

LAPORAN PRAKTIKUM
Algoritma Pemrograman

MODUL 4
EIKEL PRINST SUKATENDEL



Disusun oleh:
EIKEL PRINST SUKATENDEL
103112430232
S1IF-13-07

PROGRAM STUDI S1 INFORMATIKA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

LATIHAN SOAL

1. Tugas Latihan 1

Source code

```
package main

import "fmt"

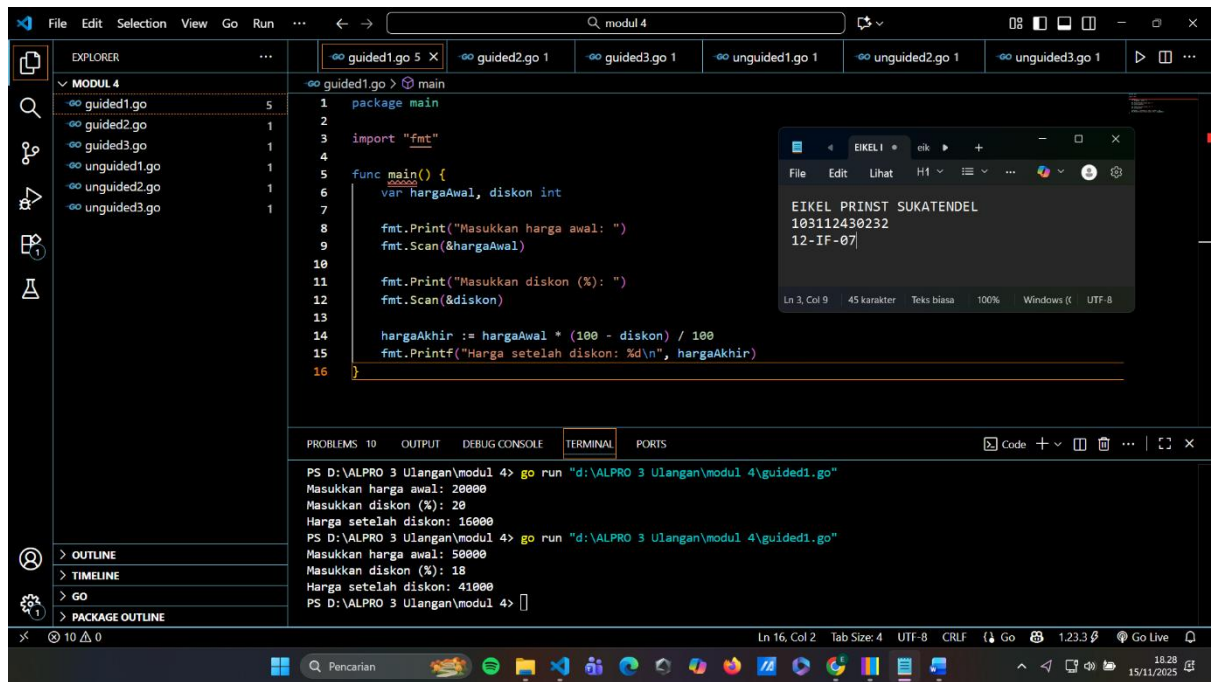
func main() {
    var hargaAwal, diskon int

    fmt.Print("Masukkan harga awal: ")
    fmt.Scan(&hargaAwal)

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

    hargaAkhir := hargaAwal * (100 - diskon) / 100
    fmt.Printf("Harga setelah diskon: %d\n", hargaAkhir)
}
```

Screenshoot program



Penjelasan

Kode program di atas digunakan untuk menghitung harga akhir setelah diberikan diskon. Program meminta pengguna memasukkan harga awal serta persentase diskon, lalu menghitung harga akhir dengan mengalikan harga awal dengan selisih antara 100 dan nilai diskon, kemudian membaginya dengan 100. Hasil perhitungan tersebut kemudian ditampilkan ke layar. Penamaan variabel dibuat lebih ringkas dan mengikuti gaya penulisan Go, sehingga kode tampak lebih rapi dan mudah dipahami.

Secara keseluruhan, program ini menunjukkan cara sederhana menggunakan input, operasi aritmatika, serta output dalam bahasa Go. Dengan struktur yang singkat namun jelas, kode ini mencerminkan gaya penulisan yang umum digunakan oleh mahasiswa dalam menyelesaikan tugas pemrograman dasar.

2. Tugas Latihan 2

Source code

```
package main

import "fmt"

func main() {

    var bmi, tinggi float64

    fmt.Print("Masukkan BMI: ")
```

```

    fmt.Scan(&bmi)

    fmt.Print("Masukkan tinggi (m): ")

    fmt.Scan(&tinggi)

    beratIdeal := bmi * tinggi * tinggi

    fmt.Printf("Berat badan ideal adalah: %d\n",
int(beratIdeal+0.5))
}

```

Screenshoot program

The screenshot shows a Go IDE with the following components:

- EXPLORER:** Lists files in 'MODUL 4': `guided1.go`, `guided2.go`, `guided3.go`, `unguided1.go`, `unguided2.go`, and `unguided3.go`.
- EDITOR:** Displays the code for `guided2.go`, which includes the `main` function with `fmt` package imports, `Scan` calls for BMI and height, a BMI calculation, and a `Printf` statement for the ideal weight.
- TERMINAL:** Shows the execution of the program. It prompts for BMI (70) and height (13), calculates an ideal weight of 11830, and then prompts for BMI (1) and height (12), calculating an ideal weight of 144.

Penjelasan

Program ini digunakan untuk menghitung berat badan ideal berdasarkan rumus BMI, di mana berat dihitung dari perkalian nilai BMI dengan kuadrat tinggi badan dalam meter. Program menerima input BMI dan tinggi dari pengguna, lalu memprosesnya menggunakan tipe data `float64` agar perhitungan menjadi lebih akurat. Hasil perhitungan kemudian dibulatkan dengan menambahkan 0.5 sebelum dikonversi ke integer sehingga output menjadi nilai berat yang lebih realistis.

Kode ini ditulis dengan gaya yang ringkas dan konsisten menggunakan *camelCase*, sesuai praktik umum mahasiswa dalam pemrograman Go. Struktur program dibuat sederhana sehingga mudah dipahami, mencakup penggunaan input, perhitungan matematika dasar, dan tampilan hasil ke layar dengan format yang rapi.

3. Guided 3

Source code

```
package main

import (
    "fmt"
    "math"
)

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

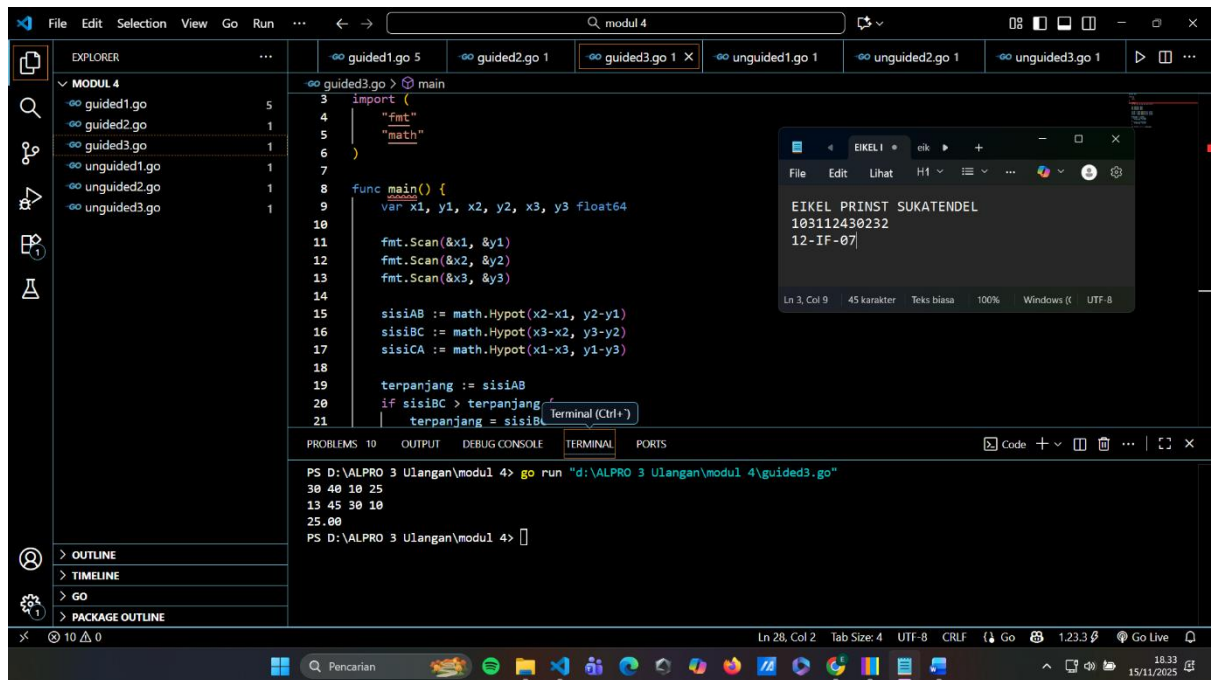
    fmt.Scan(&x1, &y1)
    fmt.Scan(&x2, &y2)
    fmt.Scan(&x3, &y3)

    sisiAB := math.Hypot(x2-x1, y2-y1)
    sisiBC := math.Hypot(x3-x2, y3-y2)
    sisiCA := math.Hypot(x1-x3, y1-y3)

    terpanjang := sisiAB
    if sisiBC > terpanjang {
        terpanjang = sisiBC
    }
    if sisiCA > terpanjang {
        terpanjang = sisiCA
    }

    fmt.Printf("%.2f\n", terpanjang)
}
```

Screenshoot program



Penjelasan

Program ini bertujuan mencari panjang sisi terpanjang dari sebuah segitiga berdasarkan tiga titik koordinat (A, B, dan C). Perhitungan jarak antar titik dilakukan menggunakan fungsi `math.Hypot`, yang secara otomatis menghitung akar dari jumlah kuadrat selisih koordinat sehingga rumus menjadi lebih singkat dan bersih. Setelah mendapatkan tiga panjang sisi—AB, BC, dan CA—program membandingkannya untuk menentukan sisi mana yang paling panjang.

Kode disusun dengan gaya penulisan yang sederhana dan mudah dibaca, sesuai standar mahasiswa yang belajar pemrograman Go. Tidak ada operasi yang berlebihan, variabelnya jelas, dan output diformat dengan dua angka di belakang koma sehingga hasil perhitungan tampil rapi dan akurat.

1. Unguided 1

Source code

```

package main

import "fmt"

func main() {
    var detik int
    fmt.Scan(&detik)
}

```

```

    jam := detik / 3600

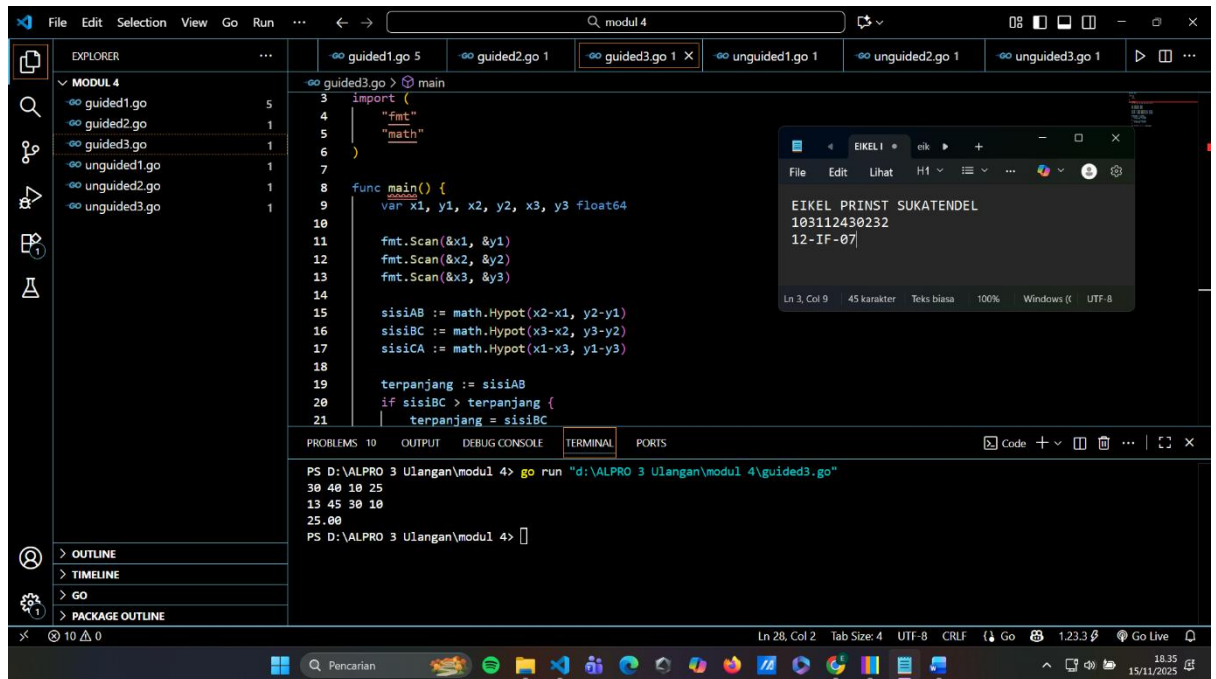
    menit := (detik % 3600) / 60

    sisaDetik := detik % 60

    fmt.Printf("%d jam %d menit dan %d detik\n", jam, menit,
sisaDetik)
}

```

Screenshoot program



Penjelasan

Program ini berfungsi mengubah total detik menjadi format waktu yang lebih mudah dipahami, yaitu jam, menit, dan detik. Prosesnya dilakukan dengan membagi input detik dengan 3600 untuk mendapatkan jumlah jam, lalu menggunakan sisa pembagian tersebut untuk menghitung menit, dan akhirnya sisa berikutnya menjadi detik. Dengan cara ini, waktu dapat diterjemahkan secara langsung ke bentuk yang lebih terstruktur.

Kode dituliskan secara ringkas dengan pemisahan variabel yang jelas dan penggunaan `fmt.Printf` agar output lebih rapi. Struktur dan gaya penulisan mencerminkan cara mahasiswa menulis program sederhana dalam Go, fokus pada kejelasan perhitungan dan hasil akhir yang mudah dibaca.

1. Unguided 2

Source code

```

package main

import "fmt"

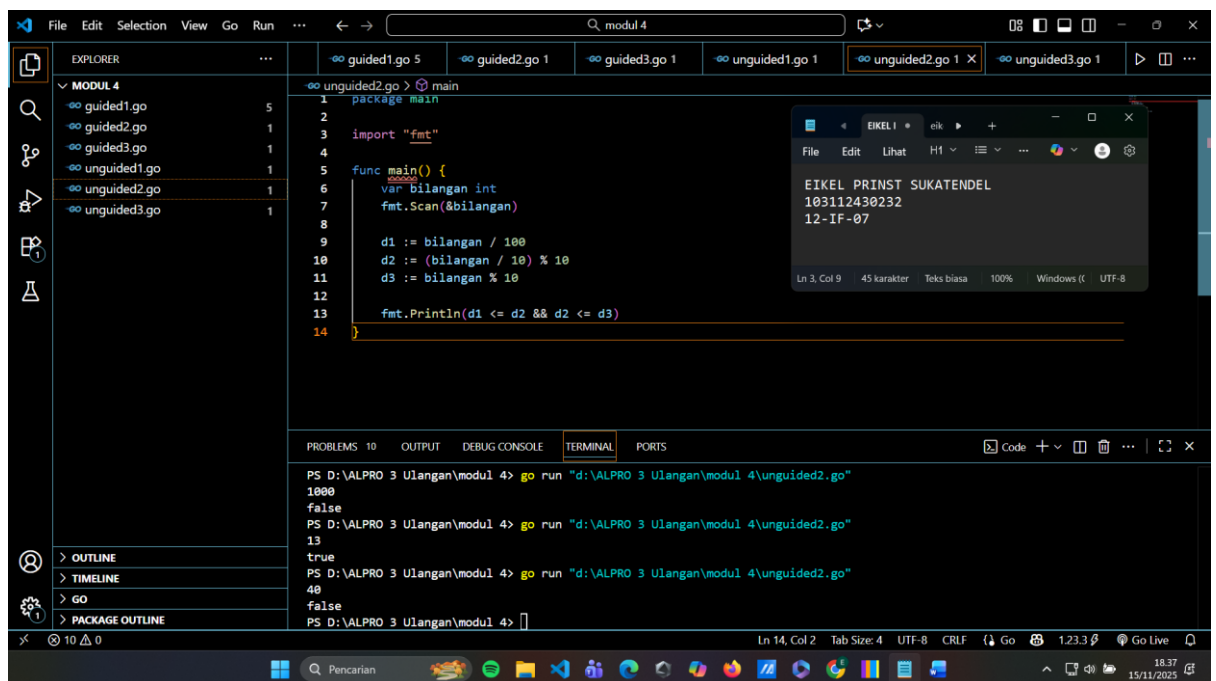
func main() {
    var bilangan int
    fmt.Scan(&bilangan)

    d1 := bilangan / 100
    d2 := (bilangan / 10) % 10
    d3 := bilangan % 10

    fmt.Println(d1 <= d2 && d2 <= d3)
}

```

Screenshot program



Penjelasan

Program ini digunakan untuk memeriksa apakah tiga digit pada sebuah bilangan tiga angka tersusun secara tidak menurun (ascending). Digit ratusan, puluhan, dan satuan diambil dengan operasi pembagian dan modulo, sehingga masing-masing digit dapat dipisahkan tanpa proses yang rumit. Setelah semua digit diperoleh, program membandingkannya dengan kondisi `d1 <= d2 && d2 <= d3` untuk menentukan apakah urutannya meningkat atau tetap.

Kode disusun dengan gaya yang ringkas dan variabel dibuat lebih jelas menggunakan *short declaration* agar lebih sesuai dengan standar penulisan Go bagi mahasiswa. Output program berupa nilai boolean (true atau false), sehingga pengguna dapat langsung mengetahui apakah bilangan tersebut memiliki pola digit yang berurutan.

3. Unguided 3

Source code

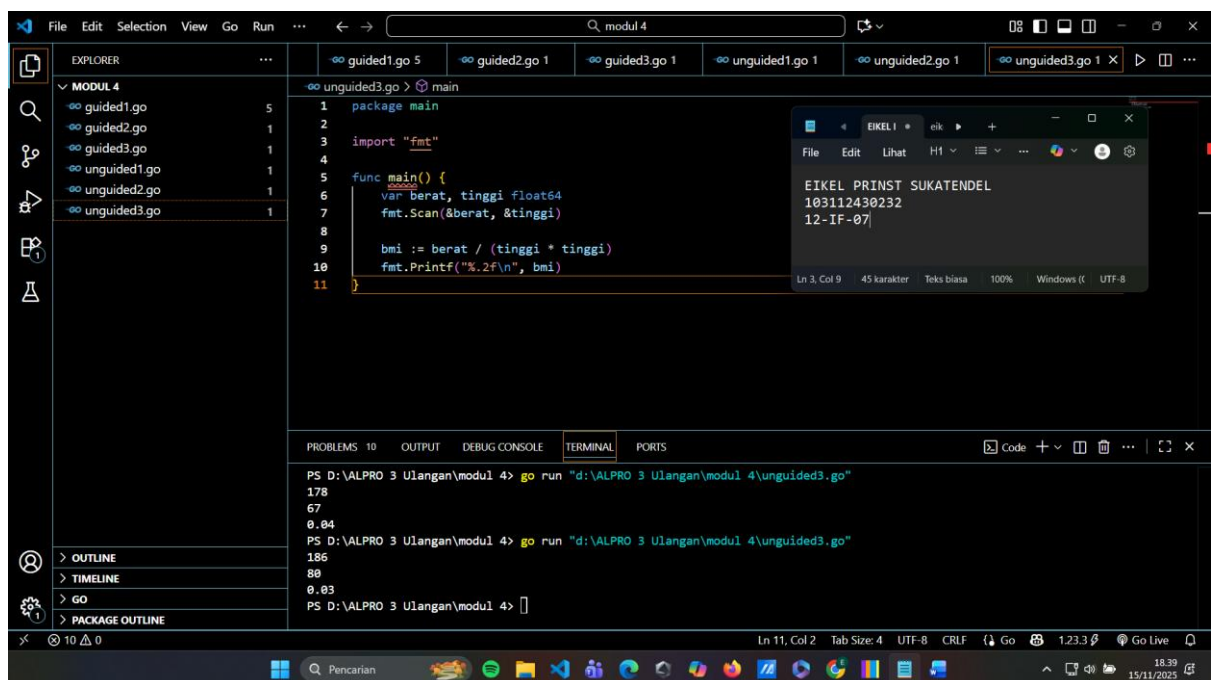
```
package main

import "fmt"

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

    bmi := berat / (tinggi * tinggi)
    fmt.Printf("%.2f\n", bmi)
}
```

Screenshot Program



Penjelasan

Program ini digunakan untuk menghitung nilai *Body Mass Index* (BMI) berdasarkan berat badan dan tinggi badan pengguna. Input yang diterima berupa berat dalam kilogram dan tinggi dalam meter, kemudian nilai BMI dihitung menggunakan rumus $\text{berat} / (\text{tinggi} * \text{tinggi})$ agar hasilnya akurat. Setelah itu, nilai BMI ditampilkan dengan format dua angka di belakang koma agar lebih rapi dan mudah dibaca.

Kode ditulis dengan gaya yang ringkas dan jelas menggunakan deklarasi variabel yang sederhana. Struktur program mengikuti praktik umum mahasiswa dalam pemrograman Go, yaitu fokus pada kejelasan perhitungan, penggunaan `fmt.Scan` untuk mengambil input, serta output terformat menggunakan `fmt.Printf`.