint64(i)

Operasi Perkalian. Nilai faktorial saat ini dikalikan dengan nilai pencacah i. Nilai i dikonversi ke uint64 agar sesuai dengan tipe data faktorial sebelum dikalikan.

10. }

Akhir Perulangan. Menutup blok kode perulangan for.

11. }

Akhir Kondisi If. Menutup blok kode kondisional if.

12. fmt.Println(faktorial)

Cetak Hasil. Mencetak nilai akhir dari variabel faktorial ke layar.

13. }

Akhir Fungsi. Menutup blok kode fungsi main().