LAPORAN PRAKTIKUM

Algoritma Pemrograman

MODUL 10

ELSE-IF

Disusun oleh:

EDWARD ABIMAS SURYA HATTA

109082500171

S1IF-13-04

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

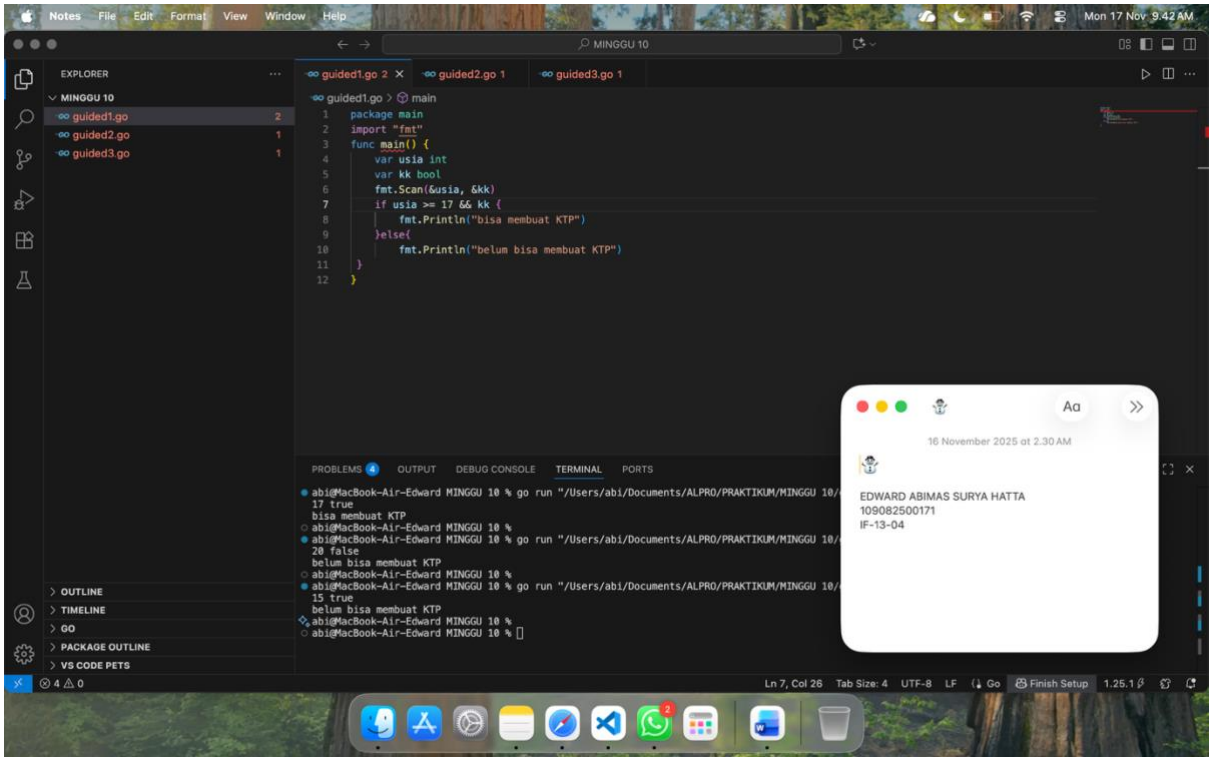
LATIHAN KELAS – GUIDED

1. **Guided 1**

Source Code

```
package main

import "fmt"

func main() {

    var usia int

    var kk bool

    fmt.Scan(&usia, &kk)

    if usia >= 17 && kk {

        fmt.Println("bisa membuat KTP")

    }else{

        fmt.Println("belum bisa membuat KTP")

 }

}
```

Screenshoot program



Deskripsi program

Program ini dibuat menggunakan bahasa pemrograman Go untuk menentukan kelayakan seseorang dalam membuat Kartu Tanda Penduduk (KTP) berdasarkan dua parameter input. Kode diawali dengan mendeklarasikan dua variabel, yaitu usiayang bertipe integer (bilangan bulat) dan kk yang bertipe boolean (bernilai *true* atau *false*). Program kemudian menggunakan perintah fmt.Scan untuk membaca dua input sekaligus dari pengguna; input pertama akan disimpan sebagai angka usia, sedangkan input kedua akan menentukan status ketersediaan dokumen Kartu Keluarga (KK).

Inti dari logika program ini terletak pada struktur percabangan if yang menggunakan operator logika AND (simbol &&). Program memeriksa apakah kedua syarat terpenuhi secara bersamaan: variabel usia harus bernilai 17 atau lebih, **dan**variabel kk harus bernilai *true* (artinya memiliki KK). Jika kedua kondisi tersebut benar, program akan mencetak pesan "bisa membuat KTP". Namun, jika salah satu saja dari syarat tersebut tidak terpenuhi—misalnya umur sudah 17 tapi tidak membawa KK (kk bernilai *false*), atau sebaliknya—maka program akan otomatis masuk ke blok else dan menampilkan pesan "belum bisa membuat KTP".

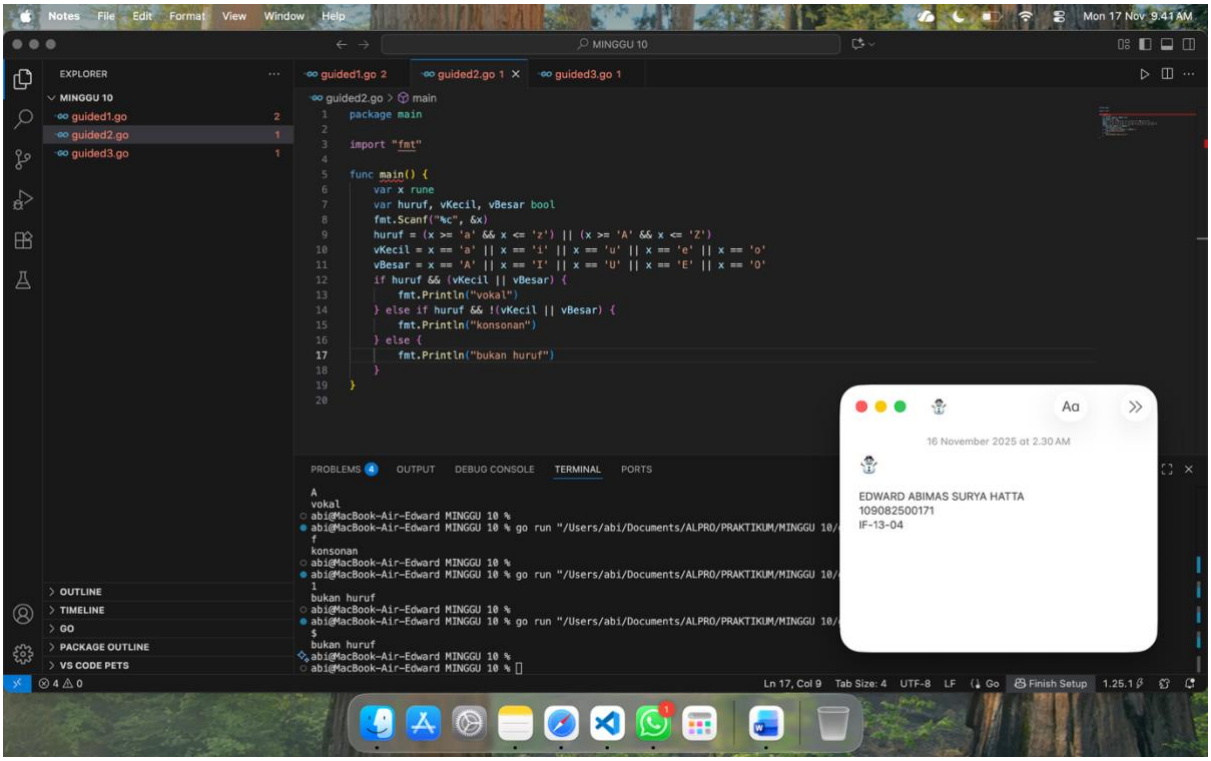
2. Guided 2

Source Code

```
package main


import "fmt"
```

```
func main() {
    var x rune
    var huruf, vKecil, vBesar bool
    fmt.Scanf("%c", &x)
    huruf = (x >= 'a' && x <= 'z') || (x >= 'A' && x <=
'Z')
    vKecil = x == 'a' || x == 'i' || x == 'u' || x ==
'e' || x == 'o'
    vBesar = x == 'A' || x == 'I' || x == 'U' || x ==
'E' || x == 'O'
    if huruf && (vKecil || vBesar) {
        fmt.Println("vokal")
    } else if huruf && !(vKecil || vBesar) {
        fmt.Println("konsonan")
    } else {
        fmt.Println("bukan huruf")
    }
}
```

Screenshoot program



Deskripsi program

Program ini dirancang menggunakan bahasa pemrograman Go untuk mengidentifikasi jenis karakter yang dimasukkan oleh pengguna, apakah itu huruf vokal, konsonan, atau karakter non-huruf (seperti angka atau simbol). Pada bagian awal, program mendeklarasikan variabel x dengan tipe data rune untuk menyimpan karakter tunggal, serta tiga variabel bertipe boolean (huruf, vKecil, vBesar) yang berfungsi sebagai penanda kondisi logika. Program kemudian meminta input satu karakter dari pengguna menggunakan fungsi fmt.Scanf dengan format %c.

Setelah karakter diterima, program melakukan evaluasi logika untuk menentukan status karakter tersebut. Variabel hurufakan bernilai *true* jika input berada dalam rentang alfabet a-z atau A-Z. Variabel vKecil memeriksa secara spesifik apakah karakter tersebut adalah salah satu dari 'a', 'i', 'u', 'e', 'o', sedangkan vBesar melakukan pemeriksaan yang sama untuk versi huruf kapitalnya. Langkah ini memisahkan logika pendeteksian jenis karakter ke dalam variabel-variabel boolean terpisah sebelum masuk ke tahap percabangan.

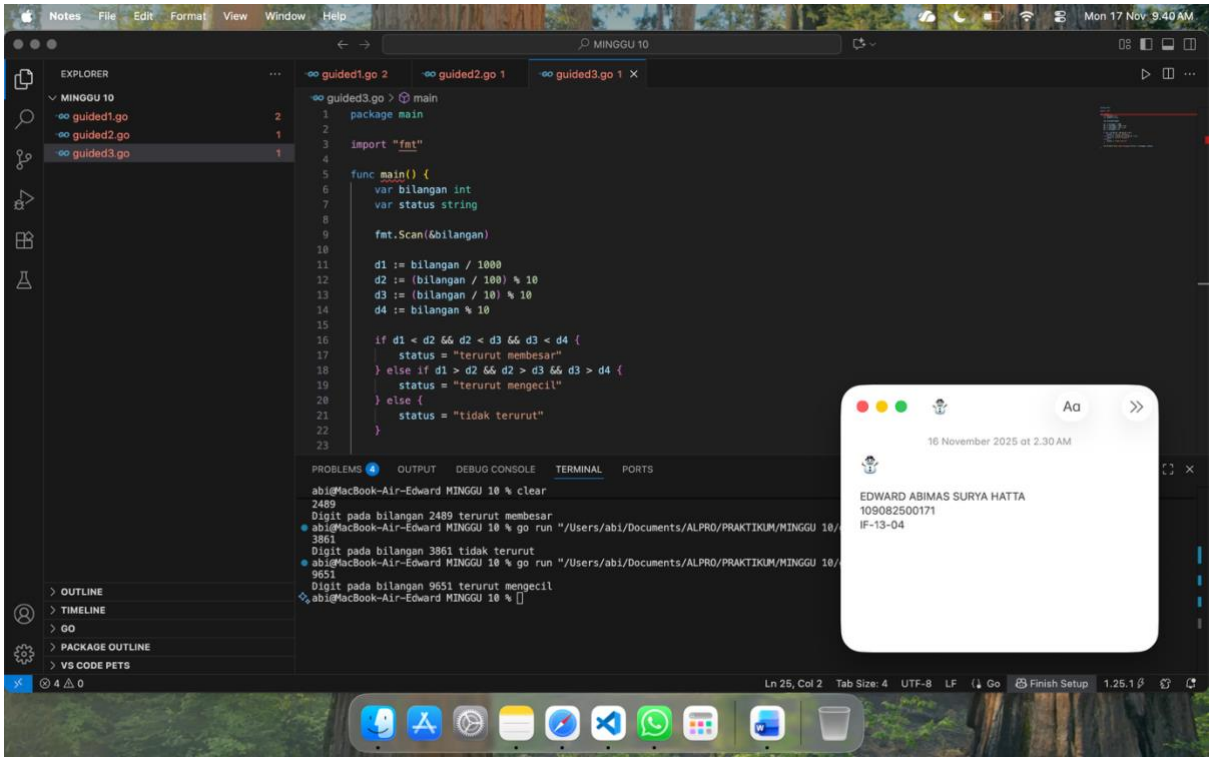
Pada tahap akhir, struktur kontrol if-else digunakan untuk menentukan keluaran. Kondisi pertama memeriksa apakah karakter tersebut valid sebagai sebuah huruf dan termasuk dalam kategori vokal (baik kecil maupun besar); jika terpenuhi, program mencetak "vokal". Jika kondisi pertama gagal, program memeriksa kondisi kedua: apakah karakter tersebut adalah huruf namun bukan vokal (menggunakan logika negasi !), yang secara otomatis mengartikannya sebagai "konsonan". Apabila kedua kondisi di atas tidak terpenuhi (artinya input bukan termasuk alfabet), program akan mengeksekusi blok else terakhir dan mencetak "bukan huruf".

3. **Guided 3**

Source Code

```
package main

import "fmt"

func main() {
    var bilangan int
    var status string

    fmt.Scan(&bilangan)

    d1 := bilangan / 1000
    d2 := (bilangan / 100) % 10
    d3 := (bilangan / 10) % 10
    d4 := bilangan % 10

    if d1 < d2 && d2 < d3 && d3 < d4 {
        status = "terurut membesar"
    } else if d1 > d2 && d2 > d3 && d3 > d4 {
        status = "terurut mengecil"
    } else {
        status = "tidak terurut"
    }

    fmt.Printf("Digit pada bilangan %d %s\n", bilangan,
status)
}
```

Screenshoot program



Deskripsi program

Program ini ditulis dalam bahasa Go untuk menganalisis pola urutan angka-angka penyusun pada sebuah bilangan bulat (khususnya bilangan empat digit). Setelah mendeklarasikan variabel bilangan untuk menampung input dan status untuk menyimpan hasil deskripsi, program membaca input dari pengguna. Tahap terpenting dalam kode ini adalah proses ekstraksi digit, di mana bilangan bulat tersebut dipecah menjadi empat bagian terpisah (d1 hingga d4) menggunakan kombinasi operasi pembagian bilangan bulat dan modulus (sisa bagi). Variabel d1 mengambil digit ribuan, sedangkan d2, d3, dan d4 mengambil digit ratusan, puluhan, dan satuan secara berturut-turut dengan membagi bilangan sesuai posisinya lalu mengambil sisa bagi 10.

Setelah keempat digit berhasil dipisahkan, program melakukan evaluasi menggunakan struktur percabangan if-else. Program membandingkan digit-digit tersebut secara berurutan dari kiri ke kanan. Jika setiap digit selalu lebih kecil dari digit di sebelah kanannya (d1 < d2 < d3 < d4), variabel status diisi dengan "terurut membesar". Sebaliknya, jika setiap digit selalu lebih besar dari digit kanannya (d1 > d2 > d3 > d4), status diisi dengan "terurut mengecil". Jika susunan angkanya acak atau memiliki angka yang sama (tidak memenuhi kedua pola ketat di atas), maka program menetapkan status sebagai "tidak terurut". Hasil akhir kemudian ditampilkan menggunakan fmt.Printf yang menggabungkan nilai bilangan asli dengan status urutannya.

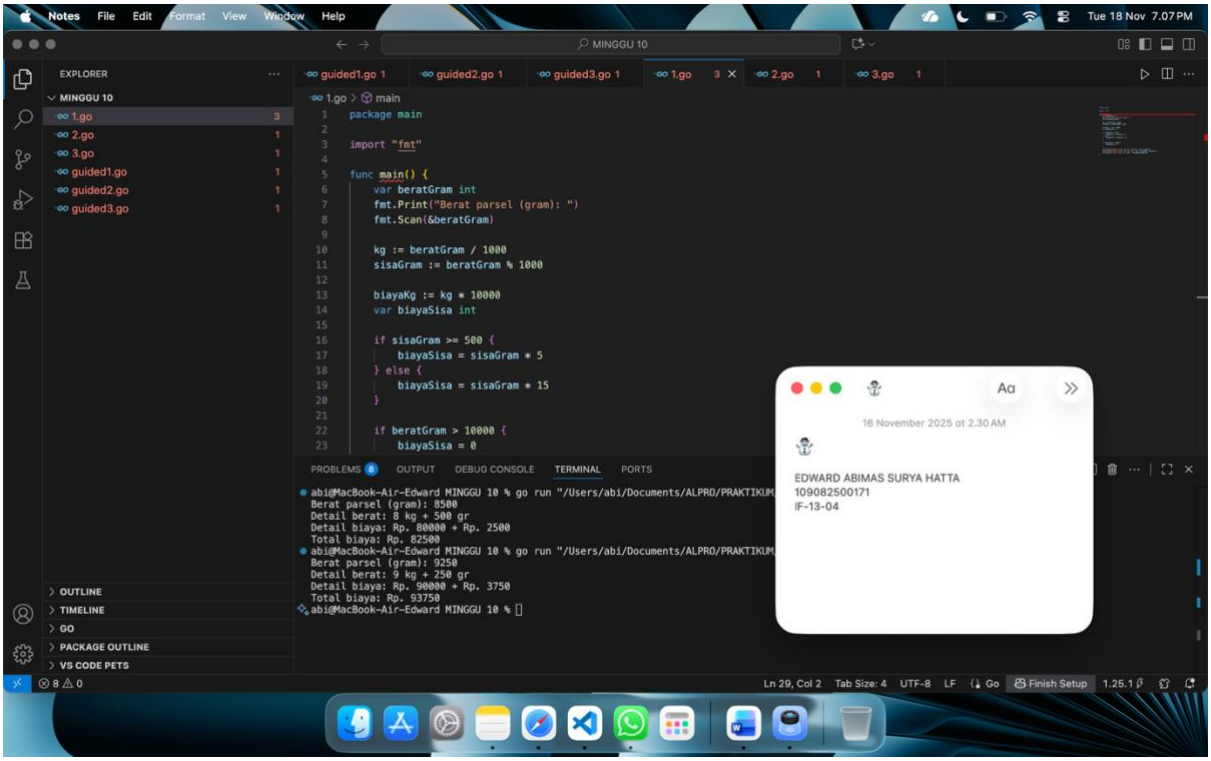
TUGAS

1. **Tugas 1**

   Source code

```
package main


import "fmt"


func main() {
    var beratGram int
    fmt.Print("Berat parsel (gram): ")
    fmt.Scan(&beratGram)


    kg := beratGram / 1000
    sisaGram := beratGram % 1000


    biayaKg := kg * 10000
    var biayaSisa int


    if sisaGram >= 500 {
        biayaSisa = sisaGram * 5
    } else {
        biayaSisa = sisaGram * 15
    }


    if beratGram > 10000 {
        biayaSisa = 0
    }


    fmt.Printf("Detail berat: %d kg + %d gr\n", kg, sisaGram)
    fmt.Printf("Detail biaya: Rp. %d + Rp. %d\n", biayaKg,
biayaSisa)
    fmt.Printf("Total biaya: Rp. %d\n", biayaKg+biayaSisa)
}
```

Screenshoot program



Deskripsi program

Kode program yang ditulis dalam bahasa Go ini dirancang sebagai sistem penghitung biaya pengiriman paket yang cerdas, di mana prosesnya dimulai dari inisialisasi paket main dan pemanggilan pustaka fmt untuk keperluan input-output standar. Alur eksekusi dimulai di dalam fungsi main, di mana program pertama-tama mendeklarasikan sebuah variabel bertipe integer bernama beratGram. Variabel ini bertugas sebagai wadah untuk menampung data mentah yang dimasukkan oleh pengguna setelah perintah fmt.Scan dieksekusi. Pada tahap ini, program mengharapkan pengguna memasukkan angka bulat yang merepresentasikan berat parsel secara keseluruhan dalam satuan gram, yang kemudian alamat memorinya diambil untuk menyimpan nilai tersebut agar bisa diproses lebih lanjut.

Setelah data berat diterima, program melakukan proses matematika untuk memisahkan satuan berat menjadi dua komponen, yaitu kilogram dan gram. Program menggunakan operator pembagian bilangan bulat (/) untuk mendapatkan nilai kilogram (kg), di mana sisa desimal akan diabaikan, dan operator modulus atau sisa bagi (%) untuk mendapatkan nilai sisa gram (sisaGram) yang tidak mencapai satu kilogram penuh. Logika ini penting karena struktur biaya dibedakan antara berat kiloan yang utuh dan sisa gramnya. Segera setelah pemisahan ini, program menghitung biaya dasar untuk berat kiloan dengan mengalikan variabel kg dengan tarif standar Rp. 10.000, dan hasilnya disimpan dalam variabel biayaKg.

Kompleksitas logika program meningkat saat menentukan biaya untuk sisa gram (biayaSisa). Program menggunakan struktur kontrol if-else untuk mengevaluasi jumlah sisa gram tersebut. Jika sisa gram ternyata lebih besar atau sama dengan 500 gram, program menerapkan tarif yang lebih murah, yaitu Rp. 5 dikalikan jumlah sisa gram. Sebaliknya, jika sisa gram kurang dari 500 gram, program menerapkan tarif

penalti yang lebih tinggi, yaitu Rp. 15 dikalikan jumlah sisa gram. Mekanisme ini menciptakan sistem tarif bertingkat yang dinamis tergantung pada seberapa "nanggung" berat paket tersebut.

Namun, sebelum perhitungan selesai, terdapat satu lapisan logika bisnis tambahan yang berfungsi sebagai aturan diskon. Program melakukan pengecekan ulang terhadap total berat awal (beratGram). Jika berat total paket melebihi 10.000 gram (atau 10 kilogram), program secara otomatis menimpa nilai biayaSisa yang sebelumnya telah dihitung menjadi nol. Ini berarti untuk paket-paket yang sangat berat, pengguna diberikan keringanan biaya berupa penggratisan biaya tambahan untuk sisa gramnya, sehingga mereka hanya perlu membayar biaya per kilogram utuhnya saja.

Pada tahap akhir, program menyajikan seluruh hasil kalkulasi tersebut kepada pengguna menggunakan fungsi fmt.Printf. Fungsi ini memformat data integer ke dalam string yang mudah dibaca manusia. Program akan mencetak tiga baris informasi secara berurutan: baris pertama menampilkan konversi berat dalam format "kg + gr", baris kedua merinci komponen biaya yang harus dibayar (biaya per kg ditambah biaya sisa), dan baris ketiga menjumlahkan kedua komponen biaya tersebut untuk menampilkan total akhir yang harus dilunasi oleh pengirim. Dengan demikian, program ini menyelesaikan tugasnya dari penerimaan data mentah hingga penyajian struk pembayaran yang akurat dan terperinci.

2. **Tugas 2**

a. Jika nam diberikan adalah 80.1, apa keluaran dari program tersebut? Apakah eksekusi program tersebut sesuai spesifikasi soal?

jawab: Jika nilai 80.1 dimasukkan, secara teknis program akan mengalami kegagalan kompilasi atau *error* karena terdapat kesalahan tipe data yang fatal di mana variabel nam yang dideklarasikan sebagai bilangan desimal (float64) dipaksa untuk menyimpan nilai teks huruf mutu ("A", "AB", dll). Namun, apabila kita mengabaikan kesalahan sintaks tersebut dan hanya menelusuri alur logikanya saja, keluaran akhirnya adalah "D" karena penggunaan struktur if yang terpisah-pisah mengakibatkan nilai yang seharusnya "A" tertimpa terus-menerus oleh kondisi-kondisi di bawahnya yang juga bernilai benar (karena 80.1 juga lebih besar dari 40). Eksekusi ini jelas tidak sesuai dengan spesifikasi soal yang seharusnya menghasilkan "A" untuk nilai di atas 80.

b. Apa saja kesalahan dari program tersebut? Mengapa demikian? Jelaskan alur program seharusnya!

jawab: Kesalahan program tersebut meliputi kesalahan sintaksis di mana variabel input nam (float) digunakan untuk menyimpan output huruf mutu padahal seharusnya menggunakan variabel nmk (string) yang sudah dideklarasikan, serta kesalahan logika karena menggunakan banyak if yang berdiri sendiri tanpa penghubung else. Hal ini menyebabkan inefisiensi komputasi dan kesalahan hasil karena komputer akan memeriksa semua kondisi satu per satu, sehingga nilai yang memenuhi kriteria tinggi akan tertimpa oleh kriteria yang lebih rendah. Alur program yang seharusnya diterapkan adalah struktur percabangan bertingkat if-else if agar pemeriksaan kondisi bersifat eksklusif atau saling

lepas, yang berarti ketika satu kondisi terpenuhi, program akan langsung mencetak hasil dan mengabaikan sisa kondisi lainnya.

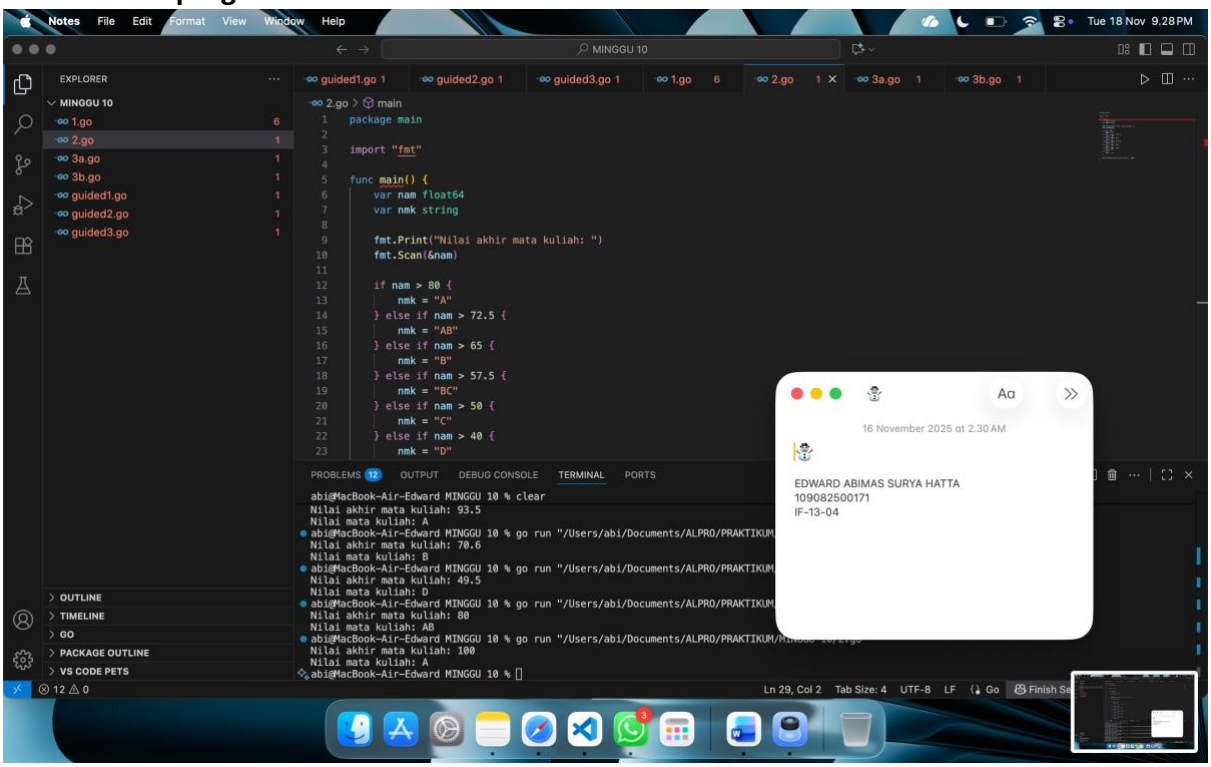
c. Perbaiki program tersebut! Ujilah dengan masukan: 93.5; 70.6; dan 49.5. Seharusnya keluaran yang diperoleh adalah 'A', 'B', dan 'D'.

Source code

```
package main


import "fmt"


func main() {
    var nam float64
    var nmk string


    fmt.Print("Nilai akhir mata kuliah: ")
    fmt.Scan(&nam)


    if nam > 80 {
        nmk = "A"
    } else if nam > 72.5 {
        nmk = "AB"
    } else if nam > 65 {
        nmk = "B"
    } else if nam > 57.5 {
        nmk = "BC"
    } else if nam > 50 {
        nmk = "C"
    } else if nam > 40 {
```

```
            nmk = "D"

        } else {

            nmk = "E"

        }


        fmt.Println("Nilai mata kuliah:", nmk)

    }
```

Screenshoot program



Deskripsi program

Program tersebut ditulis menggunakan bahasa pemrograman Go (Golang) dengan tujuan untuk mengonversi nilai angka menjadi nilai huruf (indeks prestasi). Program dimulai dengan mendefinisikan paket utama main dan mengimpor pustaka fmt yang berfungsi untuk menangani proses input dan output data. Di dalam fungsi utama, terdapat deklarasi dua variabel, yaitu nam dengan tipe data float64 untuk menyimpan nilai angka yang bisa memuat desimal, dan nmk dengan tipe data stringuntuk menyimpan hasil konversi berupa huruf.

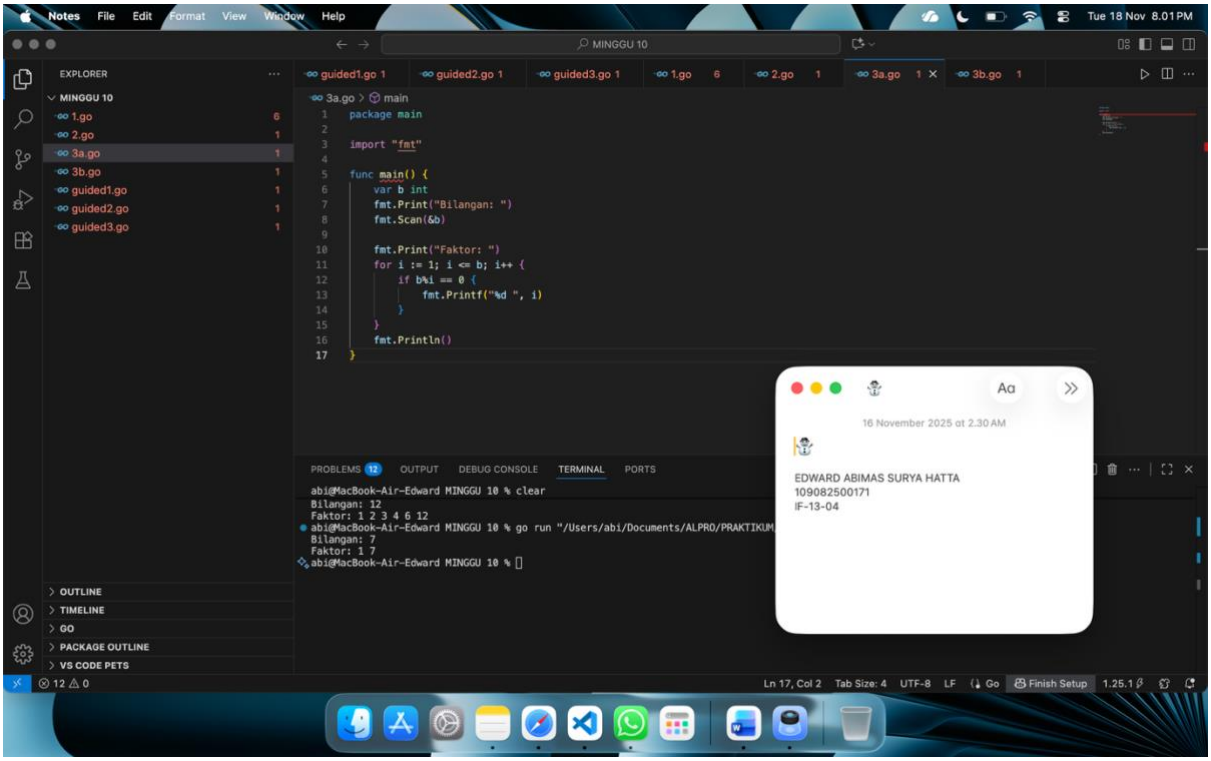
Program kemudian akan meminta pengguna untuk memasukkan "Nilai akhir mata kuliah", lalu input tersebut dibaca dan disimpan ke dalam variabel nam.
Setelah nilai angka diterima, program menjalankan logika percabangan if-else untuk menentukan kategori huruf mutunya. Pengecekan dilakukan secara berurutan dari batas nilai tertinggi. Jika nilai nam lebih dari 80, maka nmk diatur menjadi "A". Jika tidak, program lanjut memeriksa apakah nilai di atas 72.5 untuk mendapatkan "AB", di atas 65 untuk "B", di atas 57.5 untuk "BC", di atas 50 untuk "C", atau di atas 40 untuk "D". Apabila semua kondisi tersebut tidak terpenuhi, yang berarti nilai input adalah 40 atau kurang, maka program akan mengeksekusi blok else terakhir dan memberikan nilai "E".
Pada bagian akhir, setelah proses penentuan nilai huruf selesai, program akan mencetak hasilnya ke layar menggunakan fungsi fmt.Println. Output yang ditampilkan adalah teks "Nilai mata kuliah:" diikuti dengan huruf yang tersimpan dalam variabel nmk. Dengan demikian, pengguna dapat langsung melihat hasil konversi dari angka yang mereka masukkan sebelumnya.

3. **Tugas 3a**

Source code

```
package main

import "fmt"

func main() {
    var b int
    fmt.Print("Bilangan: ")
    fmt.Scan(&b)

    fmt.Print("Faktor: ")
    for i := 1; i <= b; i++ {
        if b%i == 0 {
            fmt.Printf("%d ", i)
        }
    }
    fmt.Println()
}
```

Screenshoot program



Deskripsi program

Program ini ditulis dalam bahasa pemrograman Go dan bertujuan untuk mencari serta menampilkan semua faktor pembagi dari sebuah bilangan bulat. Kode diawali dengan mendefinisikan paket main dan mengimpor pustaka fmt untuk keperluan input dan output. Di dalam fungsi utama, program mendeklarasikan variabel b bertipe integer, lalu meminta pengguna memasukkan sebuah angka dengan menampilkan pesan "Bilangan: ". Angka yang diketik pengguna kemudian dibaca oleh perintah Scan dan disimpan ke dalam variabel b.

Setelah menerima input, program mencetak teks "Faktor: " sebagai label hasil. Selanjutnya, program menjalankan sebuah perulangan (*loop*) yang dimulai dari angka 1 dan terus bertambah hingga mencapai nilai yang sama dengan b. Dalam setiap putaran perulangan tersebut, terdapat logika pengecekan menggunakan operasi modulus (sisa bagi). Program memeriksa apakah bilangan b habis dibagi oleh angka penanda perulangan saat ini (i). Jika hasil pembagian tidak menyisakan sisa (sisa bagi sama dengan 0), maka angka tersebut adalah faktor dari b.

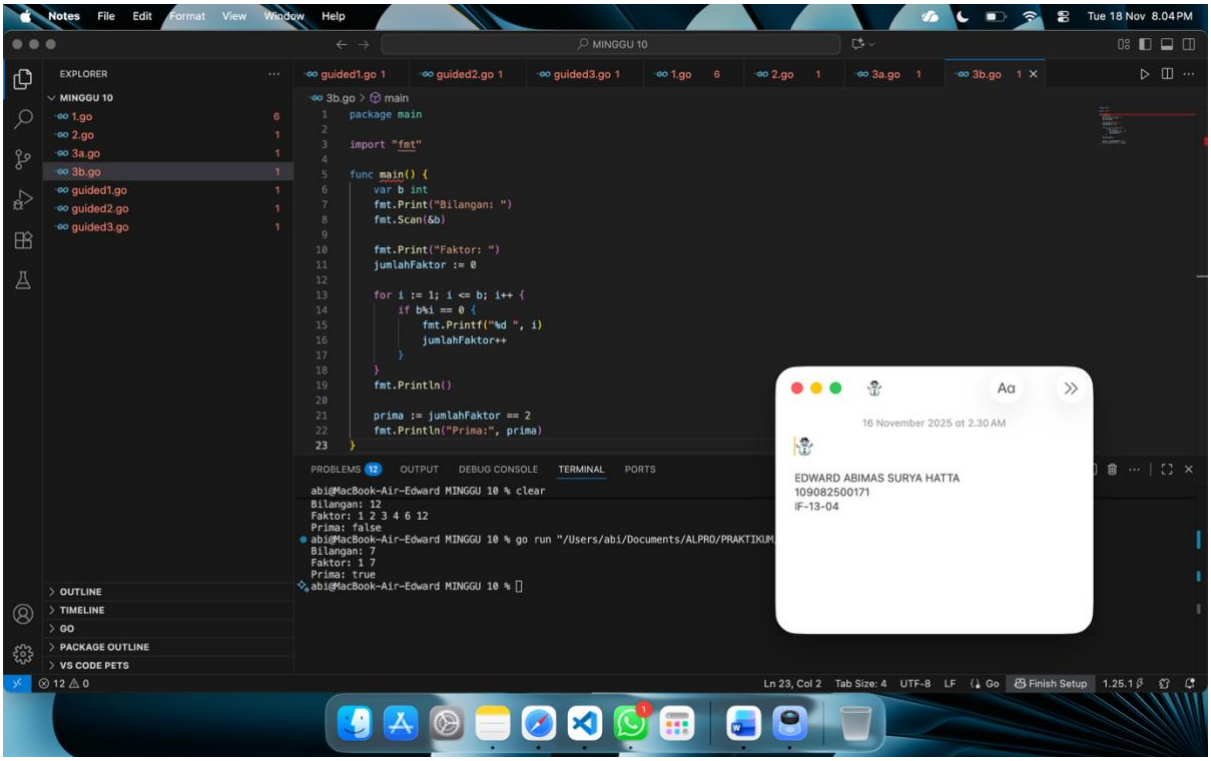
Setiap kali faktor ditemukan, angka tersebut langsung dicetak ke layar di baris yang sama, diikuti oleh sebuah spasi agar angka tidak berdempetan. Proses ini terus berlangsung sampai perulangan selesai memeriksa semua angka hingga b. Setelah seluruh faktor ditampilkan, program menutup eksekusi dengan mencetak baris baru (Println) agar tampilan terminal menjadi rapi dan kursor berpindah ke baris berikutnya.

4. **Tugas 3b**

   Source code

```
package main

import "fmt"

func main() {
    var b int
    fmt.Print("Bilangan: ")
    fmt.Scan(&b)

    fmt.Print("Faktor: ")
    jumlahFaktor := 0

    for i := 1; i <= b; i++ {
        if b%i == 0 {
            fmt.Printf("%d ", i)
            jumlahFaktor++
        }
    }
    fmt.Println()

    prima := jumlahFaktor == 2
    fmt.Println("Prima:", prima)
}
```

Screenshoot program



Deskripsi program

Program ini ditulis menggunakan bahasa pemrograman Go dengan tujuan untuk menguraikan semua faktor pembagi dari sebuah bilangan bulat sekaligus menentukan apakah bilangan tersebut termasuk bilangan prima. Kode dimulai dengan mengimpor paket fmt untuk keperluan interaksi input-output. Pada bagian awal fungsi utama, variabel b dideklarasikan untuk menampung input pengguna. Program akan meminta pengguna memasukkan angka melalui pesan "Bilangan:", kemudian angka tersebut dibaca dan disimpan. Selain itu, sebuah variabel penghitung bernama jumlahFaktor diinisialisasi dengan nilai 0 untuk melacak berapa banyak faktor yang ditemukan selama proses berjalan.

Mekanisme utama program terletak pada struktur perulangan (*loop*) yang melakukan iterasi mulai dari angka 1 hingga mencapai nilai bilangan b. Di dalam setiap putaran, program melakukan pengecekan menggunakan operasi modulus (sisa bagi). Apabila bilangan b habis dibagi oleh angka iterasi saat ini, maka angka tersebut dianggap sebagai faktor, dicetak ke layar, dan nilai variabel jumlahFaktor akan ditambahkan satu. Proses ini memastikan bahwa setiap pembagi yang valid ditampilkan secara berurutan dan dihitung jumlah totalnya.

Setelah perulangan selesai, program melakukan evaluasi akhir untuk menentukan status bilangan prima. Logika yang digunakan didasarkan pada definisi bahwa bilangan prima memiliki tepat dua faktor pembagi, yaitu angka 1 dan bilangan itu sendiri. Program memeriksa kondisi apakah nilai jumlahFaktor sama dengan 2. Hasil evaluasi logika ini disimpan dalam variabel boolean bernama prima (yang akan bernilai *true* atau *false*), dan kemudian dicetak ke layar sebagai hasil akhir.