LAPORAN PRAKTIKUM ALGORITMA

DAN PEMROGRAMAN 1

MODUL [4. 1/0]

TIPE DATA & VARIABEL (LATIHAN 2)

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

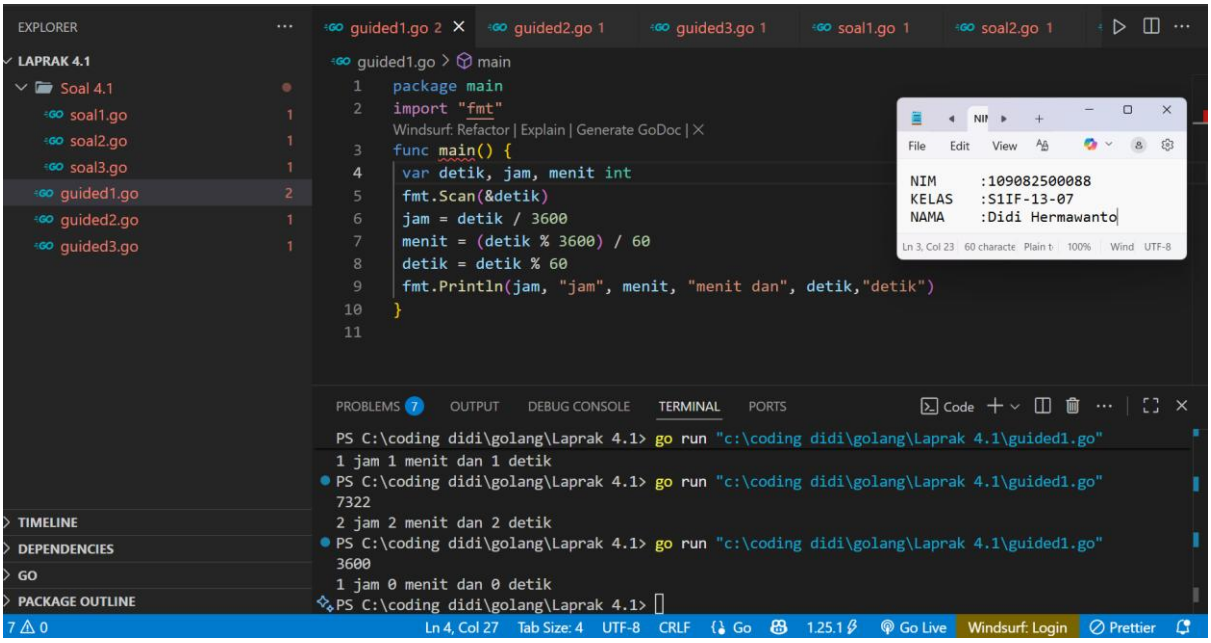
LATIHAN KELAS – GUIDED

1. **Guided 1**

Source Code

```
package main

import "fmt"

func main() {

 var detik, jam, menit int

 fmt.Scan(&detik)

 jam = detik / 3600

 menit = (detik % 3600) / 60

 detik = detik % 60

 fmt.Println(jam, "jam", menit, "menit dan", detik,"detik")

}
```

Screenshoot program



Deskripsi Program Konversi Detik ke Jam, Menit, dan Detik

Program ini dibuat untuk mengubah satuan waktu dari **detik** menjadi bentuk waktu yang lebih mudah dipahami, yaitu **jam, menit, dan detik**.

Dalam kehidupan sehari-hari, waktu sering dinyatakan dalam format jam-menit-detik, bukan hanya detik saja. Misalnya, 3.800 detik lebih mudah dipahami sebagai 1 jam, 3 menit, dan 20 detik.

Penjelasan Setiap Bagian Kode

1. **Baris pertama:**
   package main
   Baris ini menunjukkan bahwa program berada dalam *package* utama.
   Dalam bahasa Go, setiap program yang bisa dijalankan harus berada pada package bernama main.
2. **Baris kedua:**
   import "fmt"
   Baris ini berfungsi untuk mengimpor paket fmt, yang menyediakan fungsi-fungsi untuk melakukan input dan output.
   Dalam program ini, paket fmt digunakan untuk membaca input dari pengguna (fmt.Scan) dan menampilkan hasil ke layar (fmt.Println).
3. **Bagian utama program:**
   func main() {
   Fungsi main() adalah fungsi utama dalam bahasa Go. Semua program Go selalu dieksekusi mulai dari fungsi ini.
4. **Deklarasi variabel:**
   var detik, jam, menit int
   Di sini terdapat tiga variabel bertipe int (bilangan bulat), yaitu:
   - detik → menyimpan nilai input waktu dalam satuan detik,
   - jam → menyimpan hasil konversi jam,
   - menit → menyimpan hasil konversi menit.
5. **Membaca input dari pengguna:**
   fmt.Scan(&detik)
   Baris ini meminta pengguna untuk memasukkan jumlah detik. Nilai yang diketik akan disimpan ke dalam variabel detik.
6. **Proses konversi:**
   jam = detik / 3600
   Karena 1 jam sama dengan **3600 detik**, maka pembagian detik / 3600 akan menghasilkan jumlah jam.
   menit = (detik % 3600) / 60
   Setelah jam dihitung, sisa detiknya diperoleh dengan detik % 3600. Nilai sisa ini kemudian dibagi 60 untuk mendapatkan jumlah menit (karena 1 menit = 60 detik).
   detik = detik % 60
   Baris ini menghitung sisa detik yang tidak mencapai satu menit.
   Jadi prosesnya berurutan:
   - Bagi total detik menjadi jam.
   - Dari sisa waktu, hitung menit.
   - Dari sisa terakhir, ambil detik.
7. **Menampilkan hasil:**
   fmt.Println(jam, "jam", menit, "menit dan", detik, "detik")
   Baris ini mencetak hasil konversi ke layar dalam format yang mudah dibaca.
   Contohnya, jika pengguna memasukkan 3800, maka output yang muncul adalah:

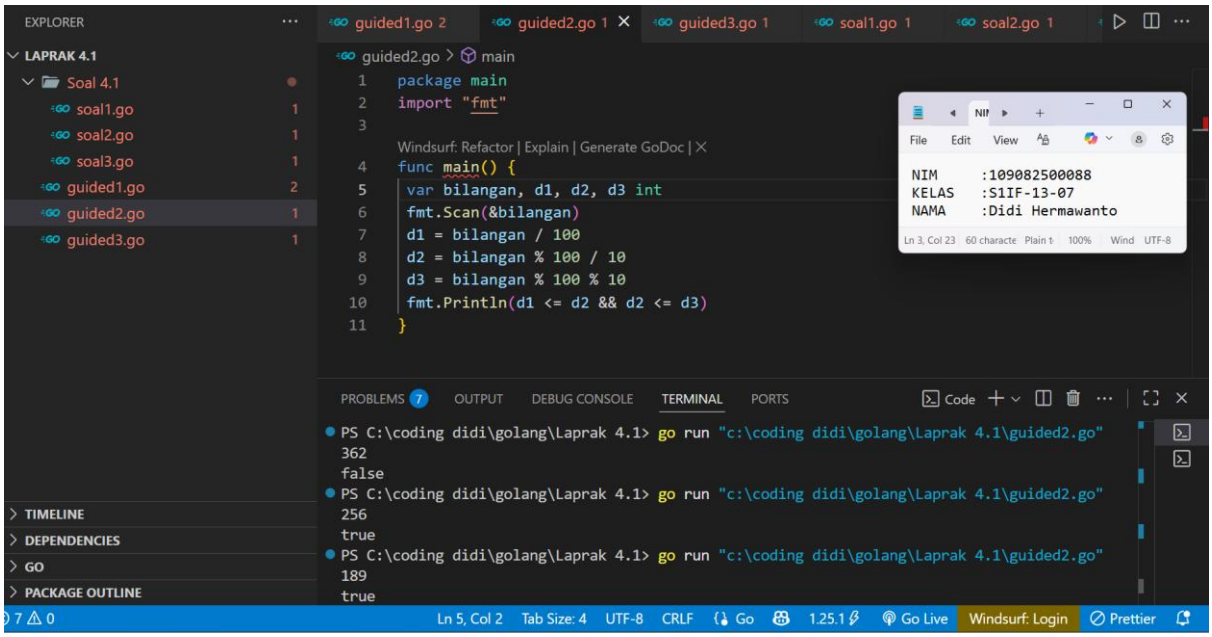
1 jam 3 menit dan 20 detik

2. Guided 2

Source Code

```
package main
import "fmt"

func main() {
 var bilangan, d1, d2, d3 int
 fmt.Scan(&bilangan)
 d1 = bilangan / 100
 d2 = bilangan % 100 / 10
 d3 = bilangan % 100 % 10
 fmt.Println(d1 <= d2 && d2 <= d3)
}
```

Screenshoot program



Deskripsi program

Penjelasan Program:

1. **Bagian Awal Program**

package main
import "fmt"

Ini bagian standar di setiap program Go.
package main artinya program ini bisa langsung dijalankan, dan import "fmt" dipakai supaya kita bisa pakai fungsi input-output seperti fmt.Scan dan fmt.Println.

2. **Deklarasi Fungsi Utama**
   func main() {
   Semua perintah program ditulis di dalam fungsi main().
   Ketika program dijalankan, bagian ini yang pertama kali dieksekusi.
3. **Deklarasi Variabel**
   var bilangan, d1, d2, d3 int
   Di sini ada empat variabel bertipe int (bilangan bulat):
   - bilangan → untuk menyimpan angka tiga digit yang dimasukkan pengguna.
   - d1 → untuk menyimpan digit ratusan.
   - d2 → untuk menyimpan digit puluhan.
   - d3 → untuk menyimpan digit satuan.
4. **Input dari Pengguna**
   fmt.Scan(&bilangan)
   Program ini meminta pengguna memasukkan sebuah angka tiga digit, misalnya **345**.
   Nilai yang dimasukkan akan disimpan di variabel bilangan.
5. **Mengambil Setiap Digit dari Bilangan**
   - **Digit Pertama (Ratusan):**
   - d1 = bilangan / 100

   Misalnya bilangan = 345
   Maka d1 = 345 / 100 = 3.
   - **Digit Kedua (Puluhan):**
   - d2 = bilangan % 100 / 10

   Di sini bilangan % 100 akan mengambil dua angka terakhir → 45,
   lalu dibagi 10 → 45 / 10 = 4, jadi d2 = 4.
   - **Digit Ketiga (Satuan):**
   - d3 = bilangan % 100 % 10

   bilangan % 100 hasilnya 45, lalu % 10 artinya ambil sisa bagi 10 → 5.
   Jadi d3 = 5.
6. **Mengecek Urutan Angka**
   fmt.Println(d1 <= d2 && d2 <= d3)
   Bagian ini melakukan pengecekan apakah digit pertama lebih kecil atau sama dengan digit kedua, dan digit kedua lebih kecil atau sama dengan digit ketiga.
   Kalau kondisi itu benar, program akan menampilkan **true**, kalau tidak, akan muncul **false**.
   Misalnya:
   - Input 345 → 3 ≤ 4 dan 4 ≤ 5 → hasilnya **true**.
   - Input 542 → 5 ≤ 4 salah → hasilnya **false**.
   - Input 333 → 3 ≤ 3 dan 3 ≤ 3 benar → hasilnya **true**.

Analisis Program:

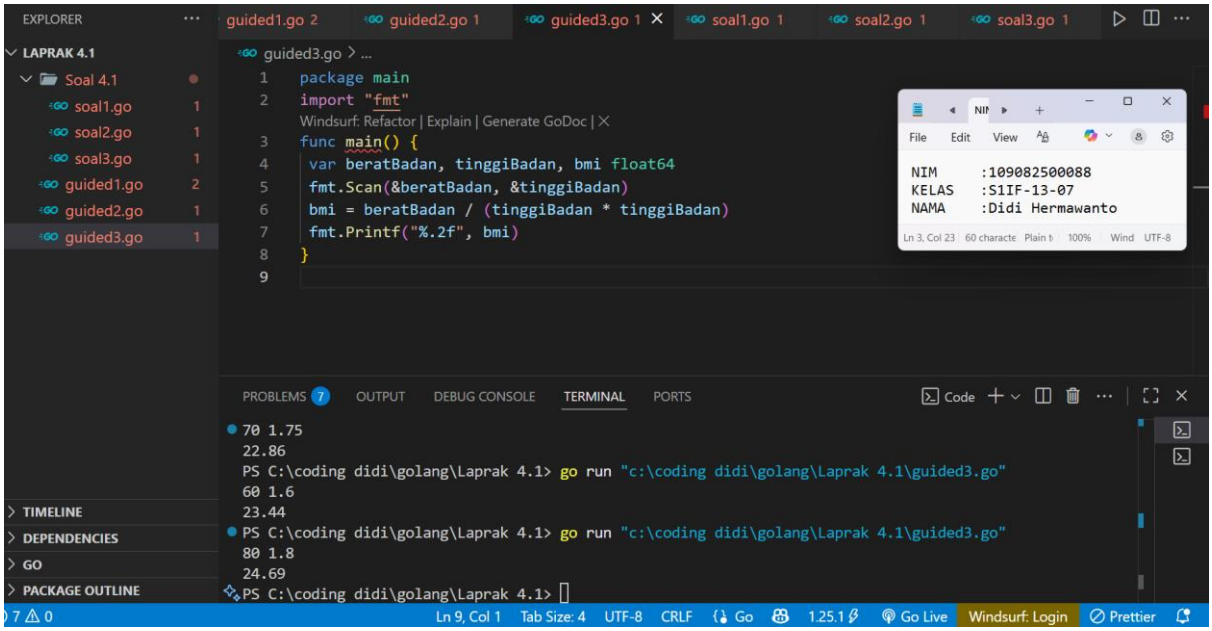
Program ini bekerja dengan cara memecah bilangan tiga digit menjadi ratusan, puluhan, dan satuan menggunakan operasi pembagian dan modulus.
Setelah itu, program membandingkan ketiga digit tersebut dengan operator logika <= dan &&.
Hasilnya berupa nilai **boolean** yaitu true atau false, tergantung apakah urutan digitnya sudah naik atau tidak.
terutama cara mengambil bagian tertentu dari sebuah angka dan bagaimana cara membandingkannya.
Selain itu, dengan memahami program ini, kita bisa bikin program lain yang mirip, seperti cek angka palindrome, cek angka unik, atau bahkan sistem validasi angka tertentu.

3. Guided 3

Source Code

```
package main

import "fmt"

func main() {

 var beratBadan, tinggiBadan, bmi float64

 fmt.Scan(&beratBadan, &tinggiBadan)

 bmi = beratBadan / (tinggiBadan * tinggiBadan)

 fmt.Printf("%.2f", bmi)

}
```

Screenshoot program



Deskripsi program

Penjelasan Program:

1. **Baris package main dan import "fmt"**
   Ini bagian awal program. package main artinya program bisa dijalankan langsung, dan import "fmt" digunakan supaya bisa memakai fungsi input/output seperti fmt.Scan dan fmt.Printf.
2. **Deklarasi Variabel**
   var beratBadan, tinggiBadan, bmi float64
   Di sini ada tiga variabel bertipe float64 (karena hasil BMI bisa berupa angka desimal):
   - beratBadan → untuk menyimpan berat badan (dalam kg).
   - tinggiBadan → untuk menyimpan tinggi badan (dalam meter).
   - bmi → untuk menyimpan hasil perhitungan BMI.
3. **Input dari Pengguna**
   fmt.Scan(&beratBadan, &tinggiBadan)
   Program meminta pengguna untuk memasukkan **dua nilai**: berat dan tinggi badan.
   Contohnya:
   70 1.75
4. **Perhitungan BMI**
   bmi = beratBadan / (tinggiBadan * tinggiBadan)
   Di sini dilakukan perhitungan dengan rumus BMI.
   Tinggi badan dikalikan dengan dirinya sendiri (pangkat dua), lalu berat badan dibagi hasilnya.
5. **Menampilkan Hasil**
   fmt.Printf("%.2f", bmi)
   Program akan menampilkan hasil dengan dua angka di belakang koma.
   Misalnya hasilnya 22.857142857, maka akan ditampilkan 22.86.

Algoritma Program Menghitung BMI

1. **Mulai program.**
2. **Deklarasikan variabel:**
   - beratBadan (float64)
   - tinggiBadan (float64)
   - bmi (float64)
3. **Minta input dari pengguna:**
   Masukkan nilai berat badan (kg) dan tinggi badan (meter).
4. **Hitung nilai BMI** dengan rumus:
5. bmi = beratBadan / (tinggiBadan * tinggiBadan)
6. **Tampilkan hasil BMI** dengan dua angka di belakang koma.
7. **Selesai.**

Contoh Jalannya Algoritma:

Input:

berat = 60
tinggi = 1.6
Langkah-langkah:

- Hitung tinggi² = 1.6 × 1.6 = 2.56
- Hitung BMI = 60 / 2.56 = 23.4375
- Tampilkan hasil → **23.44**

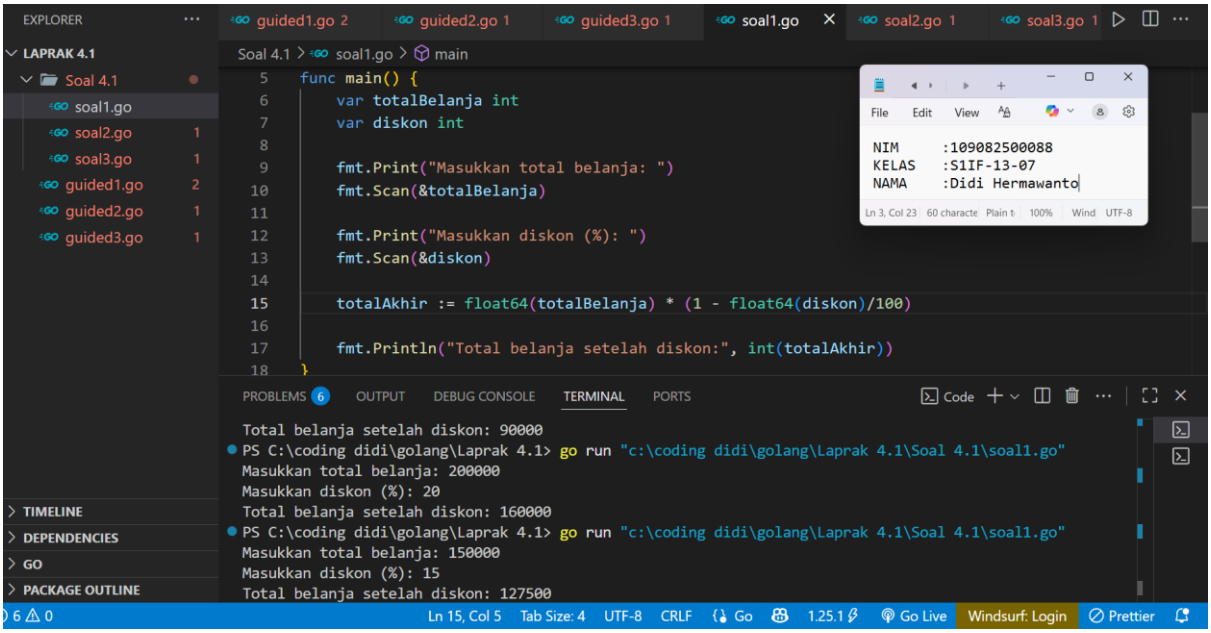
TUGAS

1. **Tugas 1**

Source code

```
package main


import "fmt"


func main() {
    var totalBelanja int
    var diskon int
    fmt.Print("Masukkan total belanja: ")
    fmt.Scan(&totalBelanja)


    fmt.Print("Masukkan diskon (%): ")
    fmt.Scan(&diskon)


    totalAkhir := float64(totalBelanja) * (1 - float64(diskon)/100)
    fmt.Println("Total belanja setelah diskon:", int(totalAkhir))
}
```

Screenshoot program



Deskripsi program

Program ini dibuat untuk menghitung total harga belanja setelah mendapatkan potongan harga atau diskon dengan persentase tertentu. Saat program dijalankan, pengguna akan diminta memasukkan dua input, yaitu total belanja sebelum diskon dan besar diskon dalam persen. Setelah itu, program akan menghitung harga akhirnya dengan rumus totalAkhir = totalBelanja × (1 - diskon/100). Rumus ini digunakan untuk mengurangi total harga sesuai dengan persentase diskon yang diberikan. Hasil perhitungannya kemudian ditampilkan di layar dalam bentuk bilangan bulat, agar terlihat seperti nilai uang dalam rupiah tanpa angka desimal.

Di dalam program, variabel totalBelanja dan diskon digunakan untuk menyimpan input dari pengguna. Karena perhitungan melibatkan pembagian dan pengurangan dalam bentuk persen, kedua nilai tersebut diubah terlebih dahulu menjadi tipe data float64 supaya hasilnya lebih akurat dan tidak dibulatkan seperti pada tipe int. Setelah proses perhitungan selesai, hasilnya dikonversi kembali ke bentuk int agar tampilannya lebih sederhana dan mudah dibaca.

Sebagai contoh, jika pengguna memasukkan total belanja sebesar 200000 dan diskon 20%, maka perhitungannya menjadi 200000 × (1 - 0.2) yang menghasilkan nilai akhir 160000. Jadi, pengguna akan tahu bahwa total yang harus dibayar setelah potongan diskon adalah 160000.

Alur algoritma di sebuah program tersebut :

1. Mulai program.
2. Deklarasikan variabel totalBelanja, diskon, dan totalAkhir.
3. Minta pengguna untuk memasukkan total belanja.
4. Minta pengguna untuk memasukkan besar diskon (dalam persen).
5. Hitung total belanja akhir dengan rumus: totalAkhir = totalBelanja × (1 - diskon / 100)

6. Tampilkan hasil total belanja setelah diskon.
7. Selesai.

Contoh Jalannya Algoritma:
Input:
totalBelanja = 150000
diskon = 15
Langkah-langkah:

- Hitung diskon → 15 / 100 = 0.15
- Hitung total akhir → 150000 × (1 - 0.15)
- totalAkhir = 150000 × 0.85 = **127500**

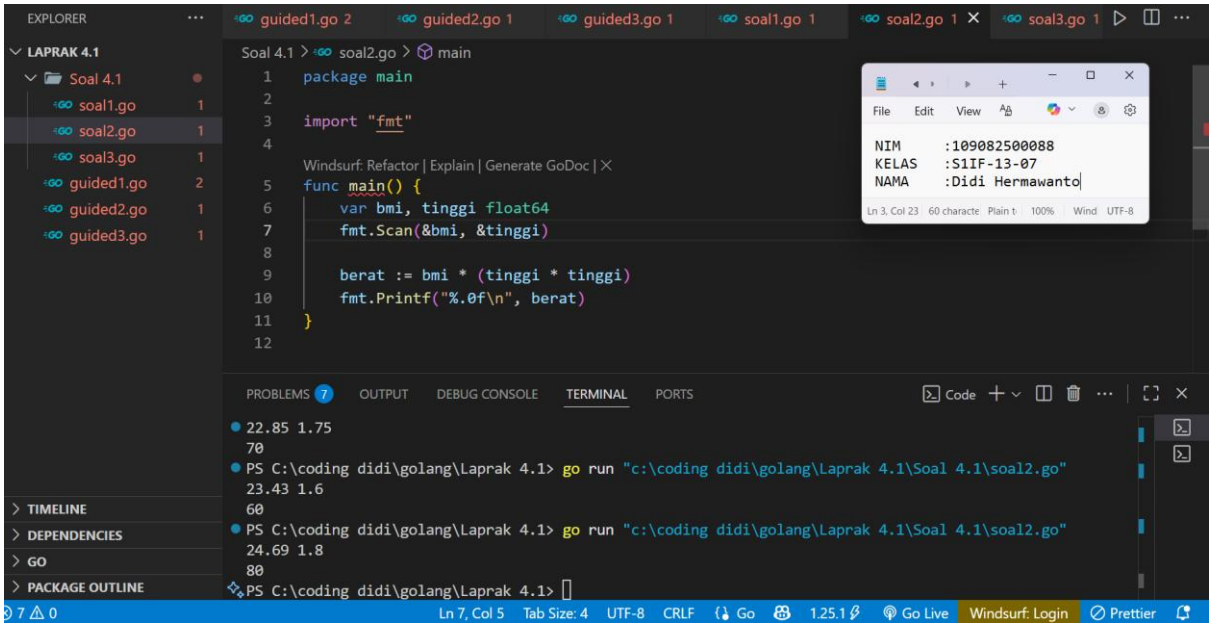
Output:
Total belanja setelah diskon: 127500

2. Tugas 2

Source code

```
package main
import "fmt"

func main() {
    var bmi, tinggi float64
      fmt.Scan(&bmi, &tinggi)

    berat := bmi * (tinggi * tinggi)
    fmt.Printf("%.0f\n", berat)
}
```

Screenshoot program



Deskripsi program

Program ini dibuat untuk menghitung berat badan seseorang berdasarkan nilai **BMI (Body Mass Index)** dan **tinggi badan** yang dimasukkan oleh pengguna. Jadi, kalau biasanya kita menghitung BMI dari berat dan tinggi, di program ini justru kebalikannya — kita cari berat badan dari nilai BMI dan tinggi badan yang sudah diketahui. Program ini cukup sederhana tapi sangat berguna, karena bisa membantu kita tahu berapa berat ideal seseorang berdasarkan nilai BMI-nya.

Saat program dijalankan, pengguna diminta untuk memasukkan dua nilai, yaitu **BMI** dan **tinggi badan dalam meter**. Setelah data dimasukkan, program akan memprosesnya menggunakan rumus:
berat = BMI × (tinggi × tinggi)
Rumus ini berasal dari rumus dasar BMI yaitu *BMI = berat / (tinggi × tinggi)*, yang kemudian diubah supaya bisa mencari berat badan. Setelah dihitung, hasil akhirnya akan ditampilkan di layar dalam satuan **kilogram**, dan dibulatkan menjadi angka tanpa desimal agar tampilannya lebih rapi dan mudah dibaca.

Di dalam kode program, digunakan dua variabel bertipe **float64** untuk menyimpan nilai BMI dan tinggi badan, karena data ini bisa mengandung angka desimal. Setelah itu, hasil perhitungannya disimpan ke variabel **berat** dan ditampilkan ke layar menggunakan perintah fmt.Printf("%.0f\n", berat) supaya hasilnya ditampilkan dalam bentuk angka bulat tanpa koma.

Algoritma Program:

1. Mulai program.
2. Siapkan variabel untuk menyimpan nilai BMI dan tinggi badan.
3. Minta pengguna memasukkan nilai BMI dan tinggi badan dalam meter.
4. Hitung berat badan dengan rumus: berat = BMI × (tinggi × tinggi).

5. Tampilkan hasil perhitungan berat badan ke layar.
6. Program selesai.

Contoh Jalannya Algoritma:
Misalnya pengguna memasukkan nilai **BMI = 22.85** dan **tinggi = 1.75**.
Maka langkah perhitungannya seperti berikut:

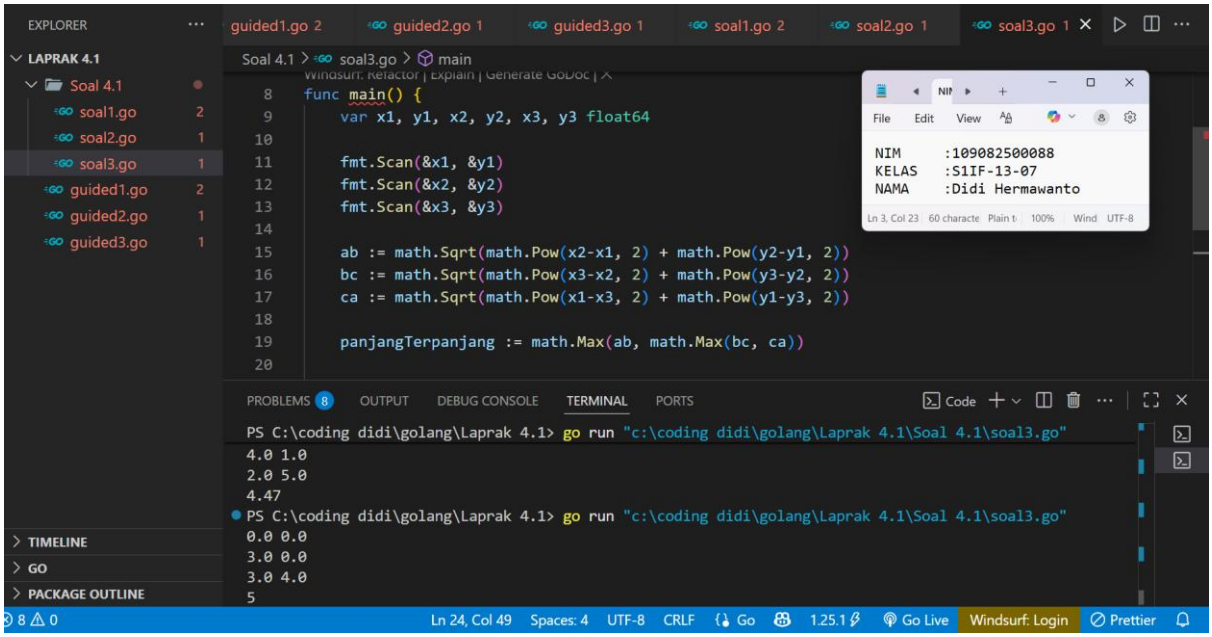
1. Program menghitung tinggi kuadrat: 1.75 × 1.75 = 3.0625
2. Lalu menghitung berat: 22.85 × 3.0625 = 70.03
3. Setelah dibulatkan, hasilnya menjadi **70**
   Jadi, program akan menampilkan **70** di layar, yang berarti berat badan orang tersebut sekitar **70 kilogram**.

3. Tugas 3

Source code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var x1, y1, x2, y2, x3, y3 float64

    fmt.Scan(&x1, &y1)
    fmt.Scan(&x2, &y2)
    fmt.Scan(&x3, &y3)

    ab := math.Sqrt(math.Pow(x2-x1, 2) + math.Pow(y2-y1, 2))
    bc := math.Sqrt(math.Pow(x3-x2, 2) + math.Pow(y3-y2, 2))
    ca := math.Sqrt(math.Pow(x1-x3, 2) + math.Pow(y1-y3, 2))

    panjangTerpanjang := math.Max(ab, math.Max(bc, ca))

    if panjangTerpanjang == math.Trunc(panjangTerpanjang) {
        fmt.Printf("%.0f\n", panjangTerpanjang)
    } else {
        fmt.Printf("%.2f\n", panjangTerpanjang)
    }
}
```

Screenshoot program



Deskripsi program

Program ini berfungsi untuk menghitung **panjang sisi terpanjang** dari sebuah segitiga yang dibentuk oleh tiga titik koordinat (A, B, dan C) pada bidang kartesius dua dimensi. Pengguna akan memasukkan enam angka, yaitu koordinat untuk tiga titik:

- Titik A = (x1, y1)
- Titik B = (x2, y2)
- Titik C = (x3, y3)

Setelah semua nilai dimasukkan, program akan menghitung panjang ketiga sisi segitiga (AB, BC, dan CA) menggunakan rumus jarak antar dua titik, kemudian mencari sisi yang paling panjang.

Program juga dibuat agar hasilnya ditampilkan secara rapi: kalau hasilnya bulat (misalnya 5.00), maka akan ditampilkan sebagai **5**, tapi kalau hasilnya punya desimal (misalnya 4.47), maka ditampilkan dengan dua angka di belakang koma.

Penjelasan Kode Program

package main

Bagian ini menunjukkan bahwa program ditulis menggunakan bahasa **Go** dan berada dalam paket utama, sehingga bisa dijalankan langsung.

import (
"fmt"
"math"
)

Bagian ini berfungsi untuk mengimpor dua library:

- fmt digunakan untuk input dan output (misalnya membaca data dan menampilkan hasil).
- math digunakan untuk operasi matematika seperti akar kuadrat (math.Sqrt) dan pemangkatan (math.Pow).

func main() {

var x1, y1, x2, y2, x3, y3 float64

Di sini program mendeklarasikan enam variabel tipe float64 untuk menyimpan nilai koordinat titik A, B, dan C.

Tipe float64 digunakan karena koordinat bisa berupa bilangan desimal, bukan hanya bilangan bulat.

fmt.Scan(&x1, &y1)

fmt.Scan(&x2, &y2)

fmt.Scan(&x3, &y3)

Ketiga baris ini membaca input dari pengguna.

Pengguna akan memasukkan nilai koordinat untuk ketiga titik (A, B, dan C) secara berurutan.

Contoh input:

1.0 1.0

4.0 1.0

1.0 5.0

ab := math.Sqrt(math.Pow(x2-x1, 2) + math.Pow(y2-y1, 2))

bc := math.Sqrt(math.Pow(x3-x2, 2) + math.Pow(y3-y2, 2))

ca := math.Sqrt(math.Pow(x1-x3, 2) + math.Pow(y1-y3, 2))

Bagian ini menghitung panjang ketiga sisi segitiga dengan rumus jarak antar dua titik:

Rumus umum:

[{Jarak} = {(x2 - x1)^2 + (y2 - y1)^2}]

- ab adalah panjang sisi antara titik A dan B.
- bc adalah panjang sisi antara titik B dan C.
- ca adalah panjang sisi antara titik C dan A.

Fungsi math.Pow(a, 2) digunakan untuk menghitung kuadrat dari suatu bilangan, dan math.Sqrt() untuk menghitung akar kuadrat.

panjangTerpanjang := math.Max(ab, math.Max(bc, ca))

Baris ini digunakan untuk mencari **sisi terpanjang** dari ketiga sisi segitiga.

Fungsi math.Max() membandingkan dua nilai dan mengembalikan nilai yang paling besar.

Di sini, math.Max(bc, ca) mencari yang terbesar antara BC dan CA, lalu dibandingkan lagi dengan AB untuk mendapatkan nilai tertinggi di antara ketiganya.

```
if panjangTerpanjang == math.Trunc(panjangTerpanjang) {
    fmt.Printf("%.0f\n", panjangTerpanjang)
} else {
    fmt.Printf("%.2f\n", panjangTerpanjang)
}
```

Bagian ini berfungsi untuk **mengatur format hasil output**.

- math.Trunc(panjangTerpanjang) akan memotong nilai desimal tanpa pembulatan.
- Jika hasilnya sama dengan nilai aslinya, berarti nilainya bulat → ditampilkan tanpa koma (%.0f).
- Kalau tidak sama (berarti punya desimal), ditampilkan dua angka di belakang koma (%.2f).

Jadi hasilnya lebih rapi dan mirip contoh di modul.

Algoritma Program

1. Mulai program.
2. Siapkan enam variabel untuk menyimpan koordinat titik A, B, dan C.
3. Minta pengguna memasukkan nilai koordinat dari ketiga titik.
4. Hitung panjang sisi AB, BC, dan CA dengan rumus jarak dua titik.
5. Tentukan sisi yang paling panjang menggunakan fungsi math.Max.
6. Cek apakah hasil sisi terpanjang berupa bilangan bulat atau desimal.
7. Jika bulat → tampilkan tanpa koma.
8. Jika desimal → tampilkan dua angka di belakang koma.
9. Program selesai.

Contoh Jalannya Program

Input:

1.0 1.0
4.0 1.0
1.0 5.0

Langkah perhitungan:

- AB = √((4−1)² + (1−1)²) = √(9 + 0) = 3.0
- BC = √((1−4)² + (5−1)²) = √(9 + 16) = 5.0
- CA = √((1−1)² + (1−5)²) = √(0 + 16) = 4.0

Sisi terpanjang = 5.0 → karena bulat, hasil yang ditampilkan:

5