

**LAPORAN PRAKTIKUM ALGORITMA  
DAN PEMROGRAMAN 1**

**MODUL 3 1/0  
TIPE DATA & VARIABEL**



**Disusun oleh:**

**Didi Hermawanto**

**109082500088**

**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

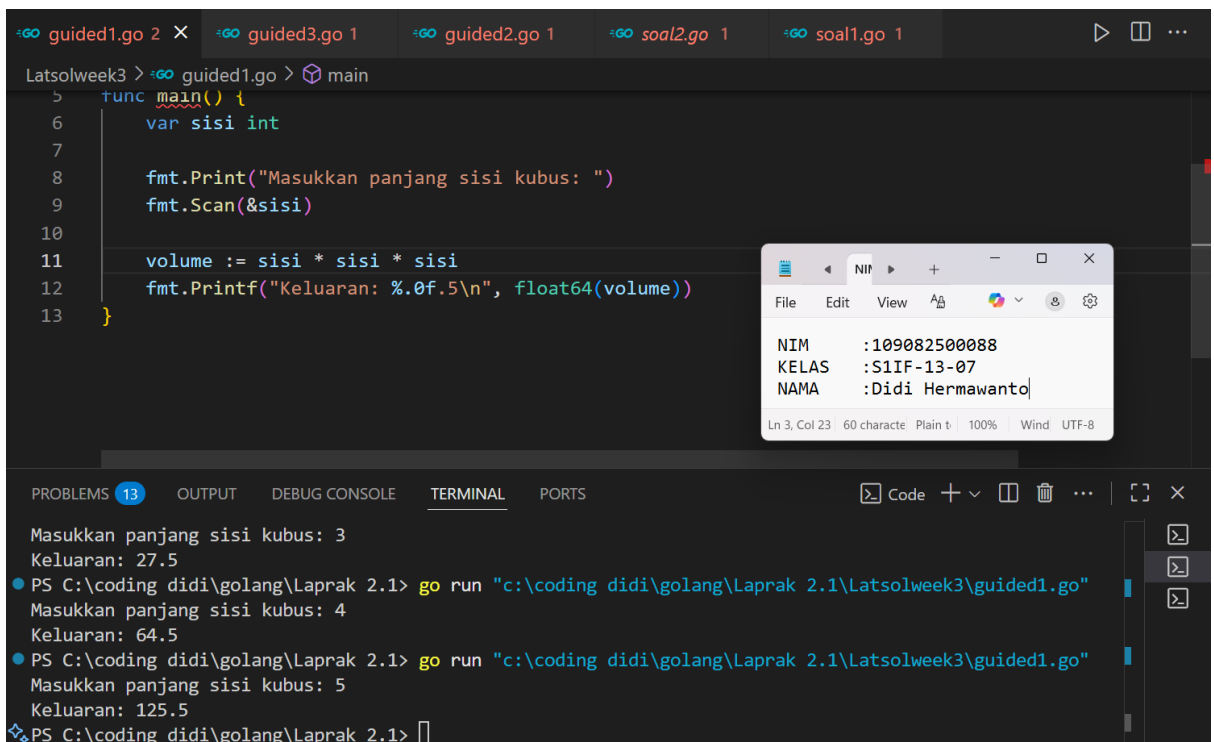
import "fmt"

func main() {
    var sisi int

    fmt.Print("Masukkan panjang sisi kubus: ")
    fmt.Scan(&sisi)

    volume := sisi * sisi * sisi
    fmt.Printf("Keluaran: %.0f.5\n", float64(volume))
}
```

#### Screenshoot program :



The screenshot shows the VS Code editor with the following Go code in `guided1.go`:

```
1 func main() {
2     var sisi int
3
4     fmt.Print("Masukkan panjang sisi kubus: ")
5     fmt.Scan(&sisi)
6
7     volume := sisi * sisi * sisi
8     fmt.Printf("Keluaran: %.0f.5\n", float64(volume))
9 }
```

The terminal output shows the program being run three times with different inputs:

```
PS C:\coding didi\golang\Laprak 2.1> go run "c:\coding didi\golang\Laprak 2.1\Latsolweek3\guided1.go"
Masukkan panjang sisi kubus: 3
Keluaran: 27.5
PS C:\coding didi\golang\Laprak 2.1> go run "c:\coding didi\golang\Laprak 2.1\Latsolweek3\guided1.go"
Masukkan panjang sisi kubus: 4
Keluaran: 64.5
PS C:\coding didi\golang\Laprak 2.1> go run "c:\coding didi\golang\Laprak 2.1\Latsolweek3\guided1.go"
Masukkan panjang sisi kubus: 5
Keluaran: 125.5
PS C:\coding didi\golang\Laprak 2.1>
```

A small window in the background displays the following information:

|       |                  |
|-------|------------------|
| NIM   | :109082500088    |
| KELAS | :S1IF-13-07      |
| NAMA  | :Didi Hermawanto |

## Deskripsi program

### Alur Program

#### 1. Deklarasi Variabel

`var sisi int`

Variabel sisi bertipe **integer** digunakan untuk menyimpan panjang sisi kubus yang dimasukkan oleh pengguna.

#### 2. Input dari Pengguna

```
fmt.Print("Masukkan panjang sisi kubus: ")
```

```
fmt.Scan(&sisi)
```

- Program menampilkan pesan ke layar: *"Masukkan panjang sisi kubus: "*
- Kemudian pengguna diminta untuk memasukkan nilai panjang sisi.
- Nilai yang dimasukkan akan disimpan ke dalam variabel sisi.

#### 3. Perhitungan Volume

```
volume := sisi * sisi * sisi
```

Hasil perhitungan disimpan ke dalam variabel volume.

#### 4. Menampilkan Output

```
fmt.Printf("Keluaran: %.0f.5\n", float64(volume))
```

- Program mencetak hasil volume dalam format **float** (float64).
- Format `%.0f` digunakan agar angka ditampilkan tanpa desimal.
- Namun, di bagian kode ini ada tambahan `.5` yang menyebabkan hasil output akan menampilkan angka volume, lalu diikuti dengan `.5`. Misalnya, jika `sisi = 3`, maka `volume = 27` → hasil keluaran: `Keluaran: 27.5`

**Catatan:** Secara logika, `.5` di akhir format output sebenarnya tidak umum digunakan dan justru membuat hasil tampak tidak konsisten. Jika maksudnya ingin menampilkan angka desimal 1 tempat (contoh 27.5), sebaiknya ditulis:

```
fmt.Printf("Keluaran: %.1f\n", float64(volume))
```

### Fungsi Program

Program ini berfungsi untuk:

- Meminta input berupa panjang sisi kubus.
- Menghitung volume kubus menggunakan rumus ( $V = s^3$ ).
- Menampilkan hasil perhitungan ke layar dalam format angka desimal.

### Contoh Eksekusi

Masukkan panjang sisi kubus: 4

Keluaran: 64.5

( hasil akan selalu ada tambahan .5)

Jika diperbaiki (menggunakan `%.1f`), output akan lebih tepat:

Masukkan panjang sisi kubus: 4

Keluaran: 64.0

---

## 2. Guided 2

### Source Code

```
package main

import "fmt"

func main() {

    var alas, tinggi, luas float64

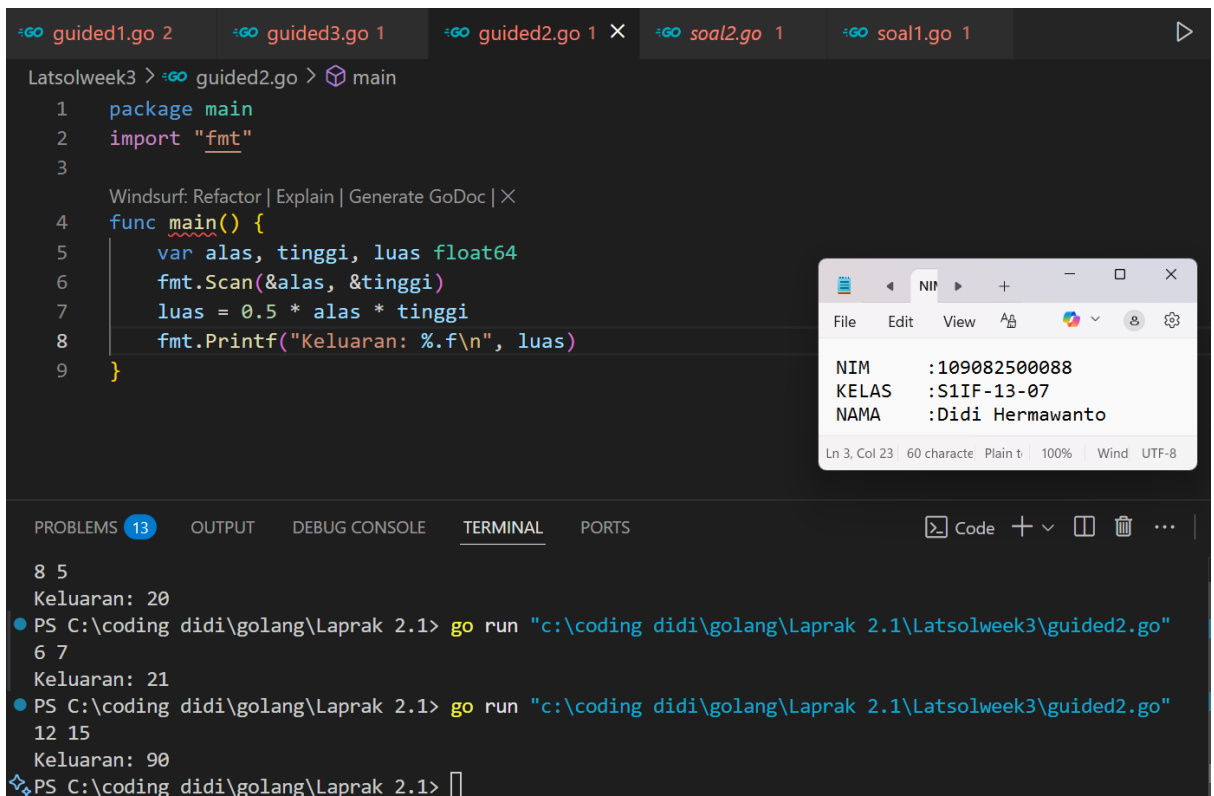
    fmt.Scan(&alas, &tinggi)

    luas = 0.5 * alas * tinggi

    fmt.Printf("Keluaran: %.f\n", luas)

}
```

### Screenshoot program



```
Latsolweek3 > go guided2.go > main
1 package main
2 import "fmt"
3
4 func main() {
5     var alas, tinggi, luas float64
6     fmt.Scan(&alas, &tinggi)
7     luas = 0.5 * alas * tinggi
8     fmt.Printf("Keluaran: %.f\n", luas)
9 }

Windsurf: Refactor | Explain | Generate GoDoc | X

8 5
Keluaran: 20
PS C:\coding didi\golang\Laprak 2.1> go run "c:\coding didi\golang\Laprak 2.1\Latsolweek3\guided2.go"
6 7
Keluaran: 21
PS C:\coding didi\golang\Laprak 2.1> go run "c:\coding didi\golang\Laprak 2.1\Latsolweek3\guided2.go"
12 15
Keluaran: 90
PS C:\coding didi\golang\Laprak 2.1>
```

NIM :109082500088  
KELAS :S1IF-13-07  
NAMA :Didi Hermawanto

Ln 3, Col 23 | 60 character | Plain text | 100% | Window | UTF-8

### Deskripsi program

#### Judul

Program Menghitung Luas Segitiga

## **Tujuan**

Membuat program sederhana dengan bahasa Go untuk menghitung luas segitiga menggunakan input alas dan tinggi dari pengguna, lalu menampilkan hasil perhitungannya ke layar.

## **Dasar Teori**

Segitiga adalah bangun datar yang memiliki tiga sisi. Rumus untuk mencari luas segitiga adalah:  $\frac{1}{2} \times \text{alas} \times \text{tinggi}$

Rumus ini digunakan karena segitiga dianggap sebagai separuh dari persegi panjang yang memiliki ukuran alas dan tinggi yang sama.

## **Alat dan Bahan**

- Bahasa pemrograman Go
- Paket fmt untuk keperluan input/output
- Perangkat komputer untuk menjalankan kode

## **Pembahasan**

### **1. Deklarasi-variabel**

Program mendeklarasikan tiga variabel bertipe float64, yaitu alas, tinggi, dan luas. Tipe data ini dipilih karena mampu menampung bilangan pecahan maupun bilangan bulat.

### **2. Input-data**

Program menggunakan perintah `fmt.Scan(&alas, &tinggi)` untuk membaca dua nilai sekaligus dari pengguna. Nilai pertama dianggap sebagai alas, sedangkan nilai kedua adalah tinggi.

### **3. Proses-perhitungan**

Setelah data masuk, program menghitung luas segitiga dengan rumus  $0.5 \times \text{alas} \times \text{tinggi}$ . Hasilnya disimpan dalam variabel luas.

### **4. Output**

Program menampilkan hasil dengan `fmt.Printf("Keluaran: %.f\n", luas)`. Format `%.f` membuat output hanya menampilkan bilangan bulat tanpa pecahan. Hal ini menyebabkan hasil pecahan akan dibulatkan (misalnya 7.5 ditampilkan sebagai 8).

## **Contoh Uji Coba**

- Input: alas = 4, tinggi = 3  
Output: Keluaran: 6
- Input: alas = 5, tinggi = 3  
Perhitungan:  $0.5 \times 5 \times 3 = 7.5$   
Output yang tampil: Keluaran: 8
- Input: alas = 7, tinggi = 2  
Output: Keluaran: 7

### 3. Guided 3

#### Source Code

```
package main

import "fmt"

const KURS_IDR_TO_USD = 15000.0

func main() {
    var idr int

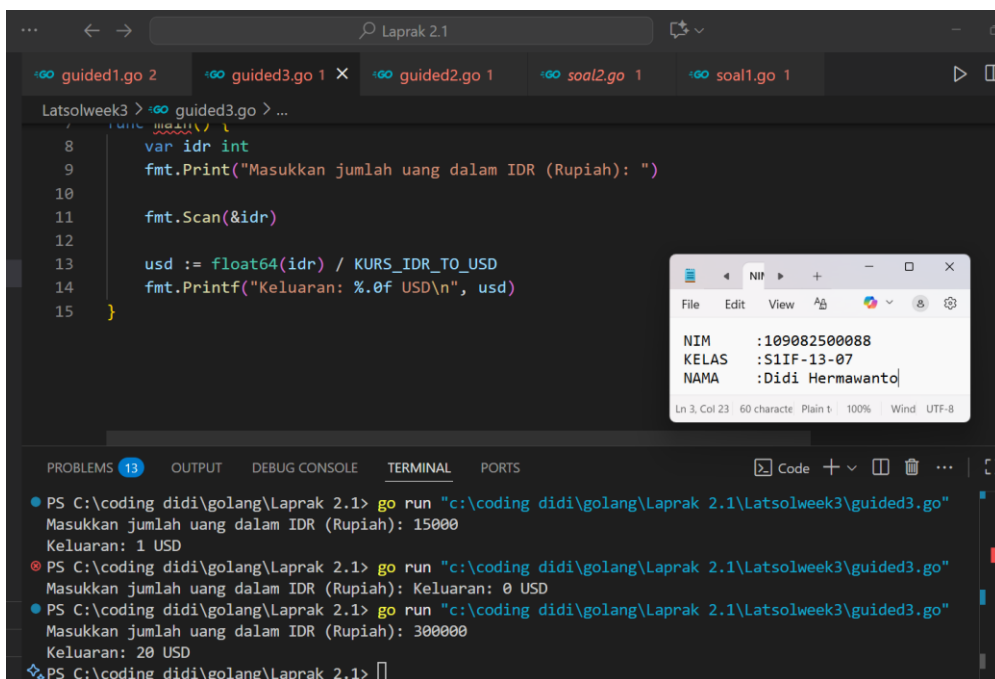
    fmt.Print("Masukkan jumlah uang dalam IDR (Rupiah): ")

    fmt.Scan(&idr)

    usd := float64(idr) / KURS_IDR_TO_USD

    fmt.Printf("Keluaran: %.0f USD\n", usd)
}
```

#### Screenshoot program



The screenshot shows a VS Code editor with a Go file named `guided3.go` open. The code is as follows:

```
7 func main() {
8     var idr int
9     fmt.Print("Masukkan jumlah uang dalam IDR (Rupiah): ")
10
11     fmt.Scan(&idr)
12
13     usd := float64(idr) / KURS_IDR_TO_USD
14     fmt.Printf("Keluaran: %.0f USD\n", usd)
15 }
```

The terminal output shows the program being run with different inputs:

```
PS C:\coding didi\golang\Laprak 2.1> go run "c:\coding didi\golang\Laprak 2.1\Latsolweek3\guided3.go"
Masukkan jumlah uang dalam IDR (Rupiah): 15000
Keluaran: 1 USD

PS C:\coding didi\golang\Laprak 2.1> go run "c:\coding didi\golang\Laprak 2.1\Latsolweek3\guided3.go"
Masukkan jumlah uang dalam IDR (Rupiah): Keluaran: 0 USD

PS C:\coding didi\golang\Laprak 2.1> go run "c:\coding didi\golang\Laprak 2.1\Latsolweek3\guided3.go"
Masukkan jumlah uang dalam IDR (Rupiah): 300000
Keluaran: 20 USD

PS C:\coding didi\golang\Laprak 2.1>
```

A small window titled "NIP" is also visible, showing the following information:

| NIP   |                  |
|-------|------------------|
| NIM   | :109082500088    |
| KELAS | :S1IF-13-07      |
| NAMA  | :Didi Hermawanto |

## Deskripsi program

### Judul

Program Konversi Mata Uang Rupiah (IDR) ke Dolar Amerika (USD)

---

### Tujuan

Membuat program sederhana dalam bahasa Go yang mampu mengonversi nilai uang dari Rupiah (IDR) ke Dollar Amerika (USD) berdasarkan kurs yang ditentukan, lalu menampilkan hasil konversi ke layar.

---

### Alat dan Bahan

1. Bahasa pemrograman Go.
  2. Paket fmt untuk input dan output.
  3. Komputer atau laptop yang sudah terpasang compiler Go.
- 

### Analisis Program / Pembahasan

#### 1. Pendeklarasian Konstanta

```
const KURS_IDR_TO_USD = 15000.0
```

Baris ini mendefinisikan nilai kurs Rupiah ke Dollar, yaitu 15.000.0. Tipe datanya float64 agar bisa digunakan dalam operasi pecahan.

#### 2. Deklarasi Variabel

```
var idr int
```

Variabel idr disiapkan untuk menyimpan jumlah uang dalam Rupiah yang dimasukkan pengguna. Tipe datanya integer karena jumlah uang biasanya berupa bilangan bulat.

#### 3. Input dari Pengguna

```
fmt.Print("Masukkan jumlah uang dalam IDR (Rupiah): ")
```

```
fmt.Scan(&idr)
```

Program menampilkan pesan ke layar dan menunggu input pengguna berupa angka Rupiah. Nilai tersebut dimasukkan ke dalam variabel idr.

#### 4. Proses Konversi

```
usd := float64(idr) / KURS_IDR_TO_USD
```

Nilai idr dikonversi ke float64 agar dapat dibagi dengan konstanta KURS\_IDR\_TO\_USD. Hasilnya berupa nilai uang dalam USD.

#### 5. Output Hasil

```
fmt.Printf("Keluaran: %.0f USD\n", usd)
```

Hasil konversi ditampilkan dalam format bilangan bulat (%.0f), sehingga pecahan desimal diabaikan.

---

### Contoh Uji Coba

#### Kasus 1:

Input: 15000

Perhitungan:  $(15000 / 15000 = 1)$

Output: Keluaran: 1 USD

**Kasus 2:**

Input: 30000

Perhitungan:  $(30000 / 15000 = 2)$

Output: Keluaran: 2 USD

**Kasus 3:**

Input: 20000

Perhitungan:  $(20000 / 15000 \approx 1.33)$

Output: Keluaran: 1 USD (karena format output dibulatkan)

---

**Alur Kerja**

Program ini dibuat untuk mengubah nilai mata uang dari Rupiah ke Dollar Amerika. Kurs yang digunakan sudah ditentukan sejak awal, yaitu 1 USD sama dengan 15.000 Rupiah. Dengan adanya kurs ini, setiap input yang dimasukkan pengguna akan otomatis dihitung berdasarkan nilai tukar tersebut.

Alur kerjanya sederhana. Pertama, pengguna diminta untuk mengetikkan jumlah uang dalam Rupiah. Nilai yang dimasukkan kemudian diproses dengan cara dibagi dengan kurs 15.000. Hasil dari pembagian inilah yang menjadi nilai dalam Dollar Amerika. Setelah itu, hasil perhitungan ditampilkan di layar.

Dalam program ini hasil keluaran ditampilkan dalam bentuk bilangan bulat. Artinya, jika hasil pembagian menghasilkan angka pecahan, maka pecahan tersebut tidak ditampilkan. Misalnya, jika konversi sebenarnya 1,33 USD, maka yang ditampilkan hanya 1 USD. Hal ini membuat tampilan lebih sederhana, tetapi kurang akurat karena angka desimal tidak ditunjukkan.

---



## TUGAS

### 1. Tugas 1

#### Source code

```
package main

import "fmt"

func main(){

    var fx float64

    fmt.Print("Masukkan nilai fx: ")

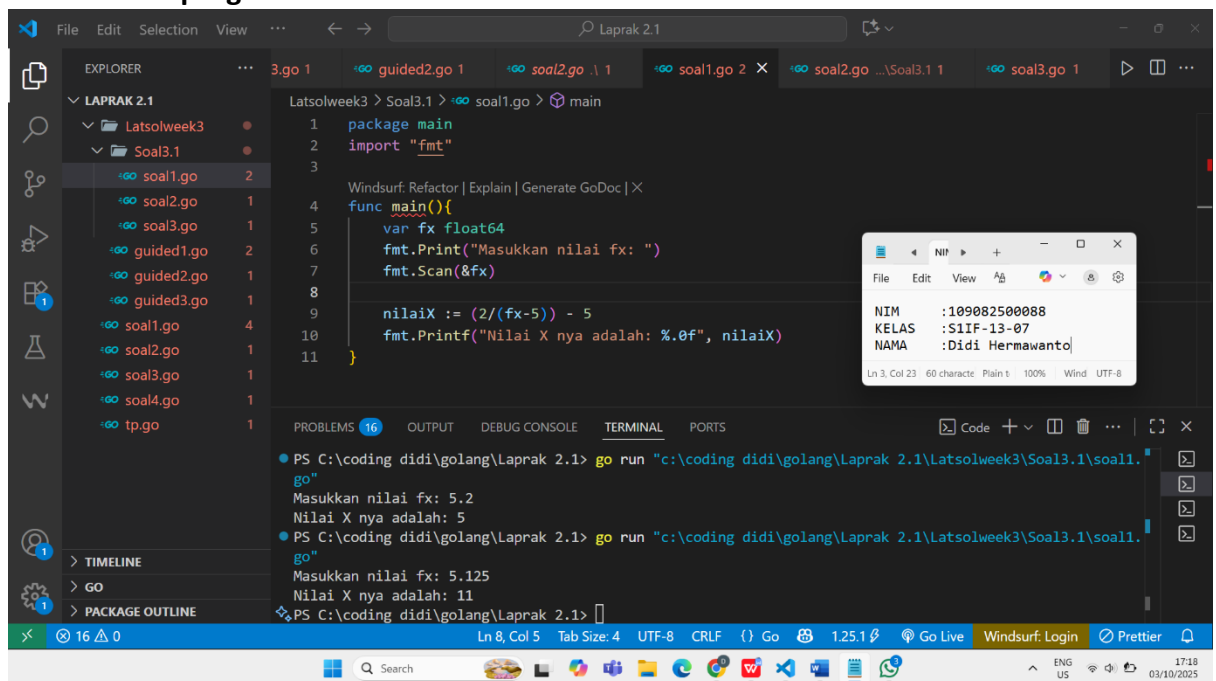
    fmt.Scan(&fx)

    nilaiX := (2/(fx-5)) - 5

    fmt.Printf("Nilai X nya adalah: %.0f", nilaiX)

}
```

#### Screenshoot program



#### Deskripsi program

#### Penjelasan Kode Program

```
package main
import "fmt"
```

Baris ini mendefinisikan bahwa program ditulis dalam bahasa Go.

- package main menunjukkan bahwa program merupakan program utama yang bisa dijalankan.
- import "fmt" berfungsi untuk memanggil paket **fmt** yang digunakan untuk melakukan input dan output (seperti Print, Println, dan Scan).

---

```
func main(){
```

Fungsi main() adalah fungsi utama yang pertama kali dieksekusi ketika program dijalankan.

---

```
var fx float64
```

Mendeklarasikan sebuah variabel fx bertipe float64. Variabel ini dipakai untuk menampung input dari pengguna, yaitu nilai (  $f(x)$  ). Tipe float64 dipilih karena nilai input bisa berupa bilangan riil (ada desimalnya).

---

```
fmt.Print("Masukkan nilai fx: ")
```

```
fmt.Scan(&fx)
```

Dua baris ini berfungsi untuk:

- Menampilkan pesan "Masukkan nilai fx: " ke layar agar pengguna tahu harus memasukkan nilai.
- Membaca input dari pengguna dan menyimpannya ke dalam variabel fx.

---

```
nilaiX := (2/(fx-5)) - 5
```

Bagian ini melakukan perhitungan matematis sesuai rumus hasil manipulasi:

$x = \{2\} / \{f(x) - 5\} - 5$

- $fx-5$  menghitung selisih nilai input dengan 5.
- $2/(fx-5)$  adalah pembagian 2 dengan hasil tersebut.
- Hasilnya kemudian dikurangi 5.
- Nilai akhirnya disimpan dalam variabel nilaiX.

---

```
fmt.Printf("Nilai X nya adalah: %.0f", nilaiX)
```

Bagian ini menampilkan hasil perhitungan ke layar.

- Printf digunakan agar bisa mengatur format output.
- %.0f artinya hasil ditampilkan dalam format bilangan riil (float), tetapi tanpa angka desimal (dibulatkan).
- Misalnya, jika hasilnya 5.0, maka yang tampil hanya 5.

---

### Ringkasan Alur Program

1. Program meminta input berupa nilai (  $f(x)$  ).
2. Input disimpan dalam variabel fx.
3. Program menghitung nilai (  $x$  ) menggunakan rumus (  $x = \{2\} / \{f(x) - 5\} - 5$  ).
4. Hasil perhitungan ditampilkan dalam bentuk bilangan bulat tanpa desimal.

---

### Alur Logika Matematis

Awalnya kita punya fungsi:

$$f(x) = \frac{2}{x+5} + 5$$

Karena yang diminta adalah mencari  $x$  jika diberikan nilai  $(f(x))$ , maka persamaan harus dimanipulasi balik.

1. **Kurangi 5 pada kedua sisi persamaan:**

$$f(x) - 5 = \frac{2}{x+5}$$

Tujuannya biar bentuk pecahan ( $\frac{2}{x+5}$ ) bisa berdiri sendiri.

2. **Balik posisi pecahan (inverse):**

$$x + 5 = \frac{2}{f(x) - 5}$$

Di sini kita pakai sifat aljabar, kalau  $(a = \frac{b}{c})$  maka  $(c = \frac{b}{a})$ .

3. **Kurangi 5 di kedua sisi:**

$$x = \frac{2}{f(x) - 5} - 5$$

Nah, dari sini ketemu rumus  $(x)$  dalam bentuk sederhana. Rumus inilah yang dipakai langsung di dalam program.

---

### Hubungan Logika dengan Program

- Input  $(f(x)) \rightarrow$  disimpan di variabel  $fx$ .
- Operasi  $fx - 5$  sesuai langkah aljabar pertama.
- $2 / (fx - 5)$  sesuai langkah inverse di atas.
- Kemudian  $- 5$  terakhir sesuai langkah ketiga.
- Hasilnya jadi nilai  $x$ , disimpan ke variabel nilai  $X$ .

## 2. Tugas 2

### Source code

```
package main

import "fmt"

func main() {
    const phi = 3.1415926535

    var jejari float64

    fmt.Print("Masukkan nilai jejari: ")
    fmt.Scan(&jejari)

    volumeBola := (4.0 / 3.0) * phi * jejari * jejari *
    jejari
    luasBola := 4 * phi * jejari * jejari
```

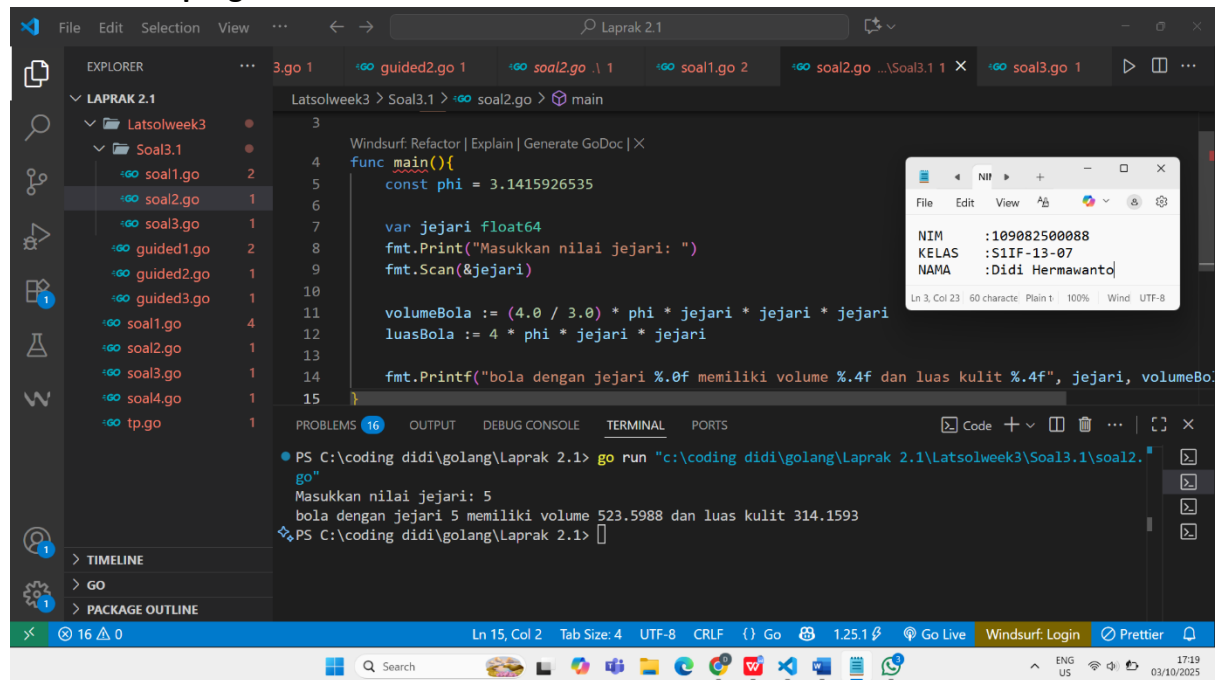
```

        fmt.Printf("bola dengan jejari %.0f memiliki volume %.4f
dan luas kulit %.4f", jejari, volumeBola, luasBola)

    }

```

## Screenshoot program



## Deskripsi Program

Program ini dibuat untuk menghitung **volume bola** dan **luas permukaan bola** apabila diketahui nilai jejari (radius) dari bola. Konsep yang digunakan dalam program berasal dari rumus matematika dasar pada geometri.

### 1. Tujuan

- Menggunakan **konstanta** di dalam program (dalam hal ini konstanta ( $\pi$ ) atau phi).
- Mengimplementasikan **rumus volume dan luas permukaan bola** ke dalam bahasa pemrograman Go.
- Melakukan perhitungan matematis menggunakan tipe data pecahan (float64).
- Menampilkan hasil perhitungan dengan format tertentu.

### 2. Rumus Dasar

Rumus volume bola:

$$V = \frac{4}{3} \pi r^3$$

Rumus luas permukaan bola:

$$L = 4 \pi r^2$$

Dimana (r) adalah jejari bola.

### 3. Alur Logika Program

1. Program mendeklarasikan sebuah konstanta phi dengan nilai (3.1415926535). Konstanta ini digunakan sebagai pengganti ( $\pi$ ).
2. Program meminta input dari pengguna berupa nilai jari bola. Input ini disimpan pada variabel jari.
3. Program menghitung volume bola dengan rumus  $((4.0/3.0) \times \pi \times r^3)$ .
4. Program menghitung luas permukaan bola dengan rumus  $(4 \times \pi \times r^2)$ .
5. Program menampilkan hasil perhitungan ke layar dengan format: jari, volume bola, dan luas permukaan bola.

---

#### 4. Penjelasan Kode Program

- `const phi = 3.1415926535` → mendefinisikan nilai konstanta phi ( $\pi$ ) agar lebih akurat.
- `var jari float64` → mendeklarasikan variabel jari bertipe float64 karena nilai radius bisa berupa desimal.
- `fmt.Scan(&jari)` → membaca input dari user dan menyimpannya ke variabel jari.
- `volumeBola := (4.0/3.0) * phi * jari * jari * jari` → menghitung volume sesuai rumus bola.
- `luasBola := 4 * phi * jari * jari` → menghitung luas permukaan bola.
- `fmt.Printf(...)` → mencetak hasil dengan format yang lebih rapi, menampilkan jari, volume, dan luas permukaan dengan 4 angka di belakang koma.

---

#### 5. Contoh Eksekusi Program

Masukkan nilai jari: 7

bola dengan jari 7 memiliki volume 1436.7550 dan luas kulit 615.7520

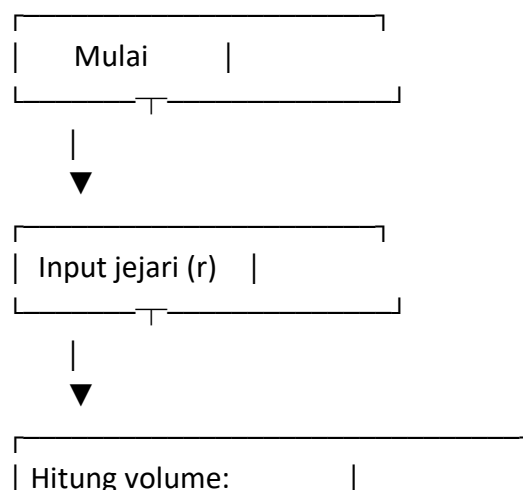
---

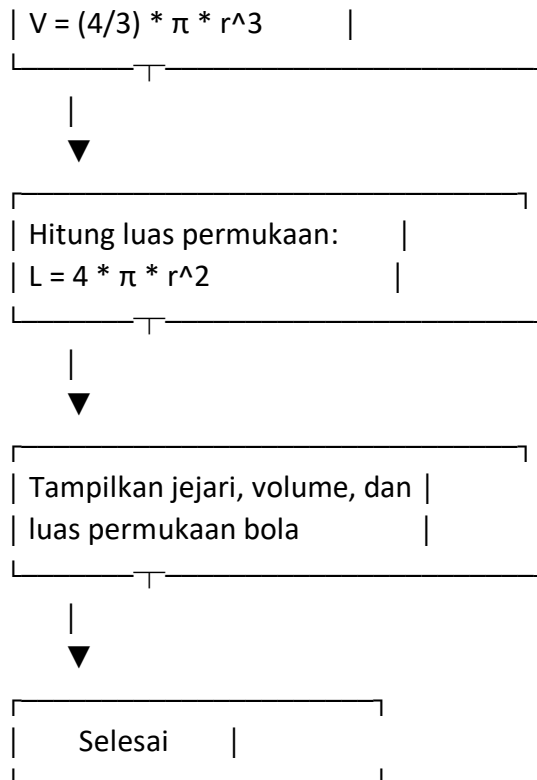
Oke, aku bikin **flowchart sederhana** untuk program perhitungan volume dan luas bola ini.

Biar enak dipahami, aku tulis dalam bentuk teks diagram (kalau di laporan bisa kamu gambar ulang pakai aplikasi kayak *draw.io*, *Lucidchart*, atau bahkan manual di kertas).

---

#### Flowchart Program Volume & Luas Bola





---

### Penjelasan Flowchart

1. Program dimulai → user diminta memasukkan nilai jejari.
2. Setelah jejari diperoleh, program langsung melakukan dua perhitungan: volume bola dan luas permukaan bola.
3. Kedua hasil tersebut kemudian ditampilkan ke layar bersamaan dengan nilai jejari.
4. Program selesai dijalankan.

### 3. Tugas 3

#### Source code

```
package main

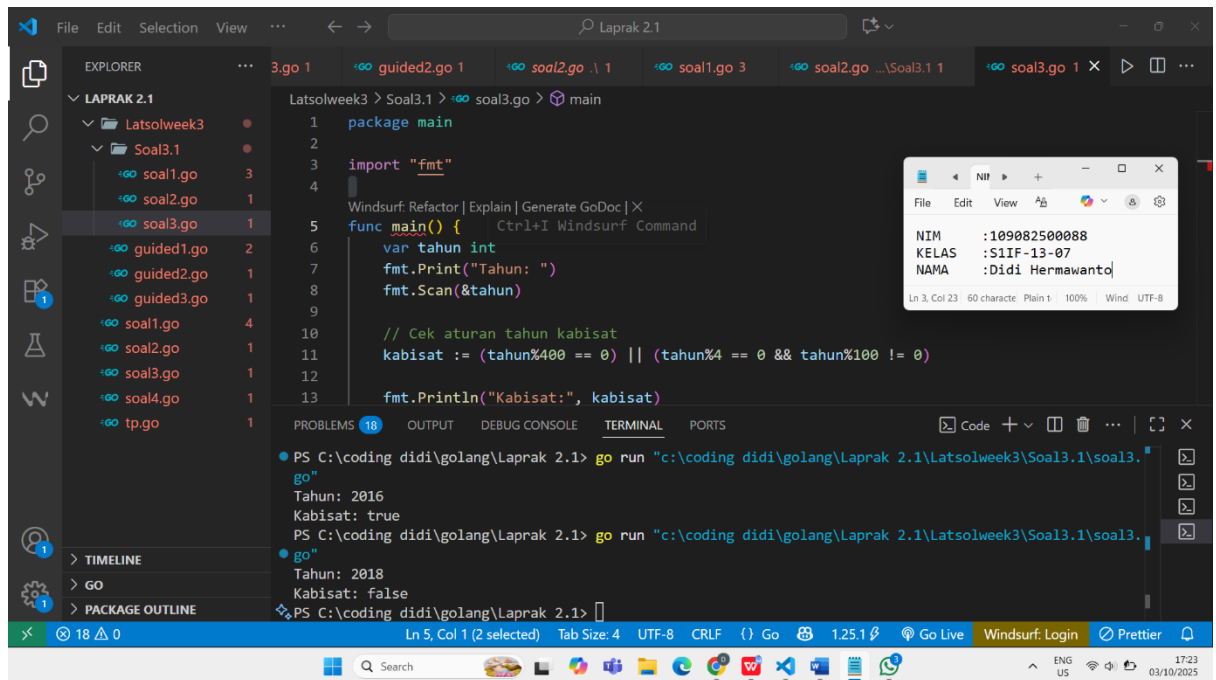
import "fmt"

func main() {
    var tahun int
    fmt.Print("Tahun: ")
    fmt.Scan(&tahun)

    kabisat := (tahun%400 == 0) || (tahun%4 == 0 && tahun%100
!= 0)

    fmt.Println("Kabisat:", kabisat)
}
```

#### Screenshoot program



## Deskripsi program

### Tujuan

Menulis program sederhana dalam bahasa Go untuk mengecek apakah sebuah tahun termasuk tahun kabisat atau bukan, lalu menampilkan hasilnya (true/false). Program ini melatih pemahaman kondisi logika, operasi modulus, dan input/output dasar.

### Dasar teori singkat

Aturan tahun kabisat yang dipakai pada kalender Gregorian:

- Tahun kabisat jika habis dibagi 400.
- Atau, tahun kabisat jika habis dibagi 4 tetapi **tidak** habis dibagi 100.  
Contoh: 2000 → kabisat (habis dibagi 400), 2016 → kabisat (habis dibagi 4 dan bukan kelipatan 100), 1900 → bukan kabisat (habis dibagi 100 tapi bukan 400).

### Alur kerja program (ringkas)

1. Program meminta input satu bilangan bulat (tahun).
2. Program mengecek kondisi kabisat berdasarkan aturan di atas.
3. Menampilkan hasil berupa Kabisat: true atau Kabisat: false.

### Penjelasan kode (baris per baris, pakai bahasa sehari-hari)

```
package main
```

```
import "fmt"
```

- Menyatakan paket utama program dan memanggil paket fmt untuk operasi input/output.

```
func main() {
```

```
    var tahun int
```

- main() adalah titik awal program. Kita membuat variabel tahun bertipe int untuk menampung masukan pengguna.

```
    fmt.Print("Tahun: ")
```

```
    fmt.Scan(&tahun)
```

- Cetak teks Tahun: agar pengguna tahu harus memasukkan angka.
- fmt.Scan(&tahun) membaca nilai dari keyboard dan menyimpannya ke variabel tahun. (Perlu dicatat: jika pengguna mengetik bukan angka, program akan gagal membaca — tanpa penanganan error.)

```
    // Cek aturan tahun kabisat
```

```
    kabisat := (tahun%400 == 0) || (tahun%4 == 0 && tahun%100 != 0)
```

- Ini inti logikanya. tahun%400 == 0 mengecek kelipatan 400.
- tahun%4 == 0 && tahun%100 != 0 mengecek kelipatan 4 tetapi bukan kelipatan 100.
- Operator || berarti salah satu kondisi benar → kabisat bernilai true. Hasilnya disimpan dalam variabel boolean kabisat.

```
    fmt.Println("Kabisat:", kabisat)
```

```
}
```

- Menampilkan hasil ke layar dalam format Kabisat: true atau Kabisat: false.

---

### Contoh input/output (uji sederhana)

- Input: 2016 → Output: Kabisat: true
  - Input: 2000 → Output: Kabisat: true
  - Input: 2018 → Output: Kabisat: false
  - Input: 1900 → Output: Kabisat: false (karena 1900 habis dibagi 100 tapi tidak 400)
-



### Catatan, keterbatasan, dan saran perbaikan (penting untuk laprak)

- **Validasi input:** Saat ini tidak ada pengecekan apakah `fmt.Scan` berhasil. Sebaiknya tangani kasus input non-angka dan tampilkan pesan error.
  - **Nilai negatif / tahun sebelum Masehi:** Program menerima angka negatif atau nol, tapi secara historis kalender berbeda; kalau perlu batasi domain input (mis. tahun  $\geq 1$ ).
  - **Pesan output:** Untuk laporan yang lebih ramah, tampilkan kalimat lengkap, mis. `fmt.Printf("%d adalah tahun kabisat\n", tahun)` atau ... bukan tahun kabisat.
  - **Unit test:** Tambahkan sekumpulan tes otomatis untuk memastikan aturan benar (mis. 1600, 1700, 2000, 2012, 2019).
  - **Penggunaan fungsi terpisah:** Untuk pengujian dan keterbacaan, buat fungsi `isLeapYear(t int) bool` dan panggil dari `main`.
  - **Perbedaan kalender:** Aturan yang dipakai adalah aturan Gregorian. Jika program ditujukan untuk tahun sebelum 1582, beri catatan atau gunakan kalender proleptik.
- 

### Kompleksitas

Operasi yang dilakukan konstanta waktu —  $O(1)$ . Memakai memori tetap  $O(1)$ .

---

## 4. Tugas 4

### Source code

```
package main

import "fmt"

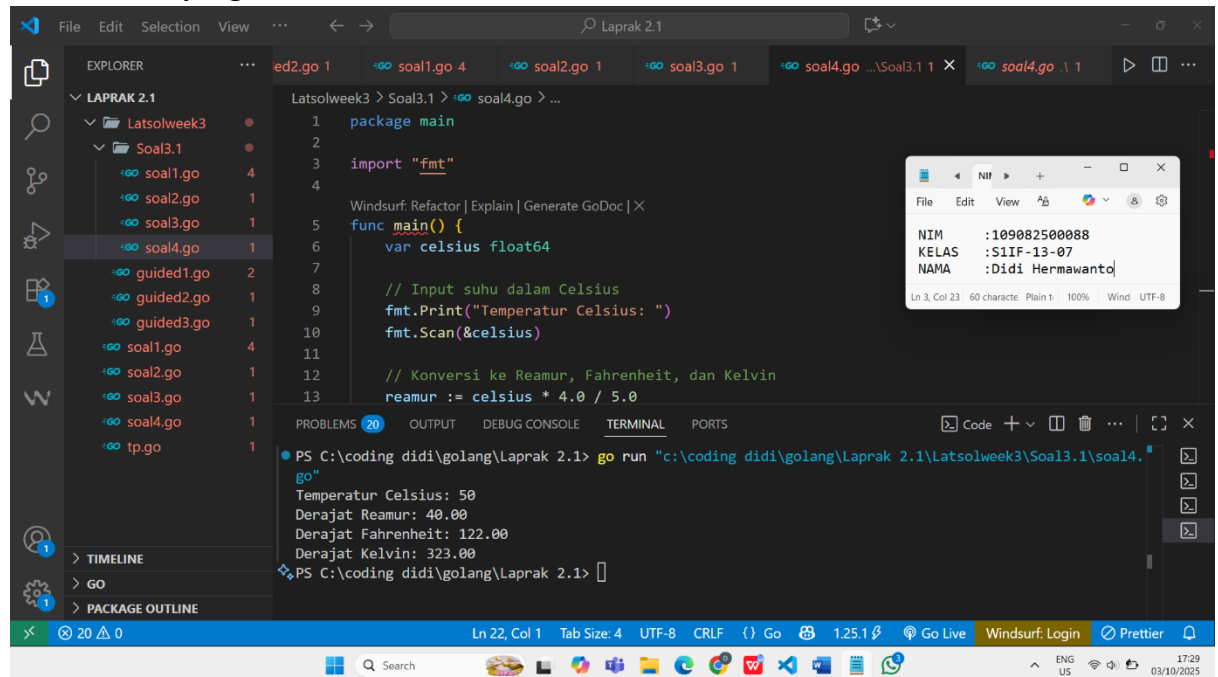
func main() {
    var celsius float64

    fmt.Print("Temperatur Celsius: ")
    fmt.Scan(&celsius)

    reamur := celsius * 4.0 / 5.0
    fahrenheit := (celsius * 9.0 / 5.0) + 32
    kelvin := celsius + 273

    fmt.Printf("Derajat Reamur: %.2f\n", reamur)
    fmt.Printf("Derajat Fahrenheit: %.2f\n", fahrenheit)
    fmt.Printf("Derajat Kelvin: %.2f\n", kelvin)
}
```

## Screenshoot program



## Deskripsi program

### Judul

Program Konversi Suhu (Celsius → Reamur, Fahrenheit, Kelvin)

### Tujuan

Membuat sebuah program sederhana menggunakan bahasa Go untuk melakukan konversi suhu yang dimasukkan dalam satuan Celsius, lalu menampilkan hasil perhitungannya ke dalam tiga satuan lain, yaitu Reamur, Fahrenheit, dan Kelvin. Program ini bertujuan untuk melatih pemahaman mahasiswa dalam menggunakan input/output, variabel, tipe data pecahan (float64), serta penerapan rumus matematis ke dalam kode.

### Dasar Teori

1. **Suhu** adalah besaran fisika yang menyatakan tingkat panas atau dinginnya suatu benda. Untuk menyatakan suhu digunakan berbagai skala, di antaranya:
  - **Celsius (°C)**: skala yang paling umum dipakai, titik beku air 0 °C, titik didih 100 °C.
  - **Reamur (°R)**: digunakan di Eropa zaman dulu, titik beku air 0 °R, titik didih 80 °R.

- **Fahrenheit (°F):** umum digunakan di Amerika Serikat, titik beku air 32 °F, titik didih 212 °F.
- **Kelvin (K):** skala absolut, titik nol mutlak pada 0 K (−273 °C).

## 2. Rumus konversi suhu:

- Reamur = Celsius  $\times$  4/5
- Fahrenheit = (Celsius  $\times$  9/5) + 32
- Kelvin = Celsius + 273

Dengan rumus ini, suhu dalam °C dapat diubah ke satuan lain secara langsung.

## Alur Logika Program

1. Program dimulai, lalu mendeklarasikan variabel untuk menyimpan input suhu dalam **Celsius**.
2. Program meminta pengguna memasukkan suhu dalam Celsius melalui `fmt.Scan`.
3. Setelah nilai terbaca, program melakukan perhitungan:
  - `reamur = celsius  $\times$  4/5`
  - `fahrenheit = (celsius  $\times$  9/5) + 32`
  - `kelvin = celsius + 273`
4. Program menampilkan hasil perhitungan dalam tiga satuan berbeda dengan format dua angka di belakang koma.
5. Program selesai dijalankan.

## Penjelasan Kode Program

```
package main
```

```
import "fmt"
```

- `package main` menandakan program utama Go.
  - `import "fmt"` digunakan agar bisa memakai fungsi input/output (`Print`, `Scan`, `Printf`).
- ```
func main() {
```
- ```
    var celsius float64
```
- `main()` adalah fungsi utama program.
  - Variabel `celsius` bertipe `float64` dipakai untuk menampung input suhu dari pengguna.

```
fmt.Print("Temperatur Celsius: ")
```

```
fmt.Scan(&celsius)
```

- Program mencetak teks "Temperatur Celsius: " agar pengguna tahu harus mengetik nilai.
- `fmt.Scan(&celsius)` membaca input angka dan menyimpannya ke variabel `celsius`.

```
reamur := celsius * 4.0 / 5.0
```

```
fahrenheit := (celsius * 9.0 / 5.0) + 32
```

```
kelvin := celsius + 273
```

- Bagian ini berisi rumus konversi. Nilai Celsius yang dimasukkan akan diproses untuk menghasilkan tiga nilai suhu dalam skala lain.

```
fmt.Printf("Derajat Reamur: %.2f\n", reamur)
```

```
fmt.Printf("Derajat Fahrenheit: %.2f\n", fahrenheit)
```

```
fmt.Printf("Derajat Kelvin: %.2f\n", kelvin)
```

```
}
```

- Hasil ditampilkan dengan format **dua angka di belakang koma** menggunakan `%.2f`.
- `\n` berfungsi untuk memberi baris baru.

---

### Contoh Input/Output

#### Kasus 1

Temperatur Celsius: 50

Derajat Reamur: 40.00

Derajat Fahrenheit: 122.00

Derajat Kelvin: 323.00

#### Kasus 2

Temperatur Celsius: 0

Derajat Reamur: 0.00

Derajat Fahrenheit: 32.00

Derajat Kelvin: 273.00

---

### Analisis Program

- Program menggunakan operasi matematika sederhana dengan kompleksitas  **$O(1)$** , artinya perhitungan dilakukan hanya sekali tanpa perulangan.
  - Tipe data float64 dipilih agar hasil perhitungan bisa menampilkan bilangan desimal dengan presisi yang cukup.
  - Output dibuat dengan format .2f agar rapi dan mudah dibaca.
- 

### Kelebihan dan Kekurangan

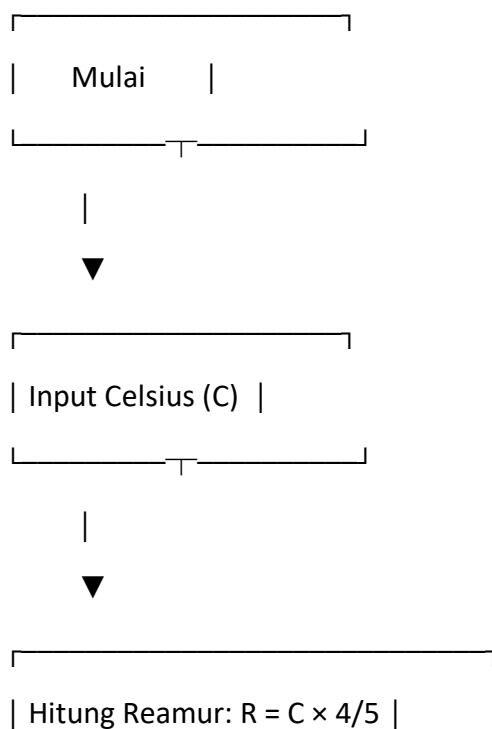
#### Kelebihan:

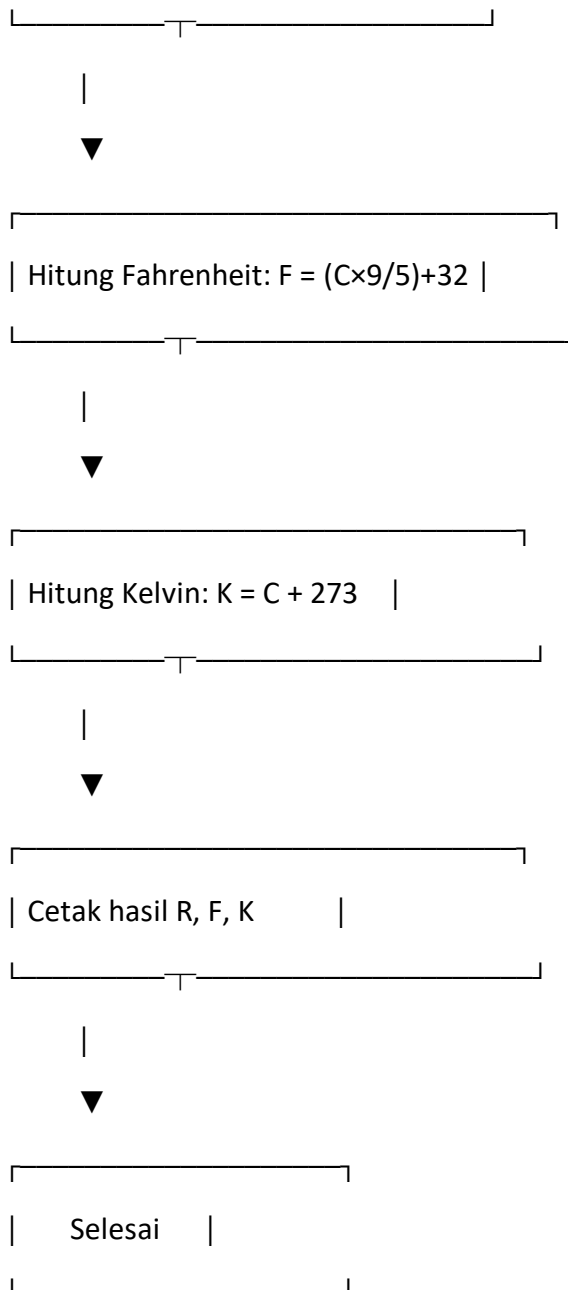
- Sederhana, mudah dipahami.
- Hasil presisi karena menggunakan tipe data float64.
- Output jelas dengan format angka dua desimal.

#### Kekurangan:

- Tidak ada validasi input. Jika pengguna memasukkan huruf/karakter lain, program akan error.
  - Konstanta konversi Kelvin biasanya lebih tepat 273.15, tetapi di sini digunakan 273 agar lebih sederhana.
- 

### Flowchart Konversi Suhu (Celsius → Reamur, Fahrenheit, Kelvin)





---

#### Penjelasan Alur:

1. Program dimulai.
2. Pengguna memasukkan nilai suhu dalam **Celsius**.
3. Program menghitung nilai suhu dalam **Reamur**, **Fahrenheit**, dan **Kelvin** berdasarkan rumus.
4. Semua hasil ditampilkan ke layar.
5. Program selesai.