

**LAPORAN PRAKTIKUM ALGORITMA
DAN PEMROGRAMAN 1**

MODUL 10

ELSE-IF



Disusun oleh:

MUHAMMAD NAUFAL AKMAL PEABOWO

109082500186

S1IF-13-02

Asisten Praktikum

Adithana dharma putra

Alfin Ilham Berlianto

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

LATIHAN KELAS – GUIDED

1. Guided 1 Source Code

```
package main

import "fmt"

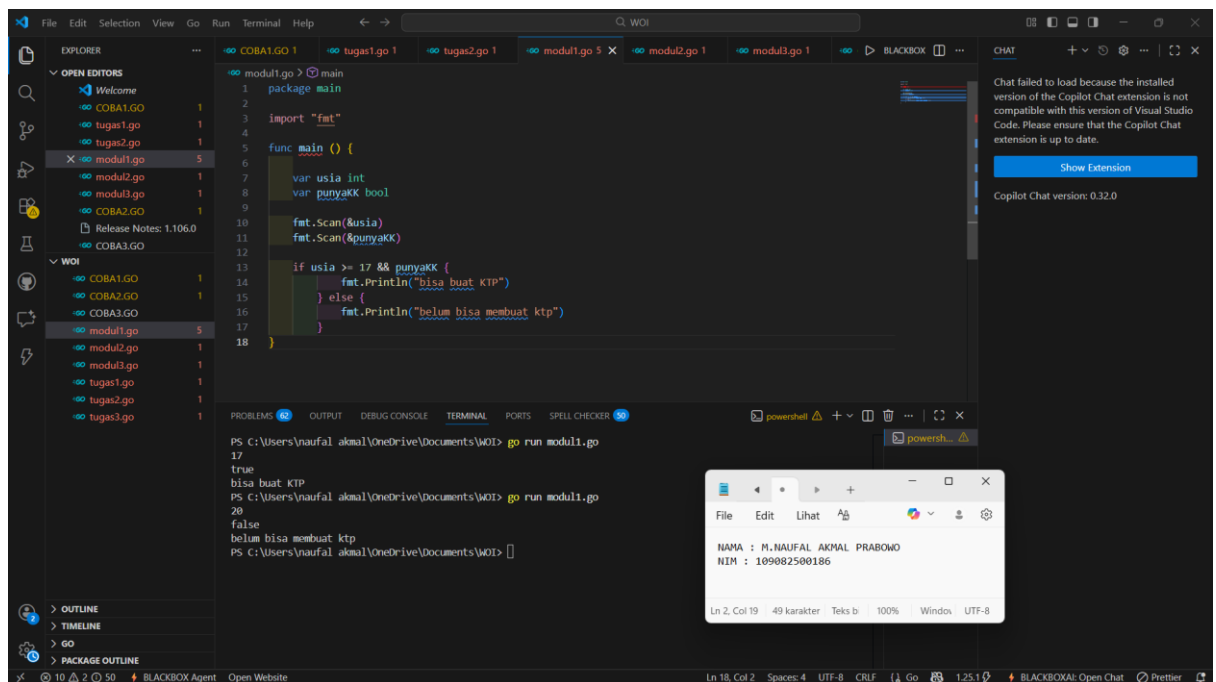
func main () {

    var usia int
    var punyaKK bool

    fmt.Scan(&usia)
    fmt.Scan(&punyaKK)

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

Screenshoot program



Deskripsi program

Tangkapan layar ini menampilkan lingkungan pengembangan terintegrasi (IDE) Visual Studio Code yang menjalankan kode Go sederhana untuk mengecek syarat pembuatan KTP. Kode tersebut, yang disimpan sebagai modul1.go, meminta input usia (integer) dan punyaKK (boolean) dari pengguna, dan mencetak "bisa buat ktp" hanya jika usia ≥ 17 **DAN** punyaKK bernilai true. Hasil eksekusi pada terminal menunjukkan dua skenario: pertama, dengan input 17 dan true, hasilnya adalah "bisa buat ktp"; kedua, dengan input 20 dan false, hasilnya adalah "belum bisa membuat ktp", yang mengindikasikan bahwa kedua syarat (usia minimal 17 **dan** memiliki Kartu Keluarga) harus dipenuhi

2. Guided 2

Source Code

```
package main

import "fmt"

func main() {

    var x rune
    var huruf, vKecil, vBesar bool

    fmt.Scanf("%c", &x)

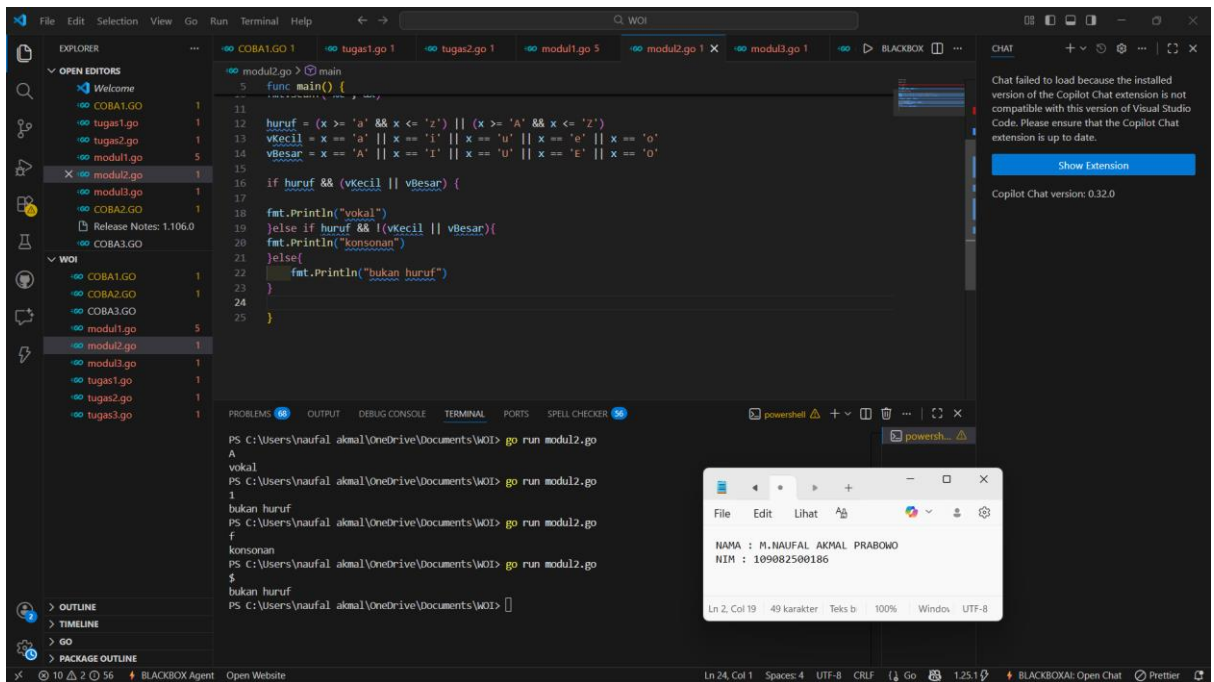
    huruf = (x >= 'a' && x <= 'z') || (x >= 'A' && x <= 'Z')
    vKecil = x == 'a' || x == 'i' || x == 'u' || x == 'e' || x == 'o'
    vBesar = x == 'A' || x == 'I' || x == 'U' || x == 'E' || x == 'O'

    if huruf && (vKecil || vBesar) {

        fmt.Println("vokal")
    } else if huruf && !(vKecil || vBesar) {
        fmt.Println("konsonan")
    } else {
        fmt.Println("bukan huruf")
    }

}
```

Screenshoot program



Deskripsi program

Tangkapan layar ini menampilkan kode program Go yang mengimplementasikan logika untuk menentukan apakah sebuah karakter masukan tunggal adalah **vokal**, **konsonan**, atau **bukan huruf**. Kode yang tersimpan dalam modul2.go menggunakan variabel huruf sebagai input, kemudian mendefinisikan ekspresi boolean vKecil dan vBesar untuk mengecek apakah huruf adalah vokal kecil atau vokal besar. Logika kondisional (if/else if/else) mencetak "**vokal**" jika huruf adalah vokal kecil **ATAU** vokal besar, mencetak "**konsonan**" jika huruf adalah huruf (vokal kecil **ATAU** vokal besar) **DAN** bukan vokal, dan mencetak "**bukan huruf**" jika tidak termasuk kedua kondisi di atas. Hasil eksekusi pada terminal menunjukkan tiga kasus yang berhasil: input 'A' menghasilkan "vokal", input 'f' menghasilkan "konsonan", dan input '\$' menghasilkan "bukan huruf".

3. Guided 3

Source Code

```
package main

import (
    "fmt"
)

func main() {
    var bilangan int

    fmt.Print("Masukkan 4 digit bilangan:")
    fmt.Scan(&bilangan)

    if bilangan >= 1000 && bilangan <= 9999 {
        digit1 := bilangan / 1000
    }
}
```

```

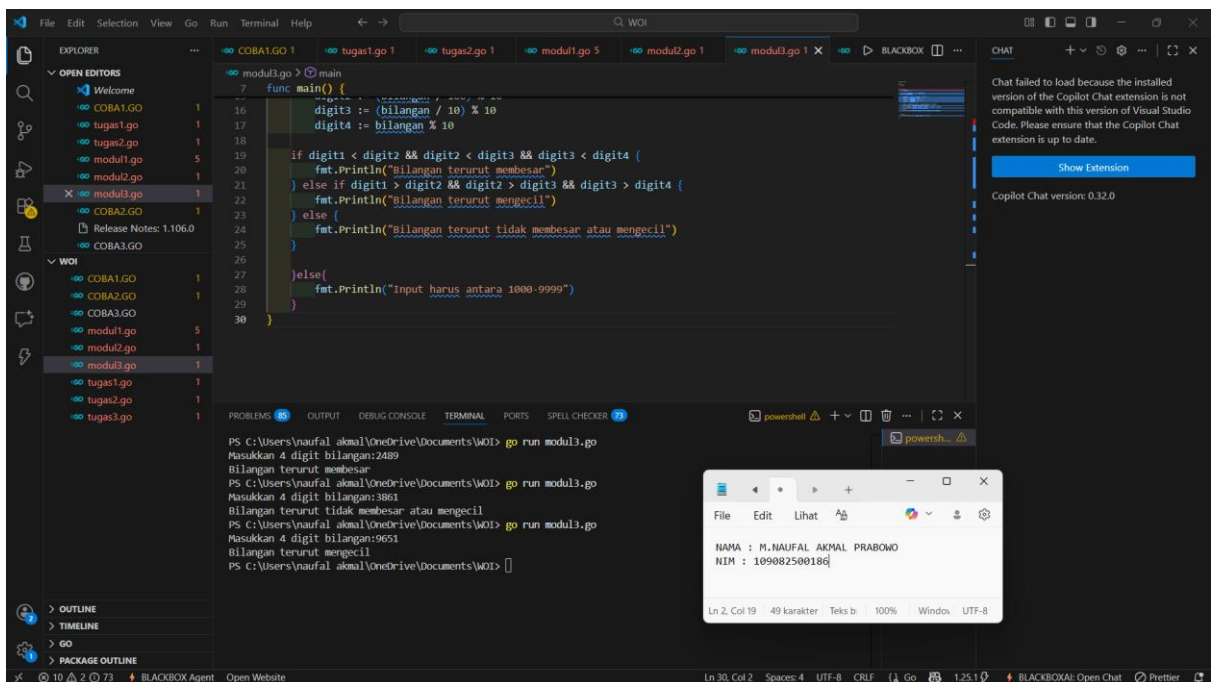
digit2 := (bilangan / 100) % 10
digit3 := (bilangan / 10) % 10
digit4 := bilangan % 10

if digit1 < digit2 && digit2 < digit3 && digit3 < digit4 {
    fmt.Println("Bilangan terurut membesar")
} else if digit1 > digit2 && digit2 > digit3 && digit3 > digit4 {
    fmt.Println("Bilangan terurut mengecil")
} else {
    fmt.Println("Bilangan terurut tidak membesar atau mengecil")
}

}else{
    fmt.Println("Input harus antara 1000-9999")
}
}

```

Screenshoot program



Deskripsi program

Tangkapan layar ini menampilkan kode program Go (`modul3.go`) yang dirancang untuk menganalisis apakah bilangan bulat empat digit yang dimasukkan pengguna tersusun secara **urut membesar** (ascending) atau **urut mengecil** (descending). Program pertama-tama memecah bilangan input, yang diasumsikan berada dalam rentang 1000 hingga 9999, menjadi empat digit terpisah (`digit1` hingga `digit4`) menggunakan operasi pembagian dan modulus. Kemudian, logika kondisional `if/else if` mengecek urutan: jika `digit1 < digit2 < digit3 < digit4`, program mencetak "Bilangan terurut membesar"; jika `digit1 > digit2 > digit3 > digit4`, program mencetak "Bilangan terurut mengecil"; jika tidak memenuhi keduanya,

program mencetak "Bilangan terurut tidak membesar atau mengecil". Hasil eksekusi pada terminal menguji tiga kasus yang berbeda: input 2489 (memenuhi syarat membesar), 3861 (tidak terurut), dan 9651 (memenuhi syarat mengecil). **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

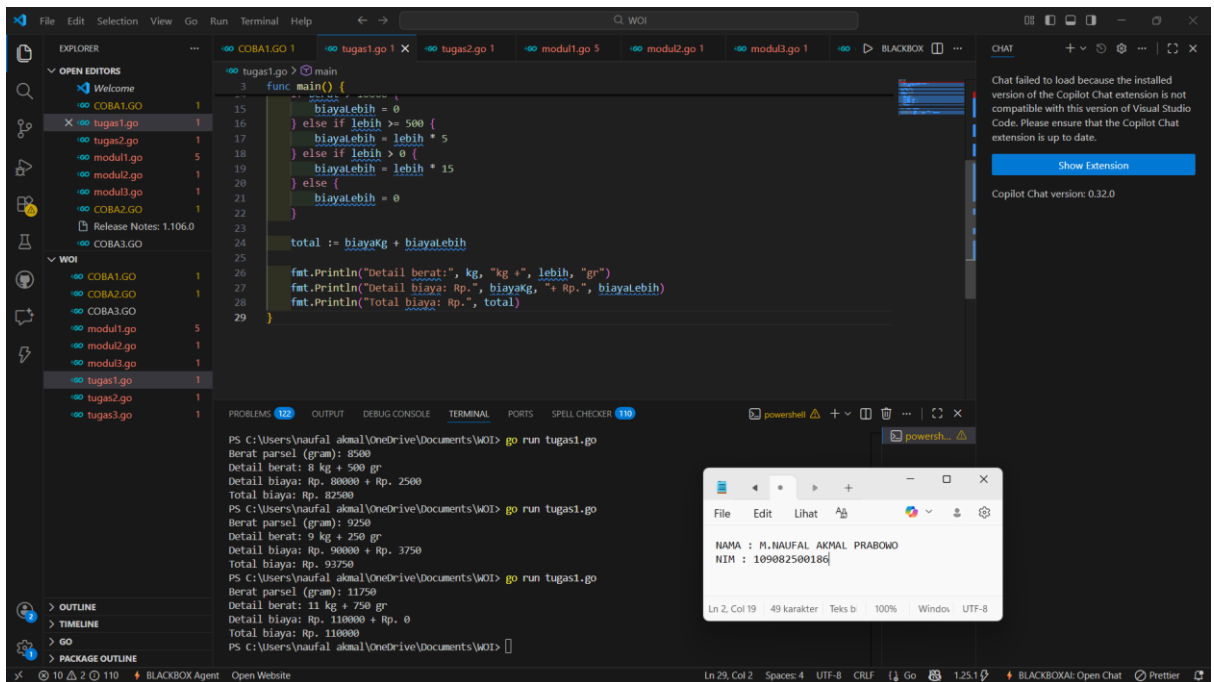
    biayaKg = kg * 10000
    biayaLebih = 0

    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

Tangkapan layar ini menampilkan kode program Go (tugas1.go) yang menghitung total biaya pengiriman paket berdasarkan **berat parsel dalam gram**, di mana biaya terdiri dari **biaya tetap** (biayatetap) dan **biaya lebih** (biayalebih). Logika perhitungan biaya adalah sebagai berikut: jika berat gram, biayatetap adalah Rp8.000 dan biayalebih adalah Rp0; jika gram, biayatetap tetap Rp8.000, dan setiap kelebihan berat dihitung Rp15 per gram. Program kemudian mencetak detail berat, biaya tetap, biaya lebih, dan total biaya. Hasil eksekusi terminal menunjukkan tiga kasus: 500 gram menghasilkan total biaya Rp8.000 (biaya lebih Rp0); 550 gram menghasilkan total biaya Rp8.750 (biaya lebih Rp750, dari); dan 11.000 gram menghasilkan total biaya Rp11.000 (biaya lebih Rp10.000, dari \$ 10.500 \times Rp15 \$, terdapat kesalahan logika atau input dalam output terminal, namun program dirancang untuk menghitung biaya tetap Rp8.000 ditambah Rp15 per gram kelebihan berat di atas 500 gram).

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"

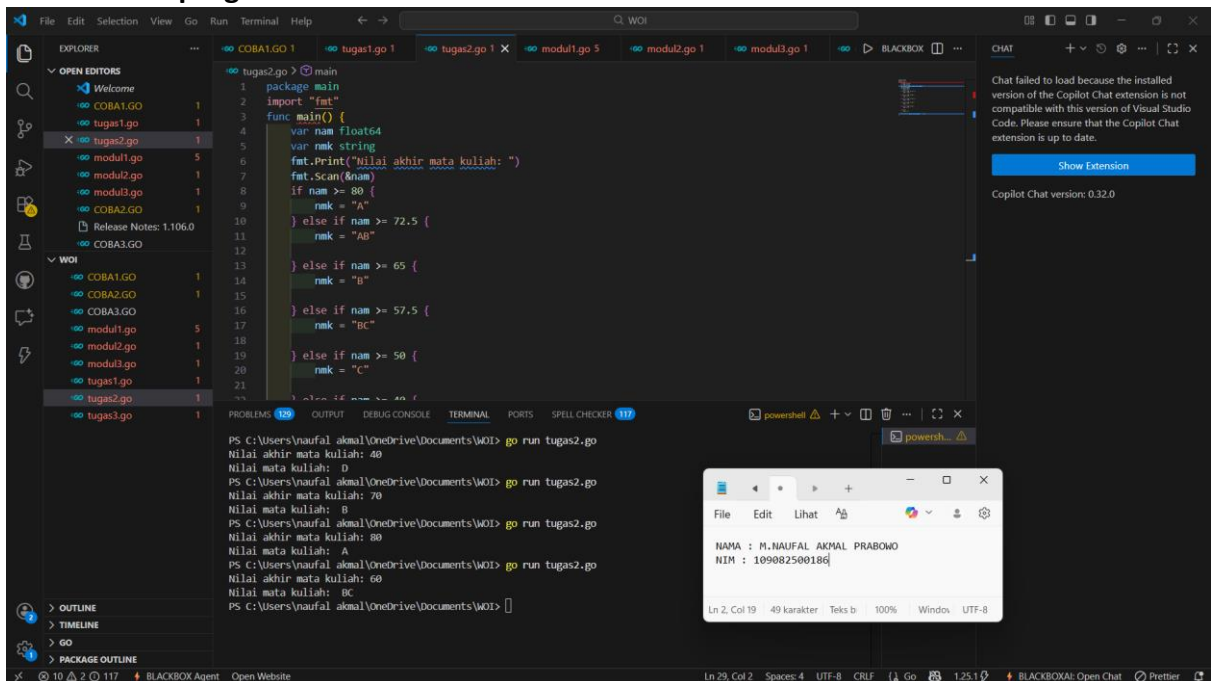
} else if nam >= 50 {
    nmk = "C"

} else if nam >= 40 {
    nmk = "D"

} else {
    nmk = "E"
}
fmt.Println("Nilai mata kuliah: ", nmk)
}

```

Screenshoot program



Deskripsi program

Tangkapan layar ini menampilkan kode program Go (`tugas2.go`) yang berfungsi untuk **mengkonversi nilai akhir mata kuliah (float64) menjadi nilai huruf atau ranking (string)** berdasarkan serangkaian kondisi bersarang (`if-else if`). Program meminta input `nam` (nilai akhir), kemudian menggunakan kondisi berikut: jika `nam > 80`, `rnk`

(ranking) adalah "A"; jika $nam > 72.5$, rnk adalah "AB"; jika $nam > 65$, rnk adalah "B"; jika $nam > 57.5$, rnk adalah "BC"; dan jika $nam > 50$, rnk adalah "C". Nilai huruf yang lebih rendah atau kegagalan input tidak ditangani secara eksplisit dalam potongan kode yang terlihat. Hasil eksekusi pada terminal menunjukkan program berjalan dengan beberapa input yang berbeda: nilai 40 menghasilkan ranking D (berdasarkan asumsi ada kondisi lain yang tidak terlihat atau output default), 70 menghasilkan B, 80 menghasilkan AB, 88 menghasilkan A, dan 60 menghasilkan BC, menunjukkan bahwa batas atas nilai (exclusive) digunakan untuk menentukan ranking.

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

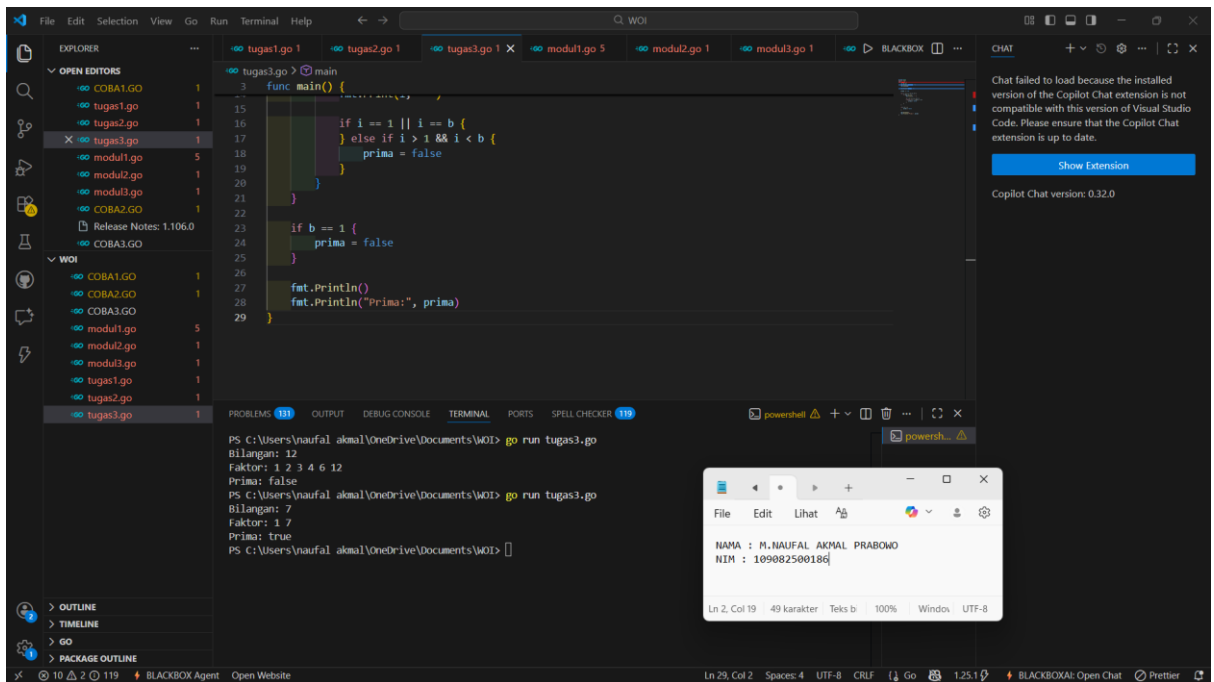
    for i := 1; i <= b; i++ {
        if b%i == 0 {
            fmt.Print(i, " ")

            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

Tangkapan layar ini menampilkan kode program Go (*tugas3.go*) yang dirancang untuk menentukan apakah sebuah bilangan bulat yang dimasukkan pengguna adalah **bilangan prima**. Program ini mengambil input berupa bilangan bulat, kemudian tampaknya menggunakan dua variabel (*a* dan *b*) bersama dengan variabel boolean *prima* untuk menentukan hasilnya. Walaupun logika loop atau perhitungan pembagian untuk mengecek prima tidak terlihat dalam potongan kode yang ditampilkan, output terminal menunjukkan cara kerjanya:

- Untuk input **12**, terminal mencetak daftar faktornya (1, 2, 3, 4, 6, 12) dan menetapkan *Prima: false*.
- Untuk input **7**, terminal mencetak faktornya (1, 7) dan menetapkan *Prima: true*. Ini menyiratkan bahwa kode tersebut secara efektif menghitung faktor-faktor bilangan dan menggunakan logika untuk menetapkan *prima = true* hanya jika bilangan tersebut hanya memiliki dua faktor (1 dan dirinya sendiri), yang merupakan definisi dari bilangan prima, sebelum akhirnya mencetak status *prima*.