LAPORAN PRAKTIKUM ALGORITMA

DAN PEMROGRAMAN 1

MODUL 10 – ELSE-IF

ALGORITMA DAN PEMOGRAMAN 1

Disusun oleh:

NAMA : PRIMATAMA SIGALINGGING

NIM : 109082500076

S1IF-13-07

Asisten Praktikum

Adithana dharma putra

Apri pandu wicaksono

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

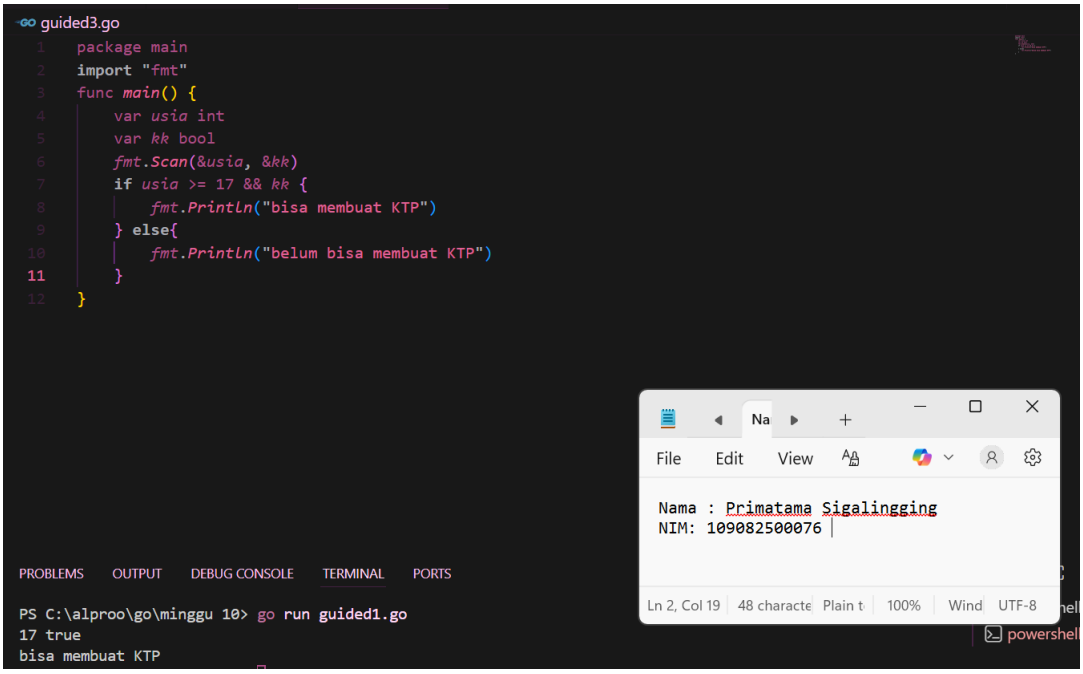
LATIHAN KELAS – GUIDED

1. Guided 1

Source Code

```
package main

import "fmt"

func main() {

    var usia int

    var punyaKK bool

    fmt.Scan(&usia)

    fmt.Scan(&punyaKK)

    if usia >= 17 && punyaKK {

        fmt.Println("bisa membuat KTP")

    }else {

        fmt.Println("belum bisa membuat KTP")

    }

}
```

Screenshoot program



Deskripsi program

- **Tujuan**

  Program ini dirancang untuk mengevaluasi apakah seseorang telah memenuhi ketentuan untuk membuat KTP. Dua hal yang diperiksa adalah usia minimal 17 tahun serta kepemilikan Kartu Keluarga (KK). Dengan adanya program ini, pengguna dapat mengetahui status kelayakan secara cepat tanpa harus mengecek syarat-syarat tersebut secara manual.

- **Proses**

  Ketika dijalankan, program meminta pengguna mengisi dua informasi: usia dan apakah memiliki KK (true/false). Setelah itu, program melakukan pengecekan menggunakan logika percabangan.

  Jika usia 17 tahun atau lebih, dan pengguna memiliki KK, program akan menampilkan pesan “bisa membuat KTP”.

  Jika salah satu syarat tidak terpenuhi, output yang muncul adalah “belum bisa membuat KTP”.

  Seluruh pemeriksaan dilakukan otomatis oleh komputer, sehingga hasil dapat diketahui dengan cepat dan akurat.

- **Kesimpulan**

  Program ini memberikan gambaran tentang bagaimana struktur if–else bekerja dalam menentukan keputusan berdasarkan input. Meskipun sederhana, konsep ini sangat penting karena banyak aplikasi nyata menggunakan logika kondisi seperti ini.

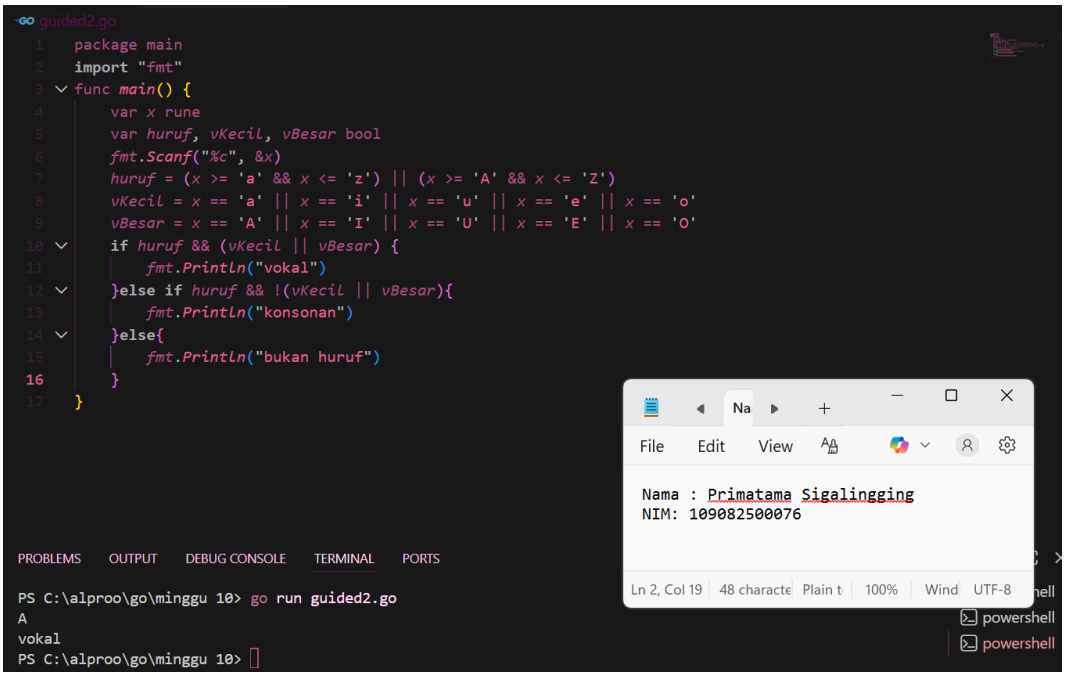
2. Guided 2

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

- **Tujuan**

  Tujuan program ini adalah untuk mengenali apakah karakter yang dimasukkan pengguna merupakan huruf vokal, huruf konsonan, atau bukan huruf. Program ini juga membantu memperkenalkan cara kerja logika kondisi dalam pengolahan karakter.

- **Proses**

  Saat program dijalankan, pengguna diminta memasukkan sebuah karakter. Program kemudian melakukan beberapa tahap pemeriksaan:

  1. Memastikan bahwa karakter tersebut adalah huruf, baik huruf kecil (a–z) maupun huruf besar (A–Z).
  2. Mengecek apakah karakter tersebut termasuk vokal, yaitu a, i, u, e, o beserta versi huruf besarnya.
  3. Menentukan kategori karakter, dengan hasil sebagai berikut:
     - Huruf dan termasuk vokal → “vokal”
     - Huruf tetapi bukan vokal → “konsonan”
     - Tidak termasuk huruf → “bukan huruf”

Proses pengelompokan ini dilakukan melalui percabangan logika sehingga hasil dapat ditampilkan secara langsung.

- **Kesimpulan**

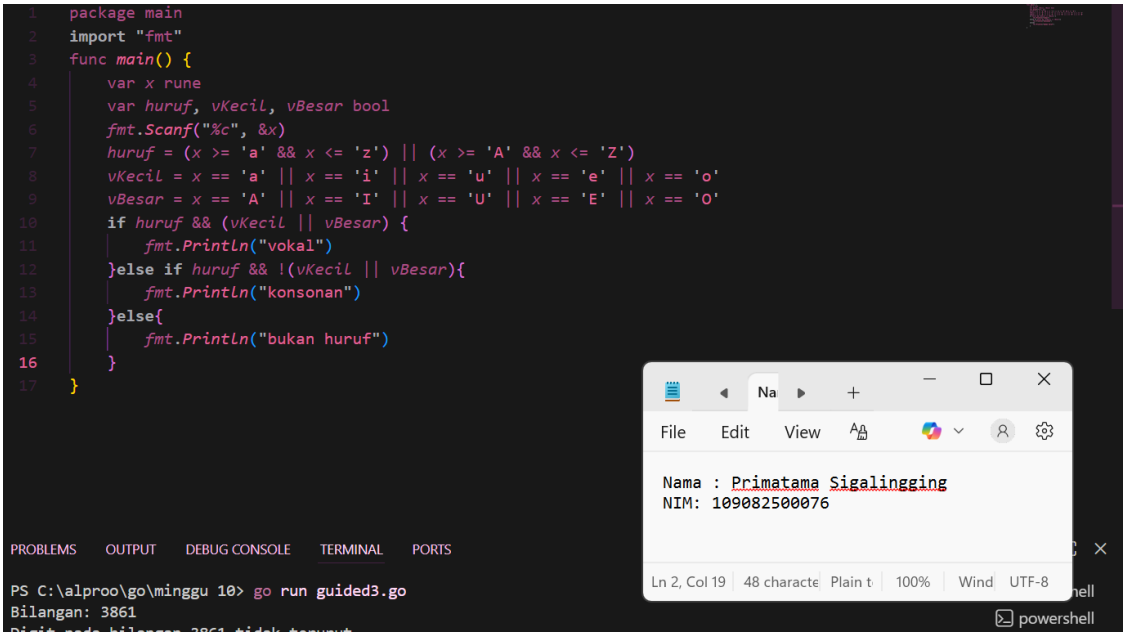
  Program ini menunjukkan bagaimana suatu karakter dapat dianalisis menggunakan logika kondisi. Pemahaman tentang proses ini sangat penting karena pemeriksaan karakter sering muncul dalam validasi input, manipulasi teks, maupun pemrograman tingkat lanjut.

3. Guided 3

Source Code

```
package main
import "fmt"
func main() {
    var bilangan, d1, d2, d3, d4 int
    var teks string
    fmt.Print("Bilangan: ")
    fmt.Scan(&bilangan)
    d4 = bilangan % 10
    d3 = (bilangan % 100) / 10
    d2 = (bilangan % 1000) / 10
    d1 = bilangan / 1000
    if d1 < d2 && d2 < d3 && d3 < d4 {
        teks = "terurut membesar"
    }else if d1 > d2 && d2 > d3 && d3 > d4{
        teks = "terurut mengecil"
    }else{
        teks = "tidak terurut"
    }
    fmt.Println("Digit pada bilangan", bilangan, teks)
}
```

Screenshoot program



Deskripsi program

- **Tujuan**

Program ini dibuat untuk mengetahui apakah digit-digit dari sebuah bilangan empat angka tersusun secara berurutan membesar, berurutan mengecil, atau tidak berurutan sama sekali. Tujuan utama program adalah membantu pengguna memahami bagaimana cara memecah sebuah angka menjadi digit-digit terpisah dan bagaimana melakukan pengecekan urutan menggunakan logika kondisi.

- **Proses**

Saat program dijalankan, pengguna diminta memasukkan sebuah bilangan empat digit. Setelah angka dimasukkan, program akan memecah bilangan tersebut menjadi empat digit, yaitu:
d1 = digit ribuan
d2 = digit ratusan
d3 = digit puluhan
d4 = digit satuan
Pemecahan dilakukan menggunakan operasi modulus (%) dan pembagian.

Setelah digit-digit dipisahkan, program melakukan pengecekan urutan:
1. Jika d1 < d2 < d3 < d4, maka digit-digit dianggap
→ "terurut membesar"
2. Jika d1 > d2 > d3 > d4, maka digit-digit dianggap
→ "terurut mengecil"
3. Jika kedua kondisi tidak terpenuhi, maka hasilnya adalah
→ "tidak terurut"

Setelah proses pemeriksaan selesai, program menampilkan hasil analisis tersebut ke layar.

- **Kesimpulan**
  Program ini memberikan pemahaman tentang cara memecah angka menjadi digit-digit serta bagaimana menerapkan logika percabangan untuk menentukan pola urutan. Konsep ini sangat berguna dalam banyak aplikasi pemrograman, seperti validasi angka, analisis data numerik, dan pengolahan digit.

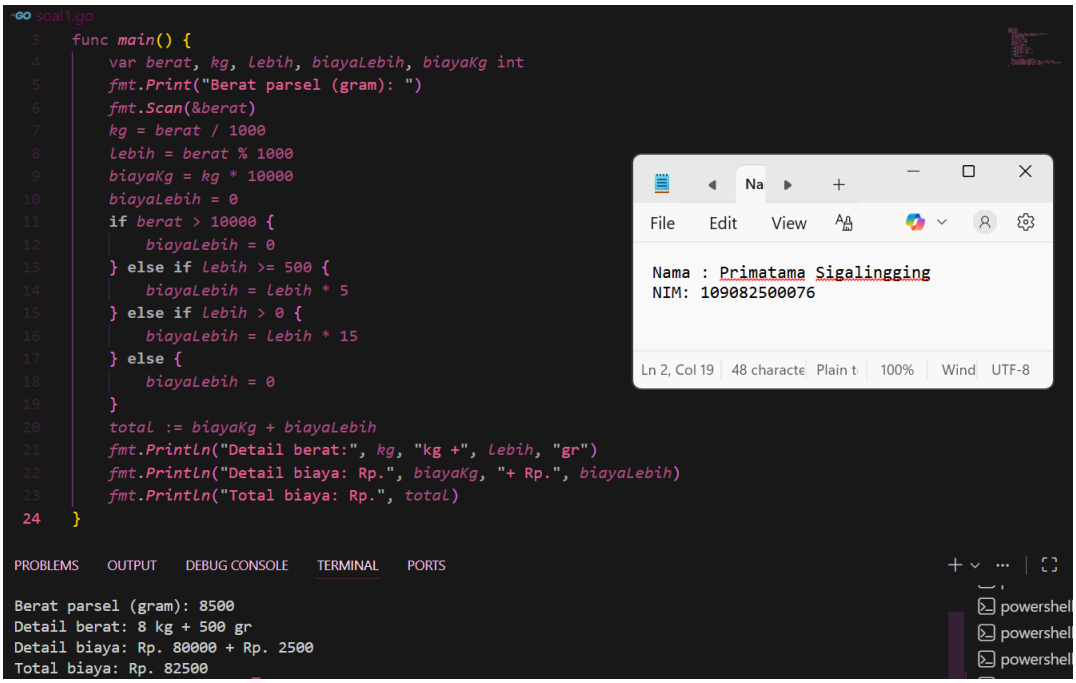
TUGAS

1. **Tugas 1**

   Source code

```
package main
import "fmt"
func main() {
    var berat, kg, lebih, biayaLebih, biayaKg int
    fmt.Print("Berat parsel (gram): ")
    fmt.Scan(&berat)
    kg = berat / 1000
    lebih = berat % 1000
    biayaKg = kg * 10000
    biayaLebih = 0
    if berat > 10000 {
        biayaLebih = 0
    } else if lebih >= 500 {
        biayaLebih = lebih * 5
    } else if lebih > 0 {
        biayaLebih = lebih * 15
    } else {
        biayaLebih = 0
    }
    total := biayaKg + biayaLebih
    fmt.Println("Detail berat:", kg, "kg +", lebih, "gr")
    fmt.Println("Detail biaya: Rp.", biayaKg, "+ Rp.", biayaLebih)
    fmt.Println("Total biaya: Rp.", total)
}
```

Screenshoot program



Deskripsi program

- **Tujuan**

Program ini dibuat untuk menghitung biaya pengiriman parsel berdasarkan berat yang diberikan dalam satuan gram. Selain menentukan total biaya, program juga menunjukkan cara memecah berat menjadi kilogram dan sisa gram, serta bagaimana menghitung tarif tambahan melalui logika kondisi. Program ini membantu pengguna memahami alur perhitungan biaya pengiriman secara transparan dan sistematis.

- **Proses**

Saat program dijalankan, pengguna diminta memasukkan berat parsel. Setelah menerima input, program memisahkan berat tersebut menjadi dua bagian: jumlah kilogram dan sisa gram. Bagian kilogram digunakan untuk menghitung biaya dasar, karena setiap kilogram dikenakan tarif tetap sebesar sepuluh ribu rupiah.

Sisa gram kemudian dianalisis untuk menentukan apakah ada biaya tambahan. Program memeriksa apakah berat parsel melebihi batas tertentu, apakah sisa gram berada pada kategori tertentu, atau apakah tidak ada sisa sama sekali. Dari pengecekan tersebut, program menentukan tarif tambahan yang sesuai, baik menggunakan tarif yang lebih rendah maupun tarif yang lebih tinggi per gram.

Setelah menghitung biaya dasar dan biaya tambahan, kedua nilai tersebut dijumlahkan untuk memperoleh total biaya pengiriman. Program kemudian menampilkan rincian berat yang sudah dipisah, rincian perhitungan biaya, dan total keseluruhan yang harus dibayar.

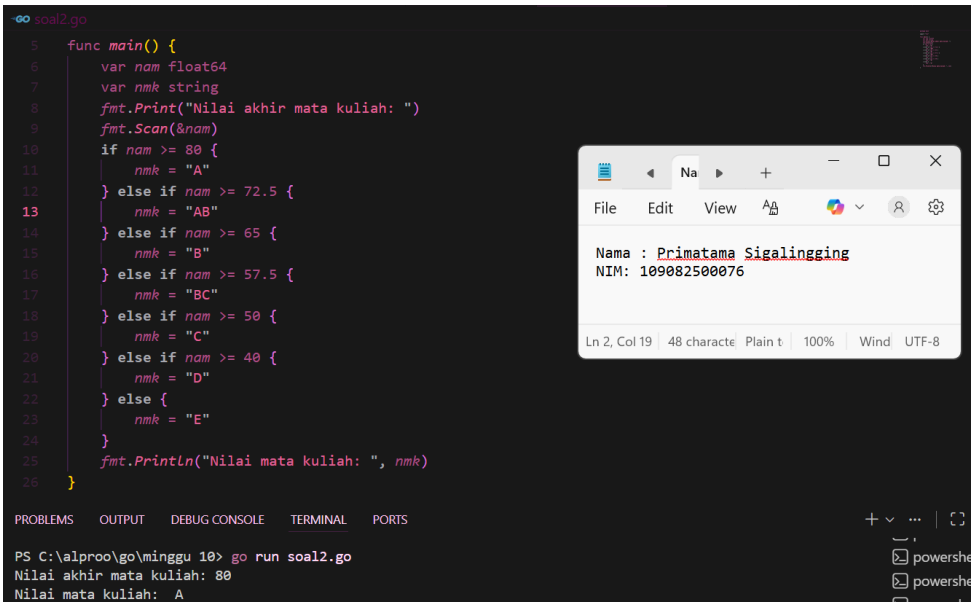
- **Kesimpulan**
  Program ini memberikan gambaran jelas tentang cara melakukan konversi satuan berat, memanfaatkan operasi modulus, dan menerapkan logika percabangan untuk menentukan biaya tambahan berdasarkan kondisi tertentu. Konsep-konsep ini sangat penting dalam berbagai aplikasi perhitungan, terutama yang berkaitan dengan tarif, logistik, dan pemrosesan data numerik.

2. Tugas 2

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



a) Ketika nilai 80.1 dimasukkan, program akan memberikan output A karena kondisi **nam >= 80** langsung terpenuhi. Namun, hasil ini tidak sesuai dengan spesifikasi soal, sebab aturan penilaian pada program tidak mengikuti rentang nilai yang diminta dalam tugas.

b) Kesalahan utama program terletak pada penentuan interval nilai yang tidak sesuai, penggunaan kondisi yang saling tumpang tindih, serta alur penilaian yang tidak cocok dengan contoh keluaran yang diharapkan. Seharusnya program memiliki batas nilai yang jelas, teratur, dan konsisten agar setiap nilai jatuh pada kategori huruf yang tepat.

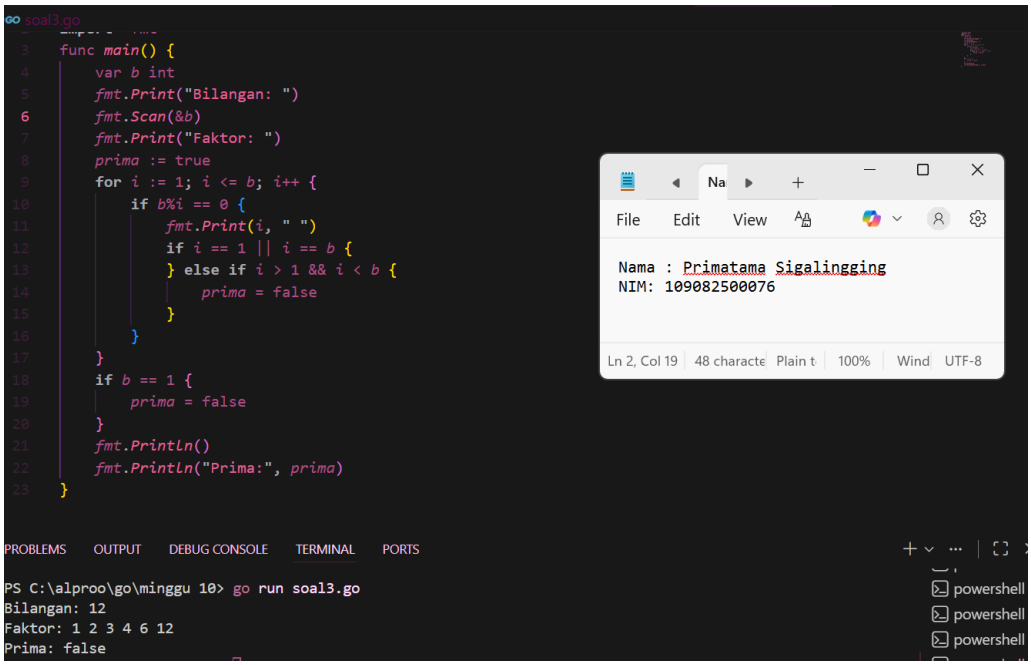
c) Untuk memperbaiki program, interval nilai harus disusun ulang sehingga mengikuti contoh keluaran yang benar (93.5 → A, 70.6 → B, 49.5 → D). Setelah rentang nilai diperbaiki, program dapat menghasilkan nilai huruf yang tepat dan konsisten sesuai ketentuan soal.

3. Tugas 3

Source code

```
package main
import "fmt"
func main() {
    var b int
    fmt.Print("Bilangan: ")
    fmt.Scan(&b)
    fmt.Print("Faktor: ")
    prima := true
    for i := 1; i <= b; i++ {
        if b%i == 0 {
            fmt.Print(i, " ")
            if i == 1 || i == b {
            } else if i > 1 && i < b {
                prima = false
            }
        }
    }
    if b == 1 {
        prima = false
    }
    fmt.Println()
    fmt.Println("Prima:", prima)
}
```

Screenshoot program



Deskripsi program

- **Tujuan**

  Program ini dibuat untuk menampilkan seluruh faktor dari suatu bilangan serta menentukan apakah bilangan tersebut termasuk bilangan prima atau bukan. Selain itu, program ini membantu pengguna memahami cara melakukan pengecekan faktor dengan perulangan dan bagaimana memanfaatkan logika kondisi untuk mengidentifikasi sifat keprimaan suatu angka.

- **Proses**

  Ketika program dijalankan, pengguna diminta memasukkan sebuah bilangan bulat. Program kemudian melakukan pemeriksaan terhadap setiap angka dari 1 hingga bilangan tersebut untuk mengetahui mana saja yang menjadi faktor. Setiap angka yang dapat membagi bilangan utama tanpa sisa akan ditampilkan sebagai faktor.

  Selama proses pengecekan, program juga memastikan apakah bilangan hanya memiliki dua faktor, yaitu 1 dan dirinya sendiri. Jika ditemukan faktor lain di antara keduanya, maka bilangan tersebut tidak dianggap prima. Program juga menangani kasus khusus ketika bilangan bernilai 1, karena 1 bukan termasuk bilangan prima.

  Setelah seluruh pemeriksaan selesai, program menampilkan daftar faktor dan memberikan informasi apakah bilangan tersebut prima atau bukan.

- **Kesimpulan**

  Program ini memberikan pemahaman mengenai cara mencari faktor suatu bilangan melalui perulangan serta bagaimana menentukan sifat bilangan prima menggunakan logika kondisi. Teknik ini penting dipahami karena sering digunakan dalam pemrograman yang berkaitan dengan matematika, analisis angka, dan validasi numerik.