

**LAPORAN PRAKTIKUM ALGORITMA**

**DAN PEMROGRAMAN 1**

**MODUL 10**

**ELSE IF**



**Disusun oleh:**

**M MAHDAN ARGYA SYARIF**

**109082500059**

**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 umur int
    var kk bool
    fmt.Print("Masukkan umur: ")
    fmt.Scan(&umur)
    fmt.Print("punya KK?: ")
    fmt.Scan(&kk)

    if umur >= 17 && kk {
        fmt.Print("bisa membuat KTP")
    } else {
        fmt.Print("belum bisa membuat KTP")
    }
}
```

#### Screenshot program

The screenshot shows a GoLang IDE interface. On the left, the code editor displays the provided Go program. On the right, the terminal window shows the execution of the program and its output.

**Code Editor:**

```
1 package main
2
3 import "fmt"
4
5 func main() {    main redeclared in this block (see details)
6     var umur int
7     var kk bool
8     fmt.Print("Masukkan umur: ")
9     fmt.Scan(&umur)
10    fmt.Print("punya KK?: ")
11    fmt.Scan(&kk)
12
13    if umur ≥ 17 && kk {
14        fmt.Print("bisa membuat KTP")
15    } else {
16        fmt.Print("belum bisa membuat KTP")
17    }
18 }
```

**Terminal Output:**

- PS C:\GoLang> go run guided1.go
- Masukkan umur: 17
- punya KK?: true
- bisa membuat KTP
- PS C:\GoLang> go run guided1.go
- Masukkan umur: 20
- punya KK?: false
- belum bisa membuat KTP
- PS C:\GoLang> go run guided1.go
- Masukkan umur: 15
- punya KK?: true
- belum bisa membuat KTP
- PS C:\GoLang> []

## Deskripsi program

Program ini menggunakan algoritma *if-else* untuk memvalidasi kelayakan pembuatan KTP, di mana syarat mutlak agar bernilai *true* adalah pengguna harus berusia minimal 17 tahun dan memiliki Kartu Keluarga (KK). Sebaliknya, jika kondisi tersebut tidak terpenuhi secara utuh—seperti pengguna yang sudah cukup umur namun tidak memiliki KK, atau pengguna yang belum cukup umur—maka algoritma akan menganggap input tidak memenuhi syarat dan menghasilkan output *false*.

## 2. Guided 2

### Source Code

```
package main

import "fmt"

func main() {

    var x rune

    var huruf, vKecil, vBesar bool

    fmt.Println("masukkan huruf: ")

    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")
    }
}
```

## Screenshot program

The screenshot shows a GoLang IDE interface. On the left, the code editor displays a Go program that classifies characters as vowels, consonants, or non-letters. On the right, the terminal window shows the execution of the program and its output for various inputs.

```
1 package main
2 import "fmt"
3 func main() {    main redeclared in this block
4     var x rune
5     var huruf, vKecil, vBesar bool
6     fmt.Println("masukkan huruf: ")
7     fmt.Scanf("%c", &x)
8
9     huruf = (x >= 'a' && x <= 'z') || (x >= 'A' && x <= 'Z')
10    vKecil = x == 'a' || x == 'i' || x == 'u' || x == 'e' || x == 'o'
11    vBesar = x == 'A' || x == 'I' || x == 'U' || x == 'E' || x == 'O'
12
13    if huruf && (vKecil || vBesar) {
14        fmt.Println("vokal")
15    }else if huruf && !(vKecil || vBesar){
16        fmt.Println("konsonan")
17    }else{
18        fmt.Println("bukan huruf")
19    }
20 }
```

Terminal Output:

- PS C:\GoLang> go run guided2.go  
masukkan huruf: A  
vokal
- PS C:\GoLang> go run guided2.go  
masukkan huruf: f  
konsonan
- PS C:\GoLang> go run guided2.go  
masukkan huruf: 1  
bukan huruf
- PS C:\GoLang> go run guided2.go  
masukkan huruf: \$  
bukan huruf

## Deskripsi program

Program ini bertujuan mengklasifikasikan input karakter menjadi huruf vokal, konsonan, atau bukan huruf dengan memanfaatkan tipe data rune untuk representasi Unicode. Alur program dimulai dengan mendeklarasikan variabel boolean bantu (huruf, vKecil, vBesar) untuk memetakan rentang alfabet dan karakter vokal baik kapital maupun kecil, yang kemudian dievaluasi menggunakan struktur kendali *if-else*. Berdasarkan logika tersebut, program akan mencetak output "vokal" jika input sesuai dengan himpunan huruf A, I, U, E, O, menghasilkan "konsonan" untuk huruf alfabet lainnya, atau menampilkan "bukan huruf" jika input berupa simbol atau angka di luar definisi alfabet.

### 3. Guided 3

#### Source Code

```
package main

import "fmt"

func main() {

    var bilangan, d1, d2, d3, d4 int

    var teks string
```

```

fmt.Println("Bilangan: ")
fmt.Scan(&bilangan)

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

The screenshot shows a code editor interface with a dark theme. On the left, there is a list of errors and warnings. In the center, the code for `guided3.go` is displayed. On the right, a terminal window shows the execution of the program and its output.

**Code Editor Errors:**

- Line 3: `func main()` is highlighted in red with the message `main redeclared in this block`.
- Line 14: `if d1 < d2 && d2 < d3 && d3 < d4 {` is highlighted in red with the message `else if d1 > d2 && d2 > d3 && d3 > d4{` is also highlighted in red.
- Line 18: `} else{` is highlighted in red with the message `teks = "tidak terurut"` is also highlighted in red.

**Terminal Output:**

```

PS C:\GoLang> go run guided3.go
Bilangan: 2489
Digit pada bilangan 2489 terurut membesar
PS C:\GoLang> go run guided3.go
Bilangan: 3841
Digit pada bilangan 3841 tidak terurut
PS C:\GoLang> go run guided3.go
Bilangan: 9651
Digit pada bilangan 9651 terurut mengecil
PS C:\GoLang>

```

### **Deskripsi program**

Program ini berfungsi untuk mengidentifikasi pola urutan pada input bilangan empat digit, menentukan apakah digit-digitnya terurut membesar, mengecil, atau tidak beraturan. Proses teknisnya dimulai dengan memecah bilangan menjadi empat variabel digit terpisah ( $d_1, d_2, d_3, d_4$ ) menggunakan operasi matematika pembagian dan modulus untuk mengekstrak nilai ribuan hingga satuan. Selanjutnya, algoritma menerapkan struktur kendali kondisional untuk membandingkan relasi antar digit; jika memenuhi kondisi  $d_1 < d_2 < d_3 < d_4$  maka statusnya terurut membesar, jika  $d_1 > d_2 > d_3 > d_4$  maka terurut mengecil, dan jika tidak memenuhi kedua pola tersebut, maka dikategorikan sebagai tidak terurut.

## TUGAS

### 1. Tugas 1

#### Source code

```
package main

import "fmt"

func main() {

    var berat, kg, g, harga, biaya, total int

    fmt.Print("Berat parsel(gram): ")

    fmt.Scan(&berat)

    kg = berat / 1000

    g = berat % 1000

    harga = kg * 10000

    if berat > 10000 {

        biaya = g * 5

        total = harga

    } else if g >= 500 {

        biaya = g * 5

        total = harga + biaya

    } else if g < 500 {

        biaya = g * 15

        total = harga + biaya

    }

}
```

```

        fmt.Printf("detail berat: %d kg + %d gr\n", kg, g)

        fmt.Printf("detail biaya: Rp. %d + Rp. %d\n", harga,
biaya)

        fmt.Printf("total biaya: Rp. %d", total)

    }

```

## Screenshoot program

```

soal1.go:1:1: main/main.go:1: package main
soal1.go:1:1: main/main.go:2: import "fmt"
soal1.go:1:1: main/main.go:4: func main() { main redeclared in this block
soal1.go:1:1: main/main.go:5:     var berat, kg, g, harga, biaya, total int
soal1.go:1:1: main/main.go:6:     fmt.Println("Berat parsel(gram): ")
soal1.go:1:1: main/main.go:7:     fmt.Scan(&berat)
soal1.go:1:1: main/main.go:8: 
soal1.go:1:1: main/main.go:9:     kg = berat / 1000
soal1.go:1:1: main/main.go:10:     g = berat % 1000
soal1.go:1:1: main/main.go:11:     harga = kg * 10000
soal1.go:1:1: main/main.go:12: 
soal1.go:1:1: main/main.go:13:     if berat > 10000 {
soal1.go:1:1: main/main.go:14:         biaya = g * 5
soal1.go:1:1: main/main.go:15:         total = harga
soal1.go:1:1: main/main.go:16:     } else if g >= 500 {
soal1.go:1:1: main/main.go:17:         biaya = g * 5
soal1.go:1:1: main/main.go:18:         total = harga + biaya
soal1.go:1:1: main/main.go:19:     } else if g < 500 {
soal1.go:1:1: main/main.go:20:         biaya = g * 15
soal1.go:1:1: main/main.go:21:         total = harga + biaya
soal1.go:1:1: main/main.go:22:     }
soal1.go:1:1: main/main.go:23: 
soal1.go:1:1: main/main.go:24:     fmt.Printf("detail berat: %d kg + %d gr\n", kg, g)
soal1.go:1:1: main/main.go:25:     fmt.Printf("detail biaya: Rp. %d + Rp. %d\n", harga, biaya)
soal1.go:1:1: main/main.go:26:     fmt.Printf("total biaya: Rp. %d", total)
soal1.go:1:1: main/main.go:27: }

PROBLEMS 12 OUTPUT TERMINAL DEBUG CONSOLE PORTS

```

- PS C:\GoLang> go run soal1.go
Berat parsel(gram): 8500
detail berat: 8 kg + 500 gr
detail biaya: Rp. 80000 + Rp. 2500
total biaya: Rp. 82500
- PS C:\GoLang> go run soal1.go
Berat parsel(gram): 9250
detail berat: 9 kg + 250 gr
detail biaya: Rp. 90000 + Rp. 3750
total biaya: Rp. 93750
- PS C:\GoLang> go run soal1.go
Berat parsel(gram): 11750
detail berat: 11 kg + 750 gr
detail biaya: Rp. 110000 + Rp. 3750
total biaya: Rp. 110000

## Deskripsi program

Program ini dirancang untuk menghitung biaya pengiriman parsel menggunakan tipe data *integer* untuk seluruh variabel operasional dan memanfaatkan pustaka fmt untuk penanganan input-output. Logika bisnisnya menetapkan tarif dasar Rp10.000 per kilogram, dengan penerapan aturan kondisional khusus untuk sisa gram: biaya tambahan digratiskan jika total berat melebihi 10 kg, namun dikenakan tarif bertingkat (Rp5 per gram jika  $\geq 500$ g, atau Rp15 per gram jika  $< 500$ g) untuk berat standar. Seluruh hasil kalkulasi, mulai dari rincian berat hingga total harga akhir, kemudian ditampilkan secara terstruktur dan terperinci kepada pengguna.

## 2. Tugas 2

### Source code

```
package main

import "fmt"

func main() {
    var nam float64
    var nmk string

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

    if nam > 80 {
        nam = "A"
    }

    if nam > 72.5 {
        nam = "AB"
    }

    if nam > 65 {
        nam = "B"
    }

    if nam > 57.5 {
        nam = "BC"
    }

    if nam > 50 {
        nam = "C"
    }
}
```

```

    }

    if nam > 40 {

        nam = "D"

    } else if nam <= 40 {

        nam = "E"

    }

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

}

```

### Screenshot program

The screenshot shows a GoLang IDE interface. On the left, there is a code editor with the following Go code:

```

1 package main
2 import "fmt"
3 func main() { main redeclared in this block
4 var nam float64
5 var nmk string
6
7 fmt.Println("Nilai akhir mata kuliah: ")
8 fmt.Scan(&nam)
9
10 if nam > 80 {
11     nam = "A" cannot use "A" (untyped string constant) as float64 value in assignment
12 }
13 if nam > 72.5 {
14     nam = "AB" cannot use "AB" (untyped string constant) as float64 value in assignment
15 }
16 if nam > 65 {
17     nam = "B" cannot use "B" (untyped string constant) as float64 value in assignment
18 }
19 if nam > 57.5 {
20     nam = "BC" cannot use "BC" (untyped string constant) as float64 value in assignment
21 }
22 if nam > 50 {
23     nam = "C" cannot use "C" (untyped string constant) as float64 value in assignment
24 }
25 if nam > 40 {
26     nam = "D" cannot use "D" (untyped string constant) as float64 value in assignment
27 } else if nam <= 40 {
28     nam = "E" cannot use "E" (untyped string constant) as float64 value in assignment
29 }
30 fmt.Println("Nilai mata kuliah: ", nmk)
31 }

```

The code editor highlights several syntax errors with red underlines and error messages. On the right, the terminal window shows the command PS C:\GoLang> go run soal2.go followed by a list of errors:

```

PS C:\GoLang> go run soal2.go
# command-line-arguments
.\soal2.go:11:8: cannot use "A" (untyped string constant) as float64 value in assignment
.\soal2.go:14:8: cannot use "AB" (untyped string constant) as float64 value in assignment
.\soal2.go:17:8: cannot use "B" (untyped string constant) as float64 value in assignment
.\soal2.go:20:8: cannot use "BC" (untyped string constant) as float64 value in assignment
.\soal2.go:23:8: cannot use "C" (untyped string constant) as float64 value in assignment
.\soal2.go:26:8: cannot use "D" (untyped string constant) as float64 value in assignment
.\soal2.go:28:8: cannot use "E" (untyped string constant) as float64 value in assignment
PS C:\GoLang> []

```

### • SOAL A

Jika **nam** diberikan adalah 80.1, apa keluaran dari program tersebut?

Apakah eksekusi program tersebut sesuai spesifikasi soal?

**Jawab:**

Program tersebut tidak dapat dieksekusi karena terdapat kesalahan fatal pada prosedur pengkondisian, di mana terjadi ketidaksesuaian tipe data

antara variabel nam (bertipe *float64*) dan nilai yang diproses. Kesalahan terletak pada logika yang mencoba membandingkan atau menetapkan nilai *string* "A" langsung pada variabel numerik nam, padahal seharusnya kondisi mengevaluasi nilai nam (misalnya  $\$ \geq 80$ ) untuk kemudian menyimpan predikat huruf mutunya ke dalam variabel nmk (bertipe *string*), sehingga konflik tipe data ini menyebabkan program mengalami *error*.

- **SOAL B**

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

**Jawab:**

Program ini memiliki dua kesalahan fundamental, yaitu ketidaksesuaian tipe data saat penetapan variabel dan cacat logika pada struktur kendali. Masalah sintaksis terjadi karena pencampuran tipe data *float64* dan *string* yang seharusnya dipisahkan antara kondisi dan penugasan nilai, namun kesalahan logika yang lebih fatal terletak pada penggunaan rangkaian if independen (bukan if-else if). Akibatnya, input nilai 80.1 yang seharusnya menghasilkan gred "A" justru tertimpa menjadi "D", karena program terus mengevaluasi kondisi hingga baris terakhir tanpa mengunci hasil yang benar sebelumnya, membuat alur penyeleksian kondisi menjadi tidak akurat.

- **SOAL C**

Perbaiki program tersebut! Ujilah dengan masukan: 93.5; 70.6; dan 49.5.

Seharusnya keluaran yang diperoleh adalah 'A', 'B', dan 'D'.

**Jawab:**

```
package main

import "fmt"

func main() {
    var nam float64
    var nmk string

    fmt.Println("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 if nam <= 40 {  
    nmk = "E"  
}  
  
fmt.Println("Nilai mata kuliah: ", nmk)  
}
```

## Screenshot program & bukti jawaban soal c

The screenshot shows a GoLang IDE interface. On the left, the code editor displays a Go program named 'soal2.go' with syntax highlighting. The code defines a function 'main()' that reads a float64 value from the user, then uses a series of if-else statements to determine a letter grade ('A', 'B', 'C', 'D', or 'E') based on the input value. The right side of the interface shows a terminal window with the command 'go run soal2.go' and its output. The output shows three runs of the program with inputs 93.5, 70.6, and 49.5, resulting in outputs 'A', 'B', and 'D' respectively.

```
1 package main
2 import "fmt"
3 func main() {
4     var nam float64
5     var nmk string
6
7     fmt.Println("Nilai akhir mata kuliah: ")
8     fmt.Scan(&nam)
9
10    if nam > 80 {
11        nmk = "A"
12    } else if nam > 72.5 {
13        nmk = "AB"
14    } else if nam > 65 {
15        nmk = "B"
16    } else if nam > 57.5 {
17        nmk = "BC"
18    } else if nam > 50 {
19        nmk = "C"
20    } else if nam > 40 {
21        nmk = "D"
22    } else if nam <= 40 {
23        nmk = "E"
24    }
25    fmt.Println("Nilai mata kuliah: ", nmk)
26 }
```

```
PS C:\GoLang> go run soal2.go
.\soal2.go:14:8: cannot use "AB" (untyped string constant) as float64 value in assignment
.\soal2.go:17:8: cannot use "B" (untyped string constant) as float64 value in assignment
.\soal2.go:20:8: cannot use "BC" (untyped string constant) as float64 value in assignment
.\soal2.go:23:8: cannot use "C" (untyped string constant) as float64 value in assignment
.\soal2.go:26:8: cannot use "D" (untyped string constant) as float64 value in assignment
.\soal2.go:28:8: cannot use "E" (untyped string constant) as float64 value in assignment
PS C:\GoLang> go run soal2.go
Nilai akhir mata Kuliah: 93.5
Nilai mata Kuliah: A
PS C:\GoLang> go run soal2.go
Nilai akhir mata Kuliah: 70.6
Nilai mata Kuliah: B
PS C:\GoLang> go run soal2.go
Nilai akhir mata Kuliah: 49.5
Nilai mata Kuliah: D
PS C:\GoLang> ]
```

### Deskripsi program

Program ini merupakan sistem konversi nilai akademik berbasis bahasa Go yang mengubah input angka presisi ganda (float64) menjadi indeks huruf mutu (string) melalui struktur percabangan if-else if berjenjang. Menggunakan paket fmt untuk manajemen input-output, algoritma mengevaluasi rentang nilai secara menurun dari batas atas hingga terendah untuk menentukan klasifikasi yang tepat. Fungsionalitas program ini telah divalidasi melalui uji coba input angka 93.5, 70.6, dan 49.5, yang secara akurat menghasilkan output klasifikasi huruf mutu berturut-turut A, B, dan D.

### 3. Tugas 3A

#### Source code

```
package main

import "fmt"

func main() {
    var n, i int
    fmt.Print("masukkan bilangan: ")
    fmt.Scan(&n)

    fmt.Print("faktor: ")
    for i = 1; i <= n; i++ {
        if n % i == 0 {
            fmt.Print(" ", i)
        }
    }
}
```

```
    }  
}
```

## Screenshot program

The screenshot shows a GoLang IDE interface. On the left, the code editor displays a Go program:

```
1 package main  
2 import "fmt"  
3  
4 func main() { main redeclared in this block  
5     var n, i int  
6     fmt.Print("masukkan bilangan: ")  
7     fmt.Scan(&n)  
8  
9     fmt.Print("faktor: ")  
10    for i = 1; i <= n; i++ {  
11        if n % i == 0 {  
12            fmt.Print(" ", i)  
13        }  
14    }  
15 }
```

On the right, the terminal window shows the execution of the program:

```
Nama: M Mahdan Argya Syarif  
NIM: 109082500059  
Kelas: IF-13-07  
Ln 3, Col 16 | 61 character | Plain t | 110% | Win
```

Below the terminal, the terminal tab shows command-line history:

- PS C:\GoLang> go run soal3\_1.go  
masukkan bilangan: 12  
faktor: 1 2 3 4 6 12
- PS C:\GoLang> go run soal3\_1.go  
masukkan bilangan: 7  
faktor: 1 7
- PS C:\GoLang> []

## Deskripsi program

Program ini dirancang untuk mengidentifikasi dan menampilkan faktor-faktor pembentuk suatu bilangan bulat. Dengan menggunakan tipe data integer pada variabel input dan iterator, program mengelola interaksi pengguna melalui paket fmt untuk menangkap nilai dan menampilkan hasil. Mekanisme intinya bertumpu pada struktur perulangan for yang menelusuri angka dari 1 hingga batas nilai input, dikombinasikan dengan logika percabangan if yang memvalidasi kelipatan menggunakan operasi modulus; jika sisa bagi bernilai nol, maka angka tersebut dikonfirmasi dan dicetak sebagai faktor yang valid.

## 4. Tugas 3B

### Source code

```
package main  
import "fmt"  
  
func main() {  
    var n, i, faktor int  
    var prima bool  
    fmt.Print("masukkan bilangan: ")
```

```

fmt.Scan(&n)
faktor = 0

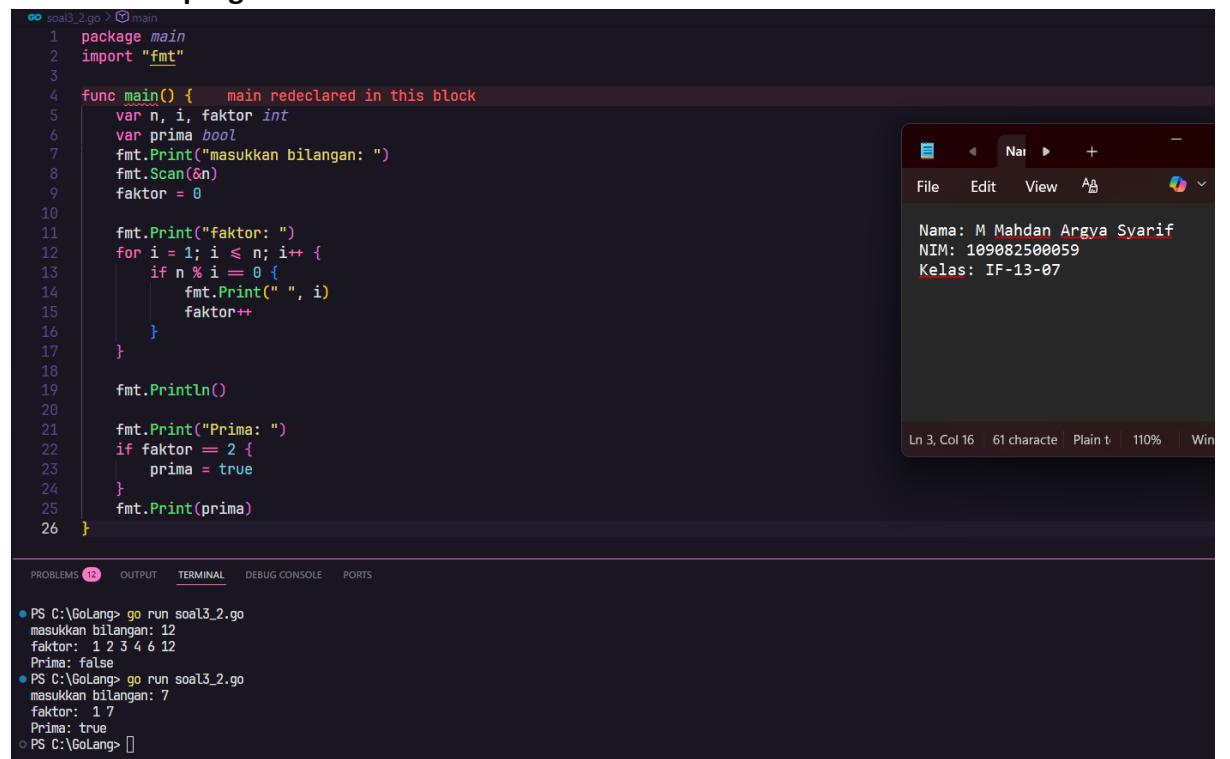
fmt.Println("faktor: ")
for i = 1; i <= n; i++ {
    if n % i == 0 {
        fmt.Println(" ", i)
        faktor++
    }
}

fmt.Println()

fmt.Println("Prima: ")
if faktor == 2 {
    prima = true
}
fmt.Println(prima)
}

```

## Screenshot program



```

soal3_2.go > main
1 package main
2 import "fmt"
3
4 func main() {    main redeclared in this block
5     var n, i, faktor int
6     var prima bool
7     fmt.Println("masukkan bilangan: ")
8     fmt.Scan(&n)
9     faktor = 0
10
11    fmt.Println("faktor: ")
12    for i = 1; i <= n; i++ {
13        if n % i == 0 {
14            fmt.Println(" ", i)
15            faktor++
16        }
17    }
18
19    fmt.Println()
20
21    fmt.Println("Prima: ")
22    if faktor == 2 {
23        prima = true
24    }
25    fmt.Println(prima)
26 }

```

PROBLEMS 12 OUTPUT TERMINAL DEBUG CONSOLE PORTS

- PS C:\GoLang> go run soal3\_2.go
 masukkan bilangan: 12
 faktor: 1 2 3 4 6 12
 Prima: false
- PS C:\GoLang> go run soal3\_2.go
 masukkan bilangan: 7
 faktor: 1 7
 Prima: true
 PS C:\GoLang> []

Nama: M Mahdan Argya Syarif  
NIM: 109082500059  
Kelas: IF-13-07

### **Deskripsi program**

Program ini menerapkan logika matematika untuk mengidentifikasi bilangan prima dengan cara memvalidasi jumlah faktor pembaginya. Menggunakan tipe data *integer* untuk pemrosesan angka dan pencacahan, serta *boolean* untuk menyimpan status validitas, algoritma bekerja dengan menelusuri seluruh pembagi potensial melalui perulangan *for*. Penentuan status prima dilakukan di akhir proses menggunakan struktur kondisional, di mana jika total faktor yang ditemukan tepat berjumlah dua, variabel indikator akan diset menjadi *true* untuk mengonfirmasi bilangan tersebut sebagai prima.