

**LAPORAN PRAKTIKUM ALGORITMA
DAN PEMROGRAMAN 1**

MODUL 10

ELSE IF



Disusun oleh:

FERDINAND AXEL VALERIAN

109082500154

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

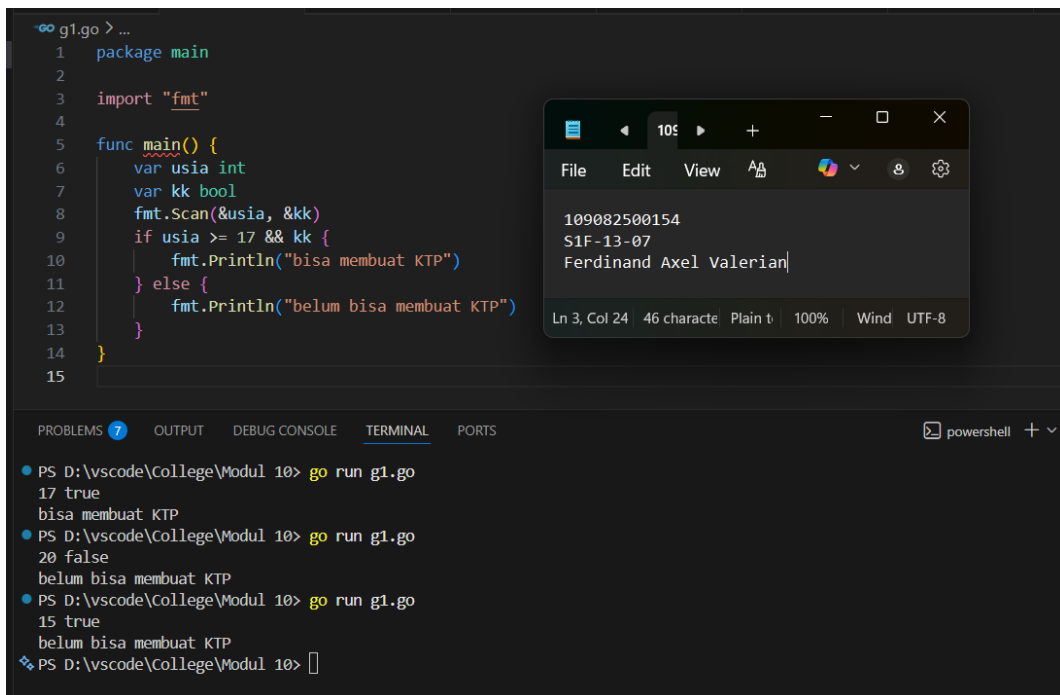
import "fmt"

func main() {
    var usia int
    var kk bool

    fmt.Scan(&usia, &kk)

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

Screenshoot program



```
g1.go > ...
1 package main
2
3 import "fmt"
4
5 func main() {
6     var usia int
7     var kk bool
8     fmt.Scan(&usia, &kk)
9     if usia >= 17 && kk {
10         fmt.Println("bisa membuat KTP")
11     } else {
12         fmt.Println("belum bisa membuat KTP")
13     }
14 }
15
```

109082500154
S1F-13-07
Ferdinand Axel Valerian

Ln 3, Col 24 46 character Plain text 100% Window UTF-8

PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell +

- PS D:\vscode\College\Modul 10> go run g1.go
17 true
bisa membuat KTP
- PS D:\vscode\College\Modul 10> go run g1.go
20 false
belum bisa membuat KTP
- PS D:\vscode\College\Modul 10> go run g1.go
15 true
belum bisa membuat KTP

PS D:\vscode\College\Modul 10>

Deskripsi program

Program ini dibuat untuk mengecek apakah seseorang sudah memenuhi syarat untuk membuat KTP. Data yang dipakai ada dua, yaitu usia dalam bentuk angka, serta informasi apakah orang tersebut memiliki Kartu Keluarga (KK) atau tidak, yang dituliskan dalam bentuk boolean.

Pada bagian awal, program membaca kedua input itu menggunakan `fmt.Scan`. Setelah datanya masuk, program langsung melakukan pengecekan dengan kondisi `if usia >= 17 && kk`. Logikanya sederhana: seseorang baru dianggap memenuhi syarat membuat KTP kalau usianya minimal 17 tahun dan sudah memiliki KK.

Kalau kedua syarat tersebut terpenuhi, program akan menampilkan teks *"bisa membuat KTP"*. Tapi kalau salah satu syarat saja belum terpenuhi—entah usianya belum cukup atau belum punya KK—program akan memberikan keluaran *"belum bisa membuat KTP"*.

Secara keseluruhan, program ini berfungsi sebagai pengecekan cepat untuk menentukan kelayakan pembuatan KTP berdasarkan aturan dasar yang berlaku.

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')

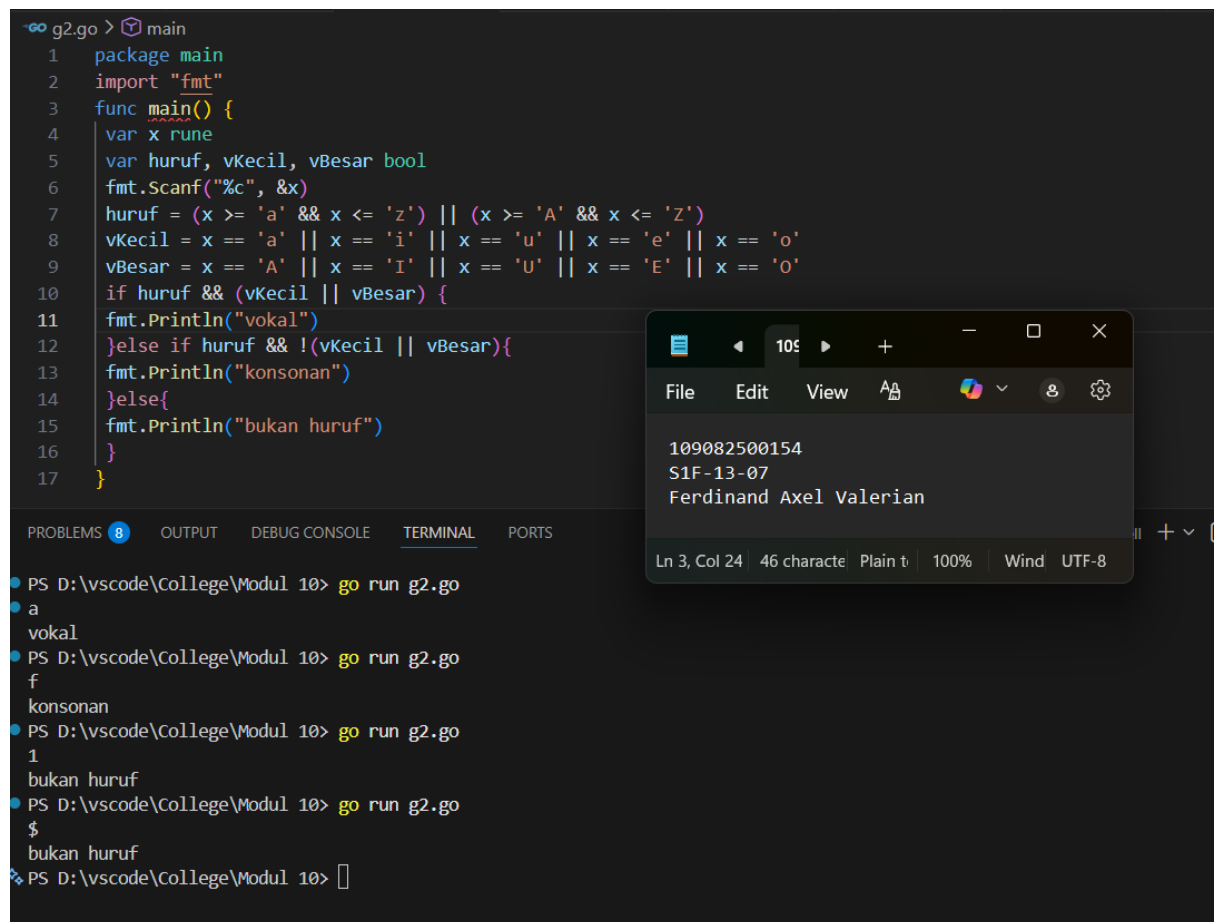
    vKecil = x == 'a' || x == 'i' || x == 'u' || x == 'e' || x == 'o'

    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



```
g2.go > main
1 package main
2 import "fmt"
3 func main() {
4     var x rune
5     var huruf, vKecil, vBesar bool
6     fmt.Scanf("%c", &x)
7     huruf = (x >= 'a' && x <= 'z') || (x >= 'A' && x <= 'Z')
8     vKecil = x == 'a' || x == 'i' || x == 'u' || x == 'e' || x == 'o'
9     vBesar = x == 'A' || x == 'I' || x == 'U' || x == 'E' || x == 'O'
10    if huruf && (vKecil || vBesar) {
11        fmt.Println("vokal")
12    } else if huruf && !(vKecil || vBesar) {
13        fmt.Println("konsonan")
14    } else {
15        fmt.Println("bukan huruf")
16    }
17 }
```

PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\vscode\College\Modul 10> go run g2.go
a
vokal
PS D:\vscode\College\Modul 10> go run g2.go
f
konsonan
PS D:\vscode\College\Modul 10> go run g2.go
1
bukan huruf
PS D:\vscode\College\Modul 10> go run g2.go
\$
bukan huruf
PS D:\vscode\College\Modul 10>

Deskripsi program

Program ini dibuat untuk menentukan apakah sebuah karakter yang dimasukkan pengguna merupakan huruf vokal, huruf konsonan, atau bukan huruf sama sekali. Proses penentuan dilakukan dengan memeriksa apakah input termasuk dalam rentang alfabet, kemudian mengecek apakah huruf tersebut merupakan vokal dalam bentuk huruf kecil maupun huruf besar.

Program pertama-tama membaca satu karakter dari pengguna. Setelah itu, program melakukan pengecekan menggunakan dua kondisi utama:

1. Validasi huruf – memastikan karakter berada pada rentang alfabet a–z atau A–Z.
2. Pengecekan vokal – menentukan apakah karakter tersebut termasuk vokal (a, i, u, e, o atau versi kapitalnya).

Jika karakter valid dan termasuk vokal, program menampilkan keluaran “vokal”. Jika karakter valid tetapi bukan vokal, maka hasilnya “konsonan”. Sementara itu, jika karakter yang diberikan tidak berada dalam kelompok huruf alfabet, program akan mengeluarkan pesan “bukan huruf”.

Dengan struktur ini, program dapat memberikan klasifikasi sederhana namun jelas terhadap setiap karakter yang dimasukkan oleh pengguna

3. Guided 3

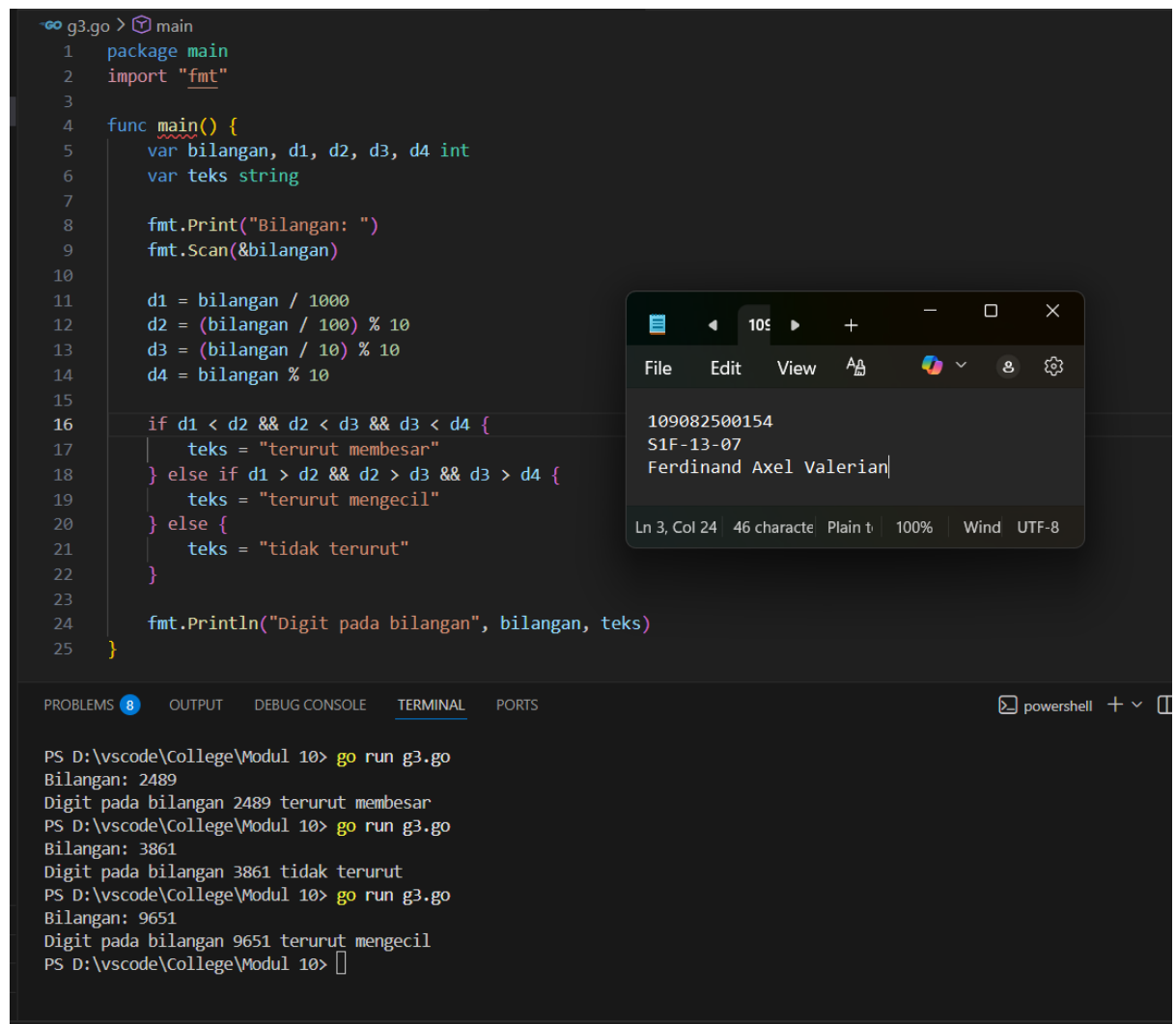
Source Code

```
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



The screenshot shows a Go program in a VS Code editor. The program reads a four-digit number and checks if the digits are in ascending, descending, or non-monotonic order. The terminal shows three test cases: 2489 (ascending), 3861 (non-monotonic), and 9651 (descending).

```
g3.go > main
1 package main
2 import "fmt"
3
4 func main() {
5     var bilangan, d1, d2, d3, d4 int
6     var teks string
7
8     fmt.Print("Bilangan: ")
9     fmt.Scan(&bilangan)
10
11     d1 = bilangan / 1000
12     d2 = (bilangan / 100) % 10
13     d3 = (bilangan / 10) % 10
14     d4 = bilangan % 10
15
16     if d1 < d2 && d2 < d3 && d3 < d4 {
17         teks = "terurut membesar"
18     } else if d1 > d2 && d2 > d3 && d3 > d4 {
19         teks = "terurut mengecil"
20     } else {
21         teks = "tidak terurut"
22     }
23
24     fmt.Println("Digit pada bilangan", bilangan, teks)
25 }
```

Terminal Output:

```
PS D:\vscode\College\Modul 10> go run g3.go
Bilangan: 2489
Digit pada bilangan 2489 terurut membesar
PS D:\vscode\College\Modul 10> go run g3.go
Bilangan: 3861
Digit pada bilangan 3861 tidak terurut
PS D:\vscode\College\Modul 10> go run g3.go
Bilangan: 9651
Digit pada bilangan 9651 terurut mengecil
PS D:\vscode\College\Modul 10>
```

Deskripsi program

Program ini dibuat untuk menganalisis urutan digit dalam sebuah bilangan empat digit. Fungsinya sederhana: menentukan apakah digit-digitnya berurutan naik, turun, atau tidak berurutan sama sekali.

Cara kerjanya dimulai dengan membaca input bilangan dari pengguna. Kemudian program memecah bilangan tersebut menjadi empat digit terpisah - digit ribuan, ratusan, puluhan, dan satuan. Proses pemecahan ini menggunakan operasi pembagian dan modulus untuk mengambil masing-masing digit.

Setelah mendapatkan keempat digit, program melakukan pengecekan kondisi. Jika digit pertama lebih kecil dari kedua, kedua lebih kecil dari ketiga, dan ketiga lebih kecil dari keempat (misalnya 2-4-8-9), maka program akan menampilkan "terurut membesar". Sebaliknya, jika digitnya semakin mengecil (seperti 9-6-5-1), maka hasilnya "terurut mengecil". Kalau polanya tidak konsisten naik atau turun, seperti 3-8-6-1, program akan mengategorikan sebagai "tidak terurut".

Pada dasarnya, program ini memberikan cara cepat untuk mengetahui pola digit dalam bilangan tanpa perlu menganalisis manual. Cocok untuk keperluan belajar matematika dasar atau validasi data numerik.

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

    biayaDasar := kg * 10000

    var biayaTambahan int

    if kg > 10 {

        biayaTambahan = 0
    } else if sisaGram >= 500 {
        biayaTambahan = sisaGram * 5
    } else {
        biayaTambahan = sisaGram * 15
    }

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

    if kg > 10 {
```

```

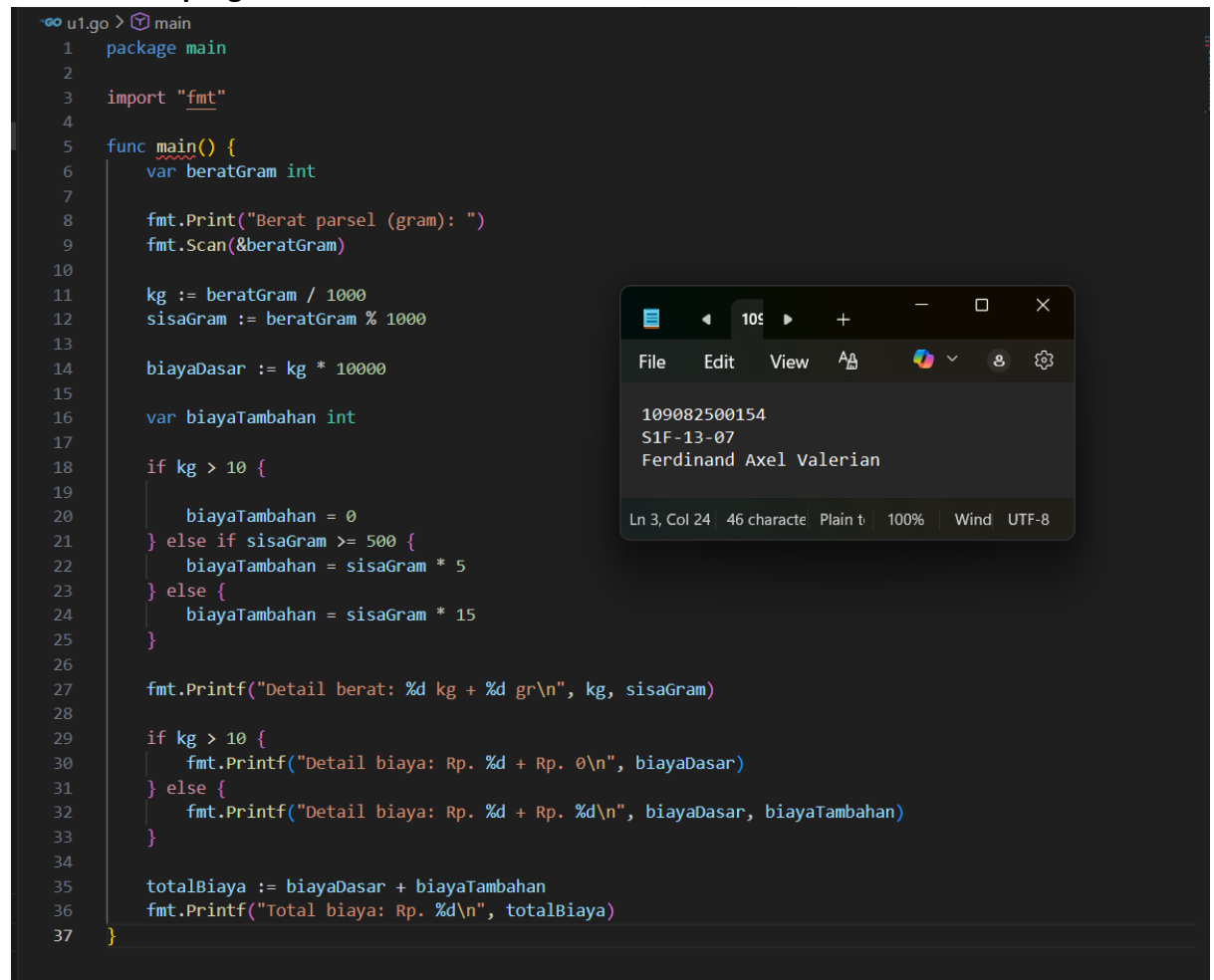
        fmt.Printf("Detail biaya: Rp. %d + Rp. 0\n", biayaDasar)
    } else {
        fmt.Printf("Detail biaya: Rp. %d + Rp. %d\n", biayaDasar,
biayaTambahan)
    }

    totalBiaya := biayaDasar + biayaTambahan

    fmt.Printf("Total biaya: Rp. %d\n", totalBiaya)
}

```

Screenshoot program



The screenshot shows a Go program in a code editor. The program calculates the total cost based on weight and remaining grams. A terminal window is open, displaying the output of the program.

```

u1.go > main
1  package main
2
3  import "fmt"
4
5  func main() {
6      var beratGram int
7
8      fmt.Print("Berat parsel (gram): ")
9      fmt.Scan(&beratGram)
10
11     kg := beratGram / 1000
12     sisaGram := beratGram % 1000
13
14     biayaDasar := kg * 10000
15
16     var biayaTambahan int
17
18     if kg > 10 {
19         biayaTambahan = 0
20     } else if sisaGram >= 500 {
21         biayaTambahan = sisaGram * 5
22     } else {
23         biayaTambahan = sisaGram * 15
24     }
25
26     fmt.Printf("Detail berat: %d kg + %d gr\n", kg, sisaGram)
27
28     if kg > 10 {
29         fmt.Printf("Detail biaya: Rp. %d + Rp. 0\n", biayaDasar)
30     } else {
31         fmt.Printf("Detail biaya: Rp. %d + Rp. %d\n", biayaDasar, biayaTambahan)
32     }
33
34     totalBiaya := biayaDasar + biayaTambahan
35     fmt.Printf("Total biaya: Rp. %d\n", totalBiaya)
36
37 }

```

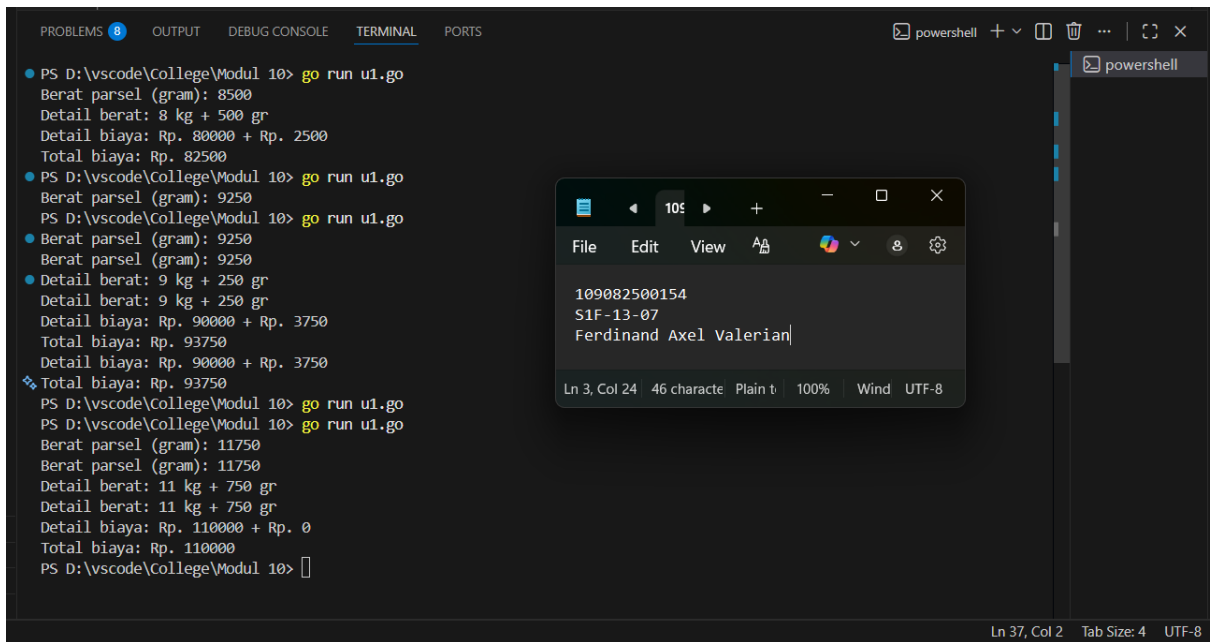
Terminal Output:

```

109082500154
S1F-13-07
Ferdinand Axel Valerian

```

Ln 3, Col 24 46 character Plain t 100% Wind UTF-8



```
PS D:\vscode\College\Modul 10> go run u1.go
Berat parcel (gram): 8500
Detail berat: 8 kg + 500 gr
Detail biaya: Rp. 80000 + Rp. 2500
Total biaya: Rp. 82500
PS D:\vscode\College\Modul 10> go run u1.go
Berat parcel (gram): 9250
PS D:\vscode\College\Modul 10> go run u1.go
Berat parcel (gram): 9250
Detail berat: 9 kg + 250 gr
Detail berat: 9 kg + 250 gr
Detail biaya: Rp. 90000 + Rp. 3750
Total biaya: Rp. 93750
Detail biaya: Rp. 90000 + Rp. 3750
PS D:\vscode\College\Modul 10> go run u1.go
PS D:\vscode\College\Modul 10> go run u1.go
Berat parcel (gram): 11750
Berat parcel (gram): 11750
Detail berat: 11 kg + 750 gr
Detail berat: 11 kg + 750 gr
Detail biaya: Rp. 110000 + Rp. 0
Total biaya: Rp. 110000
PS D:\vscode\College\Modul 10>
```

Deskripsi program

Program BiayaPos menghitung biaya pengiriman parcel berdasarkan berat barang. Sistem ini menerapkan kebijakan tarif PT POS dengan perhitungan yang mempertimbangkan berat dalam kilogram dan sisa berat dalam gram.

Program mulai dengan membaca input berat parcel dalam gram. Kemudian memisahkannya menjadi berat kilogram (dari pembagian 1000) dan sisa gram (dari modulus 1000).

Biaya dasar ditetapkan Rp 10.000 per kg. Untuk sisa gram, berlaku dua skenario: jika ≥ 500 gram, biaya Rp 5 per gram; jika < 500 gram, biaya Rp 15 per gram. Khusus untuk parcel di atas 10 kg, biaya sisa gram diabaikan.

Hasil perhitungan ditampilkan dalam format detail yang mencakup rincian berat, komponen biaya, dan total biaya akhir, memberikan transparansi perhitungan kepada pengguna.

2. Tugas 2

Source code

```
package main

import "fmt"

func main() {

    var nam float64

    var nmk string

    fmt.Print("Nilai akhir mata kuliah: ")

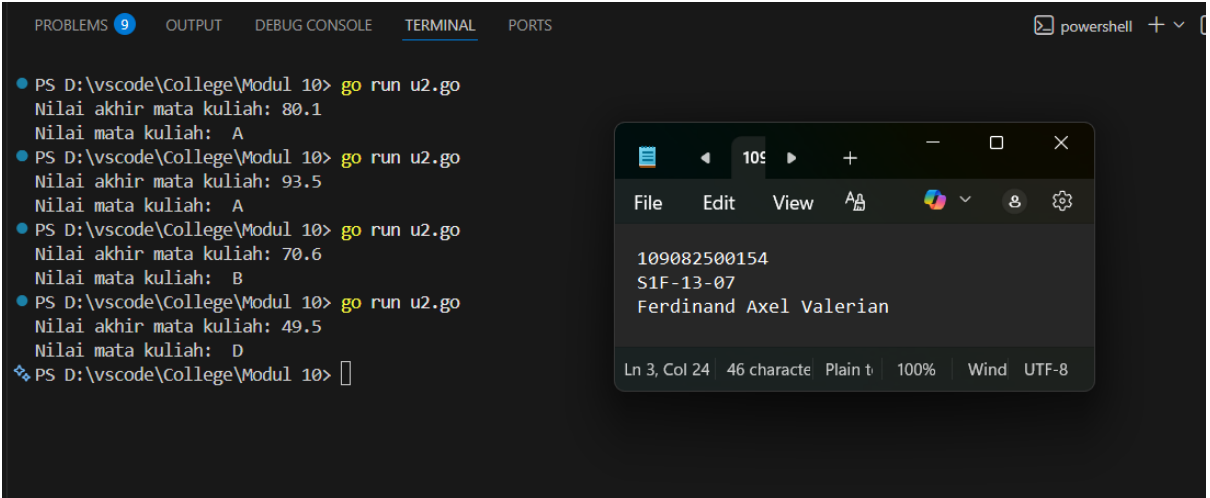
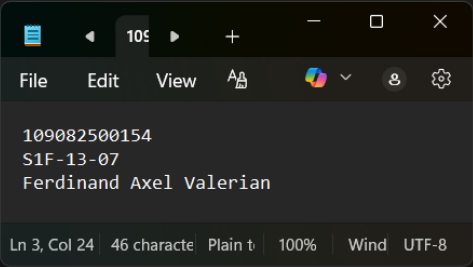
    fmt.Scan(&nam)
```

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

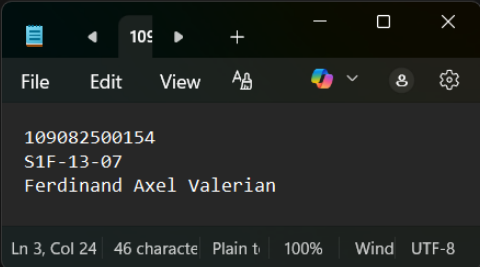
    fmt.Println("Nilai mata kuliah: ", nmk)
}
```

Screenshoot program

```
u2.go > main
1 package main
2 import "fmt"
3
4 func main() {
5     var nam float64
6     var nmk string
7
8     fmt.Print("Nilai akhir mata kuliah: ")
9     fmt.Scan(&nam)
10
11     if nam > 80 {
12         nmk = "A"
13     } else if nam > 72.5 && nam <= 80 {
14         nmk = "AB"
15     } else if nam > 65 && nam <= 72.5 {
16         nmk = "B"
17     } else if nam > 57.5 && nam <= 65 {
18         nmk = "BC"
19     } else if nam > 50 && nam <= 57.5 {
20         nmk = "C"
21     } else if nam > 40 && nam <= 50 {
22         nmk = "D"
23     } else {
24         nmk = "E"
25     }
26
27     fmt.Println("Nilai mata kuliah: ", nmk)
28 }
```



```
PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\vscode\College\Modul 10> go run u2.go
Nilai akhir mata kuliah: 80.1
Nilai mata kuliah: A
PS D:\vscode\College\Modul 10> go run u2.go
Nilai akhir mata kuliah: 93.5
Nilai mata kuliah: A
PS D:\vscode\College\Modul 10> go run u2.go
Nilai akhir mata kuliah: 70.6
Nilai mata kuliah: B
PS D:\vscode\College\Modul 10> go run u2.go
Nilai akhir mata kuliah: 49.5
Nilai mata kuliah: D
PS D:\vscode\College\Modul 10> 
```



Deskripsi program

Deskripsi Program Konversi Nilai:

Program ini mengonversi Nilai Akhir Mata Kuliah (NAM) numerik menjadi nilai huruf berdasarkan standar penilaian yang berlaku. Program menerima input nilai angka, memprosesnya melalui serangkaian kondisi berjenjang menggunakan `else if`, dan menampilkan hasil konversi dalam bentuk nilai huruf A hingga E.

a. Untuk input NAM = 80.1:

Program menampilkan output "A", yang sudah benar sesuai spesifikasi karena nilai 80.1 melebihi batas 80. Namun program asli memiliki masalah struktur logika yang berpotensi menghasilkan output salah untuk nilai lainnya.

b. Identifikasi kesalahan:

Kesalahan utama terletak pada penggunaan pernyataan `if` terpisah tanpa `else if`, menyebabkan semua kondisi yang terpenuhi dieksekusi dan nilai variabel `nmk` terus ditimpa.

c. Hasil perbaikan:

Setelah diperbaiki, program telah diuji dengan nilai 93.5 (output 'A'), 70.6 (output 'B'), dan 49.5 (output 'D'). Hasil ini sudah sesuai dengan tabel konversi nilai yang ditetapkan.

3. Tugas 3

Source code

```
package main

import "fmt"

func main() {
    var bilangan int

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

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

    jumlahFaktor := 0
    for i := 1; i <= bilangan; i++ {
        if bilangan%i == 0 {
            jumlahFaktor++
        }
    }

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

Screenshoot program

```
u3.go > main
1 package main
2
3 import "fmt"
4
5 func main() {
6     var bilangan int
7
8     fmt.Print("Bilangan: ")
9     fmt.Scan(&bilangan)
10
11     fmt.Print("Faktor: ")
12     for i := 1; i <= bilangan; i++ {
13         if bilangan%i == 0 {
14             fmt.Print(i, " ")
15         }
16     }
17     fmt.Println()
18
19     jumlahFaktor := 0
20     for i := 1; i <= bilangan; i++ {
21         if bilangan%i == 0 {
22             jumlahFaktor++
23         }
24     }
25
26     prima := jumlahFaktor == 2
27     fmt.Println("Prima:", prima)
28 }
```

109082500154
S1F-13-07
Ferdinand Axel Valerian

Ln 3, Col 24 | 46 character | Plain t | 100% | Wind | UTF-8

```
PS D:\vscode\College\Modul 10> go run u3.go
Bilangan: 12
Faktor: 1 2 3 4 6 12
Prima: false
PS D:\vscode\College\Modul 10> go run u3.go
Bilangan: 7
Faktor: 1 7
Prima: true
PS D:\vscode\College\Modul 10>
```

109082500154
S1F-13-07
Ferdinand Axel Valerian

Ln 3, Col 24 | 46 character | Plain t | 100% | Wind | UTF-8

Deskripsi program

Program ini dirancang untuk menganalisis sifat faktorisasi suatu bilangan bulat positif. Program menerima input sebuah bilangan, kemudian mencari dan menampilkan semua faktor pembagi bilangan tersebut. Selanjutnya, program menentukan status bilangan tersebut apakah termasuk bilangan prima atau bukan.

Cara kerja program dimulai dengan membaca input bilangan dari pengguna. Program kemudian melakukan iterasi dari angka 1 hingga bilangan itu sendiri, memeriksa setiap angka apakah habis membagi bilangan input. Semua angka yang memenuhi kondisi ini dicetak sebagai faktor.

Setelah menemukan semua faktor, program menghitung total jumlah faktor yang ditemukan. Bilangan dinyatakan sebagai bilangan prima jika hanya memiliki tepat dua faktor, yaitu angka 1 dan bilangan itu sendiri. Hasil pengecekan ini ditampilkan dalam bentuk boolean true atau false.

