

**LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL 2
REVIEW STRUKTUR KONTROL**



Oleh:

OKTAVANIA AYU RAHMADANTY

2311102240

S1IF-11-02

**S1 TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024**

I. DASAR TEORI

Dasar Teori

Dalam kerangka program yang ditulis dalam bahasa pemrograman Go, program utama selalu mempunyai dua komponen berikut:

- package main merupakan penanda bahwa file ini berisi program utama.
- func main() berisi kode utama dari sebuah program Go.

Komentar, bukan bagian dari kode program, dan dapat ditulis di mana saja di dalam program:

- Satu baris teks yang diawali dengan garis miring ganda ('//') s.d. akhir baris, atau.
- Beberapa baris teks yang dimulai dengan pasangan karakter '/*' dan diakhiri dengan '*/'.

```
1 // Setiap program utama dimulai dengan "package main"
2 package main
3
4 // Impor paket yang dibutuhkan, "fmt" berisi proses I/O standar
5 import "fmt"
6
7 // Kode program utama dalam "fungsi main"
8 func main() {
9     ...
10 }
```

1) Koding, Kompilasi, dan Eksekusi Go

Koding

- Tidak berbeda dengan penulisan program sumber dalam bahasa lain, program Go harus dibuat menggunakan penyunting teks dan disimpan dalam format teks, bukan dalam format dokumen (doc, docx, atau lainnya).
- Setiap program go disimpan dalam file teks dengan ekstensi *.go, dengan nama bebas. Sebaiknya nama file adalah nama untuk program tersebut.
- Setiap satu program lengkap Go disimpan dalam satu folder tersendiri. Nama folder merupakan nama program tersebut. Karena itu secara prinsip, satu program Go dapat dipecah dalam beberapa file dengan ekstensi *.go selama disimpan dalam folder yang sama.

Kompilasi

Beberapa bahasa pemrograman dirancang untuk diimplementasikan sebagai interpreter dan lainnya sebagai kompilator. Interpreter akan membaca setiap baris

instruksi dan kemudian langsung mengeksekusinya, dengan hanya sedikit pemeriksaan apakah penulisan keseluruhan program sudah benar atau belum. Kompilator akan memeriksa keseluruhan program sumber dan kemudian mengubahnya menjadi program eksekutabel, sehingga konsistensi penulisan informatics lab (seperti penggunaan tipe data) sudah diperiksa sebelum eksekusi. Selain itu karena program dibuat menjadi eksekutabel lebih dahulu, proses optimasi dapat dilakukan sehingga program menjadi sangat efisien.

Go diimplementasikan sebagai kompilator. Berikut adalah contoh sesi yang biasa dilakukan saat mengkompilasi dan mengeksekusi program dalam bahasa Go:

- Panggil shell atau terminal (program/utiliti cmd.exe di Windows)
- Masuk ke dalam (cd) folder program (normalnya ada di C:\Users\go\src\ atau yang sejenis)
- Kemudian panggil perintah go build atau go build file.go untuk mengkompilasi file.go
- Jika gagal, akan muncul pesan eror yang sesuai, pelajari dengan baik pesan tersebut, perbaiki teks program sumber, kemudian ulangi proses build-nya.
- Jika berhasil maka pada folder tersebut akan dibuat program dengan nama yang sama dan diakhiri dengan .exe (untuk Windows)
- Panggil program eksekutabel tersebut dari terminal yang sama. Jangan memanggil program tersebut dengan mengklik eksekutabel tersebut dari folder karena program kalian hanya berbasis teks, bukan/belum dirancang dengan tampilan Windows.

Catatan

Semua proses terkait bahasa Go dilakukan melalui utilitas go. Beberapa opsi dengan utilitas go:

- go build: mengkompilasi program sumber yang ada dalam folder menjadi sebuah program.
- go build file.go: mengkompilasi program sumber file.go saja.
- go fmt: membaca semua program sumber dalam folder dan mereformat penulisannya agar sesuai dengan standar penulisan program sumber Go.
- go clean: membersihkan file-file dalam folder sehingga tersisa program sumber nya saja.

II. GUIDED

Guided 1

Source Code

```
package main

import "fmt"

func main() {

    var (

        satu, dua, tiga string

        temp                string

    )

    fmt.Print("Masukan input string: ")

    fmt.Scanln(&satu)

    fmt.Print("Masukan input string: ")

    fmt.Scanln(&dua)

    fmt.Print("Masukan input string: ")

    fmt.Scanln(&tiga)

    fmt.Println("Output awal = " + satu + " " + dua +
" " + tiga)

    temp = satu

    satu = dua

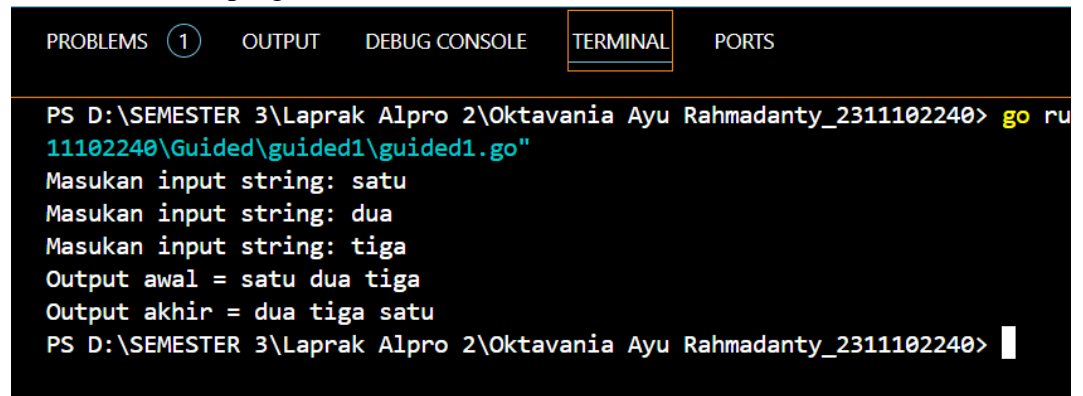
    dua = tiga

    tiga = temp

    fmt.Println("Output akhir = " + satu + " " + dua +
" " + tiga)

}
```

Screenshot hasil program

A screenshot of a terminal window showing the execution of a Go program. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (selected), and PORTS. The command prompt shows the user is in the directory D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240. The user runs 'go run 11102240\Guided\guided1\guided1.go'. The program prompts for three input strings: 'satu', 'dua', and 'tiga'. It then displays the initial output 'satu dua tiga' and the final output after swapping 'dua' and 'tiga', which is 'dua tiga satu'.

```
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go run 11102240\Guided\guided1\guided1.go
Masukan input string: satu
Masukan input string: dua
Masukan input string: tiga
Output awal = satu dua tiga
Output akhir = dua tiga satu
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240>
```

Penjelasan:

Program di bawah ini merupakan program sederhana yang meminta tiga input string dari pengguna, menampilkan string tersebut dalam urutan awal, lalu menukar nilai-nilai string tersebut dan menampilkannya kembali dalam urutan yang baru setelah pertukaran.

Guided 2

Source Code

```
package main

import (
    "fmt"
)

func main() {
    var tahun int

    fmt.Print("Masukkan tahun: ")

    fmt.Scan(&tahun)
```

```

    kabisat := (tahun%400 == 0) || (tahun%4 == 0 &&
tahun%100 != 0)

    fmt.Printf("Kabisat: %t\n", kabisat)
}

```

Screenshot hasil program

```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go
11102240\Guided\guided2\tempCodeRunnerFile.go"
Masukkan tahun: 2024
Kabisat: true
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go
11102240\Guided\guided2\tempCodeRunnerFile.go"
Masukkan tahun: 2017
Kabisat: false
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240>

```

Penjelasan:

Program berikut adalah program sederhana dalam bahasa Go yang memeriksa apakah sebuah tahun yang diberikan oleh pengguna merupakan tahun kabisat atau tidak.

Guided 3

Source Code

```

package main

import (
    "fmt"
    "math"
)

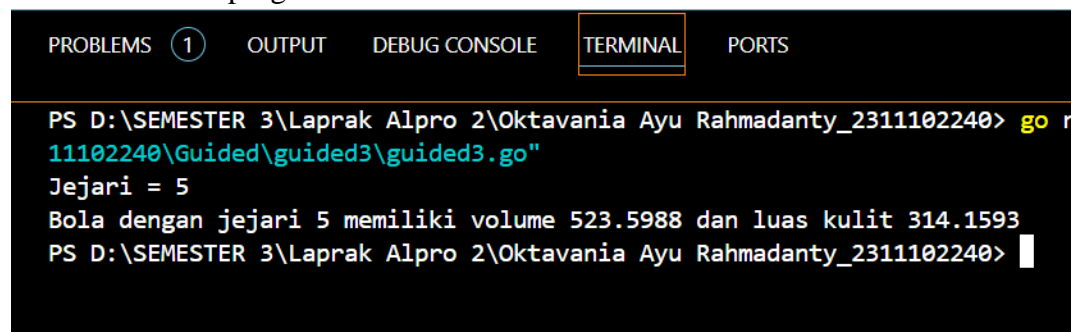
func main() {
    var r float64

    fmt.Print("Jejari = ")
    fmt.Scan(&r)
}

```

```
    volumeBola := (4.0 / 3.0) * math.Pi * math.Pow(r,  
3)  
    luasBola := 4 * math.Pi * math.Pow(r, 2)  
  
    fmt.Printf("Bola dengan jejari %.0f memiliki  
volume %.4f dan luas kulit %.4f\n", r, volumeBola,  
luasBola)  
}
```

Screenshot hasil program



```
PROBLEMS ① OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go r  
11102240\Guided\guided3\guided3.go"  
Jejari = 5  
Bola dengan jejari 5 memiliki volume 523.5988 dan luas kulit 314.1593  
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> |
```

Penjelasan:

Program berikut adalah program dalam bahasa Go yang menghitung volume dan luas permukaan bola berdasarkan jejari (radius) yang diberikan oleh pengguna.

III. UNGUIDED

Unguided 1

Source Code

```
package main

import "fmt"

func main() {
    var warna1, warna2, warna3, warna4 string
    var berhasil bool

    for i := 1; i <= 5; i++ {
        fmt.Printf("Percobaan %d:\n", i)
        fmt.Print("Warna gelas 1: ")
        fmt.Scan(&warna1)
        fmt.Print("Warna gelas 2: ")
        fmt.Scan(&warna2)
        fmt.Print("Warna gelas 3: ")
        fmt.Scan(&warna3)
        fmt.Print("Warna gelas 4: ")
        fmt.Scan(&warna4)

        if warna1 == "merah" && warna2 == "kuning" &&
warna3 == "hijau" && warna4 == "ungu" {
            berhasil = true
        } else {
            berhasil = false
            break // Jika salah satu percobaan gagal,
langsung hentikan perulangan
        }
    }

    fmt.Printf("BERHASIL: %t\n", berhasil)
}
```

Screenshot hasil program


```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\SEMESTER 3\Laparak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go
0\Unguided\unguided1\unguided1.go"
Percobaan 1:
Warna gelas 1: merah kuning hijau ungu
Warna gelas 2: Warna gelas 3: Warna gelas 4: Percobaan 2:
Warna gelas 1: merah kuning hijau ungu
Warna gelas 2: Warna gelas 3: Warna gelas 4: Percobaan 3:
Warna gelas 1: merah kuning hijau ungu
Warna gelas 2: Warna gelas 3: Warna gelas 4: Percobaan 4:
Warna gelas 1: merah kuning hijau ungu
Warna gelas 2: Warna gelas 3: Warna gelas 4: Percobaan 5:
Warna gelas 1: merah kuning hijau ungu
Warna gelas 2: Warna gelas 3: Warna gelas 4: BERHASIL: true
PS D:\SEMESTER 3\Laparak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go
0\Unguided\unguided1\unguided1.go"
Percobaan 1:
Warna gelas 1: merah kuning hijau ungu
Warna gelas 2: Warna gelas 3: Warna gelas 4: Percobaan 2:
Warna gelas 1: merah kuning hijau ungu
Warna gelas 2: Warna gelas 3: Warna gelas 4: Percobaan 3:
Warna gelas 1: merah kuning hijau ungu
Warna gelas 2: Warna gelas 3: Warna gelas 4: Percobaan 4:
Warna gelas 1: ungu kuning hijau merah
Warna gelas 2: Warna gelas 3: Warna gelas 4: BERHASIL: false
PS D:\SEMESTER 3\Laparak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> 
```

Penjelasan:

Bisa ditambahkan validasi untuk memastikan input yang dimasukkan hanya berupa warna yang ditentukan (merah, kuning, hijau, ungu). Bisa dibuat fungsi terpisah untuk mengecek apakah urutan warna sudah sesuai, sehingga kode menjadi lebih modular. Bisa menggunakan array untuk menyimpan warna dari setiap percobaan, sehingga lebih efisien untuk percobaan yang lebih banyak.

Unguided 2

Source Code

```
package main

import (
```

```
"bufio"

"fmt"

"os"

"strings"
)

func main() {

    var pita string

    jumlahBunga := 0

    scanner := bufio.NewScanner(os.Stdin)

    for {

        fmt.Printf("Bunga %d: ", jumlahBunga+1)

        scanner.Scan()

        flower := scanner.Text()

        if strings.ToUpper(flower) == "SELESAI" {

            break

        }

        if pita == "" {

            pita += flower

        } else {

            pita += " - " + flower

        }

    }
```

```

        jumlahBunga++
    }

    fmt.Println("Pita:", pita)

    fmt.Printf("Bunga: %d\n", jumlahBunga)
}

```

Screenshot hasil program

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go run
11102240\Unguided\unguided2\guided2.go"
Bunga 1: mawar
Bunga 2: melati
Bunga 3: selesai
Pita: mawar - melati
Bunga: 2
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240>

```

Penjelasan:

Program tidak lagi meminta input N (jumlah bunga). Sebagai gantinya, program terus meminta nama bunga hingga pengguna mengetikkan "SELESAI". Loop akan berjalan terus menerus sampai pengguna mengetikkan "SELESAI", dan jumlah bunga (`flowerCount`) bertambah setiap kali nama bunga yang valid dimasukkan.

Unguided 3

Source Code

```

package main

import (

```

```

    "bufio"

    "fmt"

    "os"

    "strconv"

    "strings"

    "math"
)

func main() {

    scanner := bufio.NewScanner(os.Stdin)

    for {

        fmt.Print("Masukkan berat belanjaan di kedua
kantong: ")

        scanner.Scan()

        input := scanner.Text()

        berat := strings.Split(input, " ")

        if len(berat) != 2 {

            fmt.Println("Input harus terdiri dari dua
angka.")

            continue

        }

        berat1, err1 := strconv.ParseFloat(berat[0],
64)

```

```
64)      berat2, err2 := strconv.ParseFloat(berat[1],

if err1 != nil || err2 != nil {
    fmt.Println("Masukkan berat yang valid!")
    continue
}

if berat1 < 0 || berat2 < 0 {
    fmt.Println("Proses selesai.")
    break
}

totalBerat := berat1 + berat2

if totalBerat > 150 {
    fmt.Println("Proses selesai.")
    break
}

selisihBerat := math.Abs(berat1 - berat2)

if selisihBerat >= 9 {
    fmt.Println("Sepeda motor pak Andi akan
oleng: true")
} else {
    fmt.Println("Sepeda motor pak Andi akan
oleng: false")
}
```

```

    }
}
}

```

Screenshot hasil program

```

PS D:\SEMESTER 3\Laparak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go run 11102240\Unguided\unguided3\tempCodeRunnerFile.go
Masukkan berat belanjaan di kedua kantong: 5 10
Sepeda motor pak Andi akan oleng: false
Masukkan berat belanjaan di kedua kantong: 55.6 70.2
Sepeda motor pak Andi akan oleng: true
Masukkan berat belanjaan di kedua kantong: 59.5 98.7
Proses selesai.
PS D:\SEMESTER 3\Laparak Alpro 2\Oktavania Ayu Rahmadanty_2311102240>

```

Penjelasan:

Program meminta user untuk memasukkan dua angka yang mewakili berat dari dua kantong belanjaan. Input ini dipisah menjadi dua bagian menggunakan fungsi `strings.Split()`. Jika berat salah satu kantong bernilai negatif, program akan berhenti dengan menampilkan pesan "Proses selesai."

Jika total berat dari kedua kantong melebihi 150 kg, program juga akan berhenti dan menampilkan "Proses selesai." Menggunakan fungsi `math.Abs()`, program menghitung selisih absolut antara berat kantong pertama dan kedua. Jika selisihnya lebih besar atau sama dengan 9 kg, maka ditampilkan pesan "Sepeda motor pak Andi akan oleng: true", sebaliknya ditampilkan "false." Program akan terus meminta input dari user sampai salah satu kondisi pemberhentian terpenuhi (total berat lebih dari 150 kg atau berat negatif).

Unguided 4

Source Code

```

package main

import (
    "fmt"

```

```

    "math"
)

func hitungAkar2(K int) float64 {
    hasil := 1.0
    for k := 0; k < K; k++ {
        pembilang := math.Pow(4*float64(k)+2, 2)
        penyebut := (4*float64(k) + 1) * (4*float64(k)
+ 3)
        hasil *= pembilang / penyebut
    }
    return hasil
}

func main() {
    var K int
    fmt.Print("Masukkan nilai K = ")
    fmt.Scan(&K)

    akar2Hampiran := hitungAkar2(K)
    fmt.Printf("Nilai hampiran akar 2 = %.10f\n",
akar2Hampiran)
}

```

Screenshot hasil program

```
PROBLEMS (3) OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go ru
11102240\Unguided\unguided4\unguided4.go"
Masukkan nilai K = 100
Nilai hampiran akar 2 = 1.4133299615
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> |
```

Penjelasan:

HitungAkar2(K int) float64: fungsi ini menghitung hampiran nilai $2\sqrt{2}$ dengan mengalikan suku-suku dari deret hingga suku ke-KKK. Fungsi utama meminta input nilai KKK dari pengguna, kemudian menghitung hampiran $2\sqrt{2}$ dan mencetak hasilnya dengan 10 angka di belakang koma.

Unguided 5

Source Code

```
package main

import "fmt"

func main() {

    var beratParsel int

    fmt.Print("Masukkan berat parsel (dalam gram): ")

    fmt.Scan(&beratParsel)

    kg := beratParsel / 1000

    sisaGram := beratParsel % 1000

    biayaDasar := kg * 10000

    var biayaTambahan int

    if sisaGram >= 500 {
```



```

        biayaTambahan = sisaGram * 5
    } else {
        biayaTambahan = sisaGram * 15
    }

    if kg > 10 && sisaGram < 500 {
        biayaTambahan = 0
    }

    totalBiaya := biayaDasar + biayaTambahan

    fmt.Printf("Total berat: %d kg %d gram\n", kg,
sisaGram)

    fmt.Printf("Biaya dasar: Rp %d\n", biayaDasar)

    fmt.Printf("Biaya tambahan: Rp %d\n",
biayaTambahan)

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

```

Screenshot hasil program

```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go r
11102240\Unguided\unguided5\unguided5.go"
Masukkan berat parsel (dalam gram): 8500
Total berat: 8 kg 500 gram
Biaya dasar: Rp 80000
Biaya tambahan: Rp 2500
Total biaya: Rp 82500
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240>

```

Penjelasan:

Simpan kode di atas sebagai file .go. Buka terminal atau command prompt dan navigasikan ke direktori tempat Anda menyimpan file tersebut. Jalankan program dengan perintah go run. Masukkan berat parsel ketika diminta. Program akan menampilkan hasil perhitungan biaya pengiriman.

Unguided 6

Source Code

```
package main

import (
    "fmt"
)

func main() {
    var nam float64
    var nmk string

    fmt.Print("Nilai akhir mata kuliah: ")
    fmt.Scanln(&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)
}

```

Screenshot hasil program

```

11102240\Unguided\unguided6\unguided6.go"
Nilai akhir mata kuliah: 93.5
Nilai mata kuliah: A
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go r
0\Unguided\unguided6\unguided6.go"
Nilai akhir mata kuliah: 70.6
Nilai mata kuliah: B
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go r
0\Unguided\unguided6\unguided6.go"
Nilai akhir mata kuliah: 49.5
Nilai mata kuliah: D
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240>

```

Penjelasan:

- a. Jika nam diberikan adalah 80.1, apa keluaran dari program tersebut? Apakah eksekusi program tersebut sesuai spesifikasi soal?
Tidak, eksekusi tidak sesuai spesifikasi soal. Nilai 80.1 seharusnya menghasilkan "A", tetapi karena urutan perbandingan tidak tepat dan tidak menggunakan else if, program memberikan hasil yang salah.
- b. Apa saja kesalahan dari program tersebut? Mengapa demikian? Jelaskan alur program seharusnya!
 1. Tidak ada penggunaan else if: Semua kondisi if dievaluasi secara independen. Ini menyebabkan masalah ketika nilai nam memenuhi beberapa kondisi sekaligus. Misalnya, jika nam = 80.1, ia akan melewati lebih dari satu blok if dan akhirnya memberikan nilai yang tidak diharapkan.
 2. Variabel yang tertukar: Variabel nam digunakan untuk dua hal yang berbeda, yaitu untuk menyimpan nilai input nam dan nilai

huruf seperti "A", "B", dan lain-lain. Ini membuat program menjadi ambigu dan sulit dipahami.

3. Output salah: Karena kesalahan logika dalam urutan if, hasil yang diinginkan tidak tercapai untuk beberapa nilai.

- else if: Digunakan untuk memastikan bahwa hanya satu kondisi yang akan dievaluasi dan mencegah eksekusi beberapa blok if yang tidak diinginkan.
- nmk: Variabel nmk digunakan untuk menyimpan nilai huruf (grade), sehingga tidak ada konflik dengan variabel nam yang menyimpan nilai input.

Unguided 7

Source Code

```
package main

import (
    "fmt"
)

func cariFaktor(b int) []int {
    var faktor []int
    for i := 1; i <= b; i++ {
        if b%i == 0 {
            faktor = append(faktor, i)
        }
    }
    return faktor
}
```

```
func cekPrima(b int) bool {  
    faktor := cariFaktor(b)  
  
    return len(faktor) == 2  
}  
func main() {  
    var b int  
    fmt.Print("Masukkan bilangan: ")  
    fmt.Scan(&b)  
  
    faktor := cariFaktor(b)  
    fmt.Printf("Faktor: ")  
    for _, f := range faktor {  
        fmt.Printf("%d ", f)  
    }  
    fmt.Println()  
  
    if cekPrima(b) {  
        fmt.Println("Prima: true")  
    } else {  
        fmt.Println("Prima: false")  
    }  
}
```

Screenshot hasil program

```
PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go run "c:\11102240\Unguided\unguided7\unguided7.go"
Masukkan bilangan: 12
Faktor: 1 2 3 4 6 12
Prima: false
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> go run "c:\11102240\Unguided\unguided7\unguided7.go"
Masukkan bilangan: 7
Faktor: 1 7
Prima: true
PS D:\SEMESTER 3\Laprak Alpro 2\Oktavania Ayu Rahmadanty_2311102240> 
```

Penjelasan:

Fungsi ini menghitung dan mengembalikan semua faktor dari bilangan bbb. Ini mengecek apakah bilangan bbb adalah bilangan prima dengan memeriksa jumlah faktornya. Jika jumlah faktornya tepat dua, maka bilangan tersebut adalah bilangan prima. Fungsi utama meminta input bbb, menampilkan semua faktor dari bbb, dan mencetak apakah bilangan tersebut adalah bilangan prima atau tidak.