

**LAPORAN PRAKTIKUM ALGORITMA
DAN PEMROGRAMAN 1**

MODUL 10

ELSE IF



Disusun oleh:

MOHAMAD ERLANGGA ZEIN

109082500020

S1IF-13-07

Asisten Praktikum

Adithana dharmaputra

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")

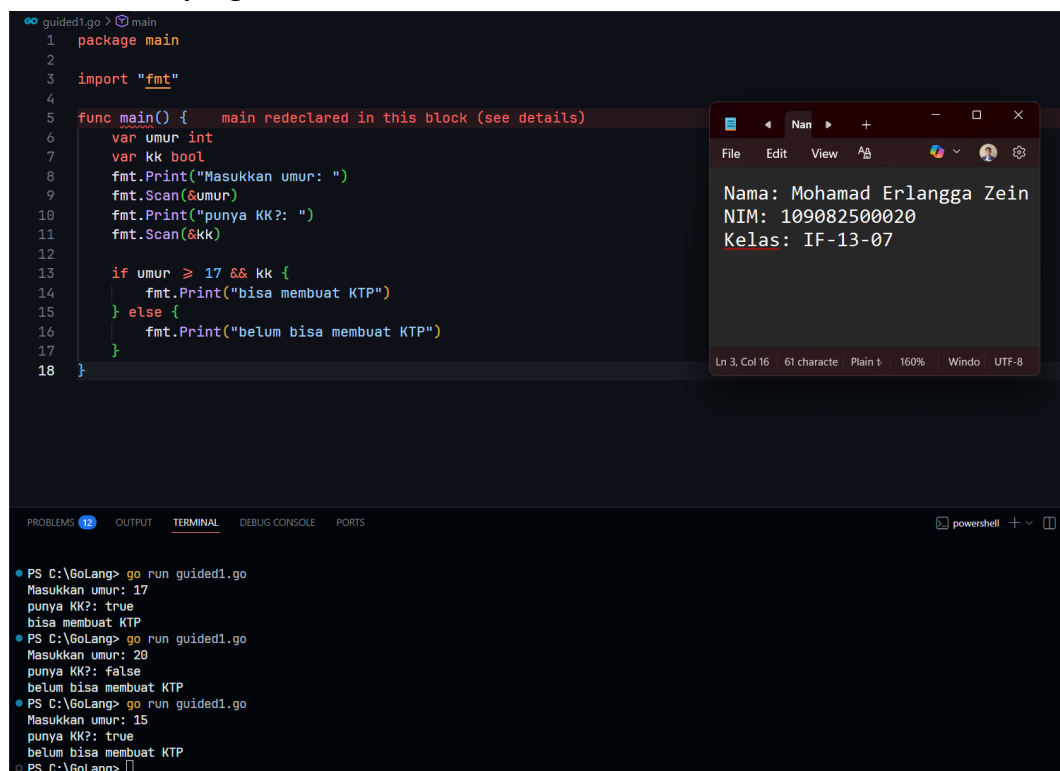
    } else {

        fmt.Print("belum bisa membuat KTP")

    }

}
```

Screenshoot program



The screenshot shows a Go program running in a terminal window. The program prompts the user for their age and whether they have a KTP (ID card). The user enters 17 for age and true for having a KTP, and the program outputs "bisa membuat KTP".

```
guided1.go > main
1 package main
2
3 import "fmt"
4
5 func main() {
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>
```

Text box content:

```
Nama: Mohamad Erlangga Zein
NIM: 109082500020
Kelas: IF-13-07
```

Deskripsi program

Pada program tersebut dijelaskan bahwa ketika user berumur 17 tahun dan memiliki Kartu Keluarga(KK) maka ia bisa membuat KTP, namun apabila user sudah berumur lebih dari 17 tahun namun belum mempunyai KK, maka ia belum bisa membuat KTP, dan bila user berumur kurang dari 17 tahun dan belum mempunyai KK ia juga belum bisa membuat KTP, program ini dibuat dengan algoritma if-else dimana ketika suatu algoritma bernilai true maka output yang dihasilkan true, sedangkan apabila input yang dimasukkan/algoritmanya belum memenuhi maka output yang dihasilkan false.

2. Guided 2

Source Code

```
package main

import "fmt"

func main() {
    var x rune
    var huruf, vKecil, vBesar bool

    fmt.Print("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")
    }
}
```

Screenshoot program

The screenshot shows a Go program in a text editor and its execution in a terminal. The program, named `guided2.go`, defines a `main` function that prompts the user to enter a character. It then checks if the character is a vowel (vokal), consonant (konsonan), or not a letter (bukan huruf) based on specific Unicode ranges. The terminal shows three test cases: 'A' is a vowel, 'f' is a consonant, and '1' is not a letter.

```
1 package main
2 import "fmt"
3 func main() {
4     var x rune
5     var huruf, vKecil, vBesar bool
6     fmt.Print("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
PS C:\GoLang>
```

Deskripsi program

Program ini mengeluarkan output berupa huruf vokal, konsonan, dan bukan huruf, dengan mendeklarasikan variable dengan tipe data rune, yang dimana tipe data rune ini mengkonversikan suatu bilangan/number berubah menjadi Unicode, yang dimana Unicode ini dapat berupa huruf alphabet, sehingga dalam deklarasinya, menggunakan tanda petik 1 saja ('...') kemudian deklarasikan Unicode tersebut ke dalam pengkondisian if else if dan else, yang pertama sebelum memasukkan ke dalam pengkondisian, baiknya dibuat dulu variable lain untuk mendeklarasi nilai dari apa yang ingin di buat perbandingan, dimana disini variable yang akan dibuat menggunakan tipe data Boolean, di dalam program tersebut variable tersebut antara lain variable huruf, vKecil, vBesar, dimana setiap variable berbeda beda perbandingannya, dimana pada variable huruf digunakan sebagai perbandingan dari huruf alphabet, dari alphabet yang besar, dimulai dari `x >= 'A' && x <= 'Z' || x >= 'a' && x <= 'z'` kemudian pada variable vKecil dan vBesar dideklarasikan huruf vokal(A,I,U,E,O) dengan huruf besar dan kecil, kemudian setelah itu di buat pengkondisian seperti pada program di gambar tersebut, sehingga outputnya adalah bila user memasukkan huruf A,I,U,E,O dengan huruf besar maupun huruf kecil, maka output an programnya adalah vokal, ketika huruf selain A,I,U,E,O maka output nya konsonan, dan bila symbol, maka output nya bukan huruf.

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) / 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

```

1 package main
2 import "fmt"
3 func main() {
4     var bilangan, d1, d2, d3, d4 int
5     var teks string
6     fmt.Print("Bilangan: ")
7     fmt.Scan(&bilangan)
8
9     d4 = bilangan % 10
10    d3 = (bilangan % 100) / 10
11    d2 = (bilangan % 1000) / 100
12    d1 = bilangan / 1000
13
14    if d1 < d2 && d2 < d3 && d3 < d4 {
15        teks = "terurut membesar"
16    }else if d1 > d2 && d2 > d3 && d3 > d4{
17        teks = "terurut mengecil"
18    }else{
19        teks = "tidak terurut"
20    }
21    fmt.Println("Digit pada bilangan", bilangan, teks)
22 }

```

```

PS C:\GoLang> go run guided3.go
Bilangan: 2489
Digit pada bilangan 2489 terurut membesar
PS C:\GoLang> go run guided3.go
Bilangan: 3861
Digit pada bilangan 3861 tidak terurut
PS C:\GoLang> go run guided3.go
Bilangan: 9651
Digit pada bilangan 9651 terurut mengecil
PS C:\GoLang>

```

Nama: Mohamad Erlangga Zein
 NIM: 109082500020
 Kelas: IF-13-07

Deskripsi program

Program ini menghasilkan output dimana ketika user menginputkan angka dengan 4 digit, kemudian algoritmanya akan memproses apakah bilangan yang di inputkan oleh user itu terurut membesar, terurut mengecil, atau tidak terurut, sehingga dalam pemrogramannya, perlu mendeklarasikan operasi matematika terhadap d1, d2, d3, d4 dengan membaginya dengan membaginya dengan 1000, lalu moduluskan dengan 1000, 100, dan 10 dengan urutan ribuan, ratusan, puluhan, satuan, sehingga dalam pengkondisiannya adalah $d1 < d2 \ \&\& \ d2 < d3 \ \&\& \ d3 < d4$ yang mendeklarasikan terurut membesar, lalu $\text{else if } d1 > d2 \ \&\& \ d2 > d3 \ \&\& \ d3 > d4$ yang mendeklarasikan terurut mengecil lalu ketika digitnya acak maka tidak terurut.

TUGAS

1. Tugas 1

Source code

```
package main

import "fmt"

func main() {
    var parsel, kilogram, gram, harga, biaya, total int
    fmt.Print("Berat parsel(gram): ")
    fmt.Scan(&parsel)

    kilogram = parsel / 1000
    gram = parsel % 1000
    harga = kilogram * 10000

    if parsel > 10000 {
        biaya = gram * 5
        total = harga
    } else if gram >= 500 {
        biaya = gram * 5
        total = harga + biaya
    } else if gram < 500 {
        biaya = gram * 15
        total = harga + biaya
    }
```

```

    }

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

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

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

}

```

Screenshoot program

The screenshot shows a Go program in VS Code and its execution output in a terminal window. The program calculates the total cost based on weight (kilogram and gram) and applies different rates for different weight ranges.

```

1 package main
2 import "fmt"
3
4 func main() {
5     var parsek, kilogram, gram, harga, biaya, total int
6     fmt.Print("Berat parsek(gram): ")
7     fmt.Scan(&parsek)
8
9     kilogram = parsek / 1000
10    gram = parsek % 1000
11    harga = kilogram * 10000
12
13    if parsek > 10000 {
14        biaya = gram * 5
15        total = harga
16    } else if gram >= 500 {
17        biaya = gram * 5
18        total = harga + biaya
19    } else if gram < 500 {
20        biaya = gram * 15
21        total = harga + biaya
22    }
23
24    fmt.Printf("detail berat: %d kg + %d gr\n", kilogram, gram)
25    fmt.Printf("detail biaya: Rp. %d + Rp. %d\n", harga, biaya)
26    fmt.Printf("total biaya: Rp. %d", total)
27 }

```

The terminal output shows the program's execution for three different inputs:

```

PS C:\GoLang> go run soal1.go
detail biaya: Rp. 80000 + Rp. 2500
total biaya: Rp. 82500
PS C:\GoLang> go run soal1.go
Berat parsek(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 parsek(gram): 11750
detail berat: 11 kg + 750 gr
detail biaya: Rp. 110000 + Rp. 3750
total biaya: Rp. 110000
PS C:\GoLang>

```

Deskripsi program

Dalam program ini, seluruh variabel operasional yaitu *parsek*, *kilogram*, *gram*, *harga*, *biaya*, dan *total* dideklarasikan menggunakan tipe data *int* (integer) karena program ini hanya memproses bilangan bulat untuk satuan berat dan nilai rupiah tanpa desimal. Fungsi utama *main()* berperan sebagai titik awal eksekusi logika, yang di dalamnya memanfaatkan pustaka standar *fmt* untuk menangani input dan output. Secara spesifik, fungsi *fmt.Print* digunakan untuk menampilkan teks *prompt* tanpa baris baru, *fmt.Scan* berfungsi menangkap input angka dari pengguna ke alamat memori variabel *parsek*, dan *fmt.Printf* digunakan untuk menampilkan hasil perhitungan akhir dengan format khusus (menggunakan *%d*) agar data angka dapat disisipkan rapi ke dalam kalimat output. Dan program ini juga menggunakan logika penentuan harga di mana setiap 1 kilogram dikenakan biaya Rp10.000, dan menerapkan aturan kondisional untuk sisa gram: jika total berat melebihi 10 kg maka

sisanya gram tidak dikenakan biaya tambahan, namun jika berat standar, sisanya gram dikenakan biaya Rp5 per gram (jika ≥ 500 g) atau Rp15 per gram (jika < 500 g), yang kemudian hasil akhirnya ditampilkan secara terperinci meliputi detail berat, kalkulasi biaya, dan total harga.

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 {

        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)

}

```

Screenshoot program

The screenshot shows a Go program in an IDE with several compilation errors. The program defines a `main` function that declares `nam` as `float64` and `nmk` as `string`. It uses `fmt.Scan(&nam)` to read input and then checks `nam` against several conditions, assigning values to `nmk`. The errors indicate that string literals like "A", "AB", "B", "BC", "C", "D", and "E" cannot be used as `float64` values in assignments.

On the right, a terminal window shows the program's output for the input `80.1`:

```

Nama: Mohamad Erlangga Zein
NIM: 109082500020
Kelas: IF-13-07

```

At the bottom, a PowerShell terminal shows the command to run the program and the resulting error messages:

```

PS C:\GoLang> go run soal2.go
# command-line-arguments
.\soal2.go:10:8: 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:18:8: cannot use "B" (untyped string constant) as float64 value in assignment
.\soal2.go:22:8: cannot use "BC" (untyped string constant) as float64 value in assignment
.\soal2.go:26:8: cannot use "C" (untyped string constant) as float64 value in assignment
.\soal2.go:30:8: cannot use "D" (untyped string constant) as float64 value in assignment
.\soal2.go:32:8: 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?

Jawaban:

Jawabannya program tidak dapat dieksekusi, karena prosedur dari program tersebut salah, dimana harusnya pada pengkondisian if then nya harusnya if nam = > 80 { nmk = "A" } bukan pada program sebelumnya yang dimana mendeklarasikannya nam = "A", karena tipe data pada nam dan nmk berbeda, nam dengan tipe data float64, sedangkan nmk tipe data string, sehingga programnya tidak dapat tereksekusi/error.

• SOAL B:

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

Jawaban:

Kesalahan dalam program tersebut adalah, kesalahan dalam mendeklarasikan variable, dimana seharusnya pada pengkondisiannya seharusnya menggunakan if nam > 80 { nmk = "A" }, karena perbedaan tipe data, dimana nam adalah float64 dan nmk adalah string, sehingga mengalami error, dan setelah diubah juga output an dari 80.1 belum benar, hasil output an nya adalah D, seharusnya adalah A, karena dalam program tersebut hanya mendeklarasikan if then, bukan if – else if, sehingga dalam pemrosesannya hanya memproses pengkondisian yang memiliki aksi, serta dalam pengkondisiannya juga mengapa output an nya D karena pada 2 pengkondisian if then terakhir itu menggunakan if – else if, sehingga dalam pemrosesannya, ketika user menginput 80.1 maka hasilnya akan D, bukan A.

• SOAL C:

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

Jawaban:

```
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 if nam <= 40 {
nmk = "E"
    }

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

```

Deskripsi program

Program ini berfungsi sebagai sistem konversi nilai akademik sederhana yang mengklasifikasikan nilai angka menjadi indeks huruf mutu (A hingga E) menggunakan bahasa Go. Variabel input `nam` dideklarasikan dengan tipe data `float64` agar dapat memproses nilai desimal presisi ganda, sedangkan variabel output `nmk` bertipe `string` digunakan untuk menampung hasil konversi berupa teks huruf. Di dalam blok fungsi `main()`, interaksi input-output dikelola oleh paket `fmt`—di mana `fmt.Print` menampilkan *prompt*, `fmt.Scan` menangkap input ke alamat memori variabel `nam`, dan `fmt.Println` mencetak hasil akhir—sementara logika penentuan nilai dijalankan melalui struktur percabangan `if-else if` berjenjang yang mengevaluasi rentang nilai secara menurun, mulai dari batas atas (> 80 untuk A) hingga batas bawah (≤ 40 untuk E).

Sehingga dalam soal ini, ketika diuji memasukkan input 93.5, 70.6, dan 49.5 hasilnya A, B, D

BUKTI:

The screenshot displays a Go IDE with the following source code and terminal output:

```

1 package main
2 import "fmt"
3 func main() {
4     var nam float64
5     var nmk string
6
7     fmt.Print("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 }

```

The terminal output shows the results of running the program with different inputs:

```

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>

```

3. Tugas 3A

Source code

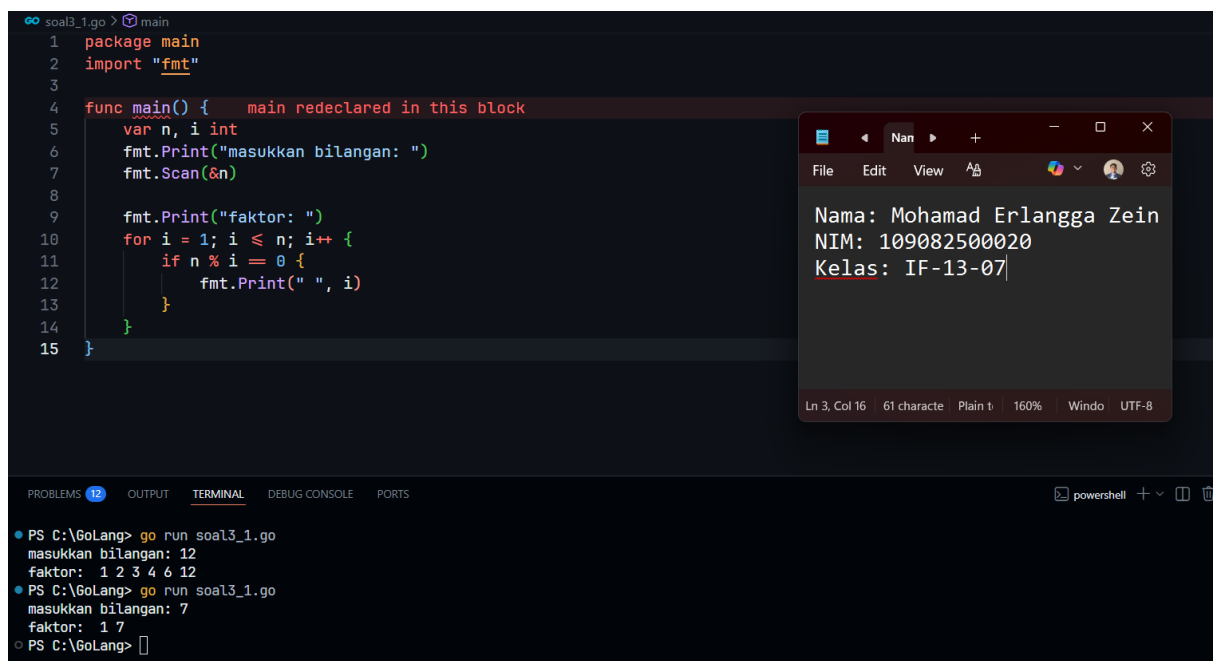
```
package main

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)
        }
    }
}
```

Screenshoot program



The screenshot displays a Go IDE with the source code for a program that calculates the factors of a given number. The code is as follows:

```
1 package main
2 import "fmt"
3
4 func main() {
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 }
```

The IDE also shows the output of the program in the terminal. The first run shows the input 12 and the output factors 1 2 3 4 6 12. The second run shows the input 7 and the output factor 1 7.

```
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 merupakan aplikasi berbasis bahasa Go yang dirancang untuk mencari dan menampilkan faktor-faktor pembentuk dari sebuah bilangan bulat yang diinputkan oleh pengguna. Variabel utama yang digunakan adalah `n` untuk menampung nilai input dan `i` sebagai iterator dalam perulangan, di mana keduanya menggunakan tipe data `int` (integer) karena operasi matematika modulus memerlukan bilangan bulat. Fungsi `main()` berperan sebagai blok eksekusi utama yang memanggil fungsi-fungsi dari paket `fmt`: `fmt.Print` untuk menampilkan teks prompt dan hasil faktor secara menyamping, serta `fmt.Scan` untuk menangkap input pengguna ke variabel `n`. Algoritmanya bekerja dengan menggunakan struktur kontrol perulangan `for` yang mengevaluasi angka dari 1 hingga `n`, disertai logika percabangan `if` yang memeriksa apakah sisa bagi (%) antara `n` dan `i` bernilai 0; jika kondisinya terpenuhi, maka angka `i` tersebut ditampilkan 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.Print("faktor: ")
    for i = 1; i <= n; i++ {
        if n % i == 0 {
            fmt.Print(" ", i)
            faktor++
        }
    }

    fmt.Println()
```

```

        fmt.Print("Prima: ")

        if faktor == 2 {
            prima = true
        }

        fmt.Print(prima)
    }
}

```

Screenshoot program

The screenshot shows a Go program in VS Code and its execution output in the terminal. The program is a function `main()` that takes an integer `n` and returns a boolean `prima`. It uses a loop to find factors of `n` and a conditional statement to determine if `n` is a prime number.

```

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

```

The terminal output shows the program being run twice. In the first run, the input is 12, and the output is "faktor: 1 2 3 4 6 12" and "Prima: false". In the second run, the input is 7, and the output is "faktor: 1 7" and "Prima: true".

Deskripsi program

Program ini merupakan pengembangan logika matematika untuk menentukan apakah sebuah bilangan bulat termasuk bilangan prima atau bukan berdasarkan jumlah faktor pembaginya. Variabel `n`, `i`, dan `faktor` dideklarasikan dengan tipe data `int` (integer) untuk menangani input angka, iterasi perulangan, dan pencacahan jumlah faktor, sementara variabel `prima` menggunakan tipe data `bool` (boolean) untuk menyimpan status logis `true` atau `false`. Fungsi utama `main()` mengontrol alur eksekusi yang memanfaatkan paket `fmt` seperti `fmt.Scan` untuk input dan `fmt.Print`/`fmt.Println` untuk output di mana algoritma bekerja dengan mencari faktor-faktor bilangan tersebut melalui perulangan `for`, lalu menggunakan struktur `if` untuk memeriksa apakah jumlah faktor (`faktor`) tepat berjumlah 2; jika ya, maka variabel `prima` diubah menjadi `true` yang menandakan bilangan tersebut adalah prima.