

**LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK**

**MODUL II
REVIEW STRUKTUR KONTROL**



Oleh:

NAMA: M.Arif Rachman

NIM: 2311102300

KELAS: S1IF-11-07

**S1 TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

2024

I. DASAR TEORI

Struktur kontrol dalam pemrograman adalah elemen penting yang mengarahkan alur eksekusi program berdasarkan kondisi tertentu. Dalam bahasa Go (Golang), struktur kontrol memberikan cara untuk membuat keputusan, mengulangi tindakan, dan mengelola alur program secara efisien. Berikut adalah beberapa struktur kontrol dasar dalam Go:

1. Struktur Kontrol Seleksi

- **If Statement:**

- Digunakan untuk mengeksekusi blok kode jika suatu kondisi terpenuhi.

- Sintaks:

```
if kondisi {  
    // Eksekusi jika kondisi benar  
}
```

- **If-Else Statement:**

- Menyediakan alternatif eksekusi jika kondisi tidak terpenuhi.

- Sintaks:

```
if kondisi {  
    // Eksekusi jika kondisi benar  
} else {  
    // Eksekusi jika kondisi salah  
}
```

- **If-Else If-Else Statement:**

- Memungkinkan pemeriksaan beberapa kondisi.

- Sintaks:

```
if kondisi1 {  
    // Eksekusi jika kondisi1 benar  
} else if kondisi2 {  
    // Eksekusi jika kondisi2 benar  
} else {  
    // Eksekusi jika semua kondisi salah  
}
```

- **Switch Statement:**

- Memilih satu dari beberapa blok kode untuk dieksekusi berdasarkan nilai ekspresi.
- Sintaks:

```
switch ekspresi {  
  case nilai1:  
    // Eksekusi jika ekspresi == nilai1  
  case nilai2:  
    // Eksekusi jika ekspresi == nilai2  
  default:  
    // Eksekusi jika tidak ada kasus yang cocok  
}
```

2. Struktur Kontrol Perulangan

- **For Loop:**

- Satu-satunya struktur perulangan dalam Go, digunakan untuk berbagai jenis iterasi.
- Sintaks:

```
for init; kondisi; pembaruan {  
  // Blok kode yang diulang  
}
```

- **For Range Loop:**

- Digunakan untuk mengiterasi elemen dalam array, slice, map, atau channel.
- Sintaks:

```
for indeks, nilai := range koleksi {  
  // Akses setiap elemen  
}
```

- **While Loop (Simulasi):**

- Tidak ada while loop eksplisit di Go, tetapi dapat disimulasikan dengan for loop.
- Sintaks:

```
for kondisi {  
  // Blok kode yang diulang  
}
```

- **Infinite Loop:**

- Loop yang berjalan tanpa batas, dihentikan dengan break.
- Sintaks:

```
for {  
  // Eksekusi terus menerus  
  if someCondition {  
    break  
  }  
}
```

3. Struktur Kontrol

- **Break and Continue Branch:**

"Break" adalah istilah yang digunakan untuk keluar dari perselisihan. terus digunakan untuk melanjutkan ke iterasi berikutnya.

- **Fallthrough, atau Switch:**

Meskipun kondisi case saat ini terpenuhi, switch melakukan fallthrough untuk melanjutkan eksekusi ke case berikutnya.

4. Penerapan Sistem Kontrol

- Efisiensi: Struktur kontrol menjamin bahwa program hanya menjalankan kode yang diperlukan.
- Keputusan dinamis memungkinkan pengambilan keputusan berdasarkan kondisi runtime.
- Pemeliharaan: Kode dengan struktur kontrol yang baik lebih mudah dipahami dan digunakan.

Untuk menulis program yang efektif dan efisien dalam Go, pengembang harus memahami struktur kontrol karena dirancang untuk menjadi mudah dan efektif.

II. GUIDED

1. Source Code

```
package main

import "fmt"

func main() {

    var nama string = "M. ARIF RACHMAN"

    var umur int = 19

    var tinggi float64 = 177.5

    var isSunny bool = false

    var inisial rune = 'A'


    fmt.Println("Nama:", nama)

    fmt.Println("Umur:", umur)

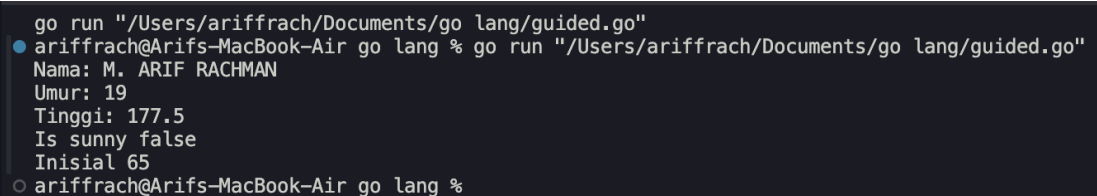
    fmt.Println("Tinggi:", tinggi)

    fmt.Println("Is sunny", isSunny)

    fmt.Println("Inisial", inisial)


}
```

Screenshot Output



```
go run "/Users/ariffrach/Documents/go lang/guided.go"
● ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Nama: M. ARIF RACHMAN
Umur: 19
Tinggi: 177.5
Is sunny false
Inisial 65
○ ariffrach@Arifs-MacBook-Air go lang %
```

Deskripsi Program

Program ini menyimpan dan menampilkan informasi mengenai seseorang, seperti nama "M. ARIF RACHMAN", umur 19 tahun, tinggi badan 177.5 cm, inisial huruf 'A', serta status cuaca yang ditandai dengan variabel boolean `isSunny` bernilai `false`, yang artinya cuaca sedang tidak cerah. Semua data tersebut ditampilkan ke layar menggunakan perintah `fmt.Println`.

2.Source Code

```
package main

import "fmt"

func main() {

    //Tahun kabisat

    var year int

    fmt.Print("Masukkan tahun: ")

    fmt.Scanf("%d", &year)

    if (year%400 == 0) || (year%4 == 0 && year%100 != 0) {

        fmt.Println(year, "adalah tahun kabisat (true)")

    } else {

        fmt.Println(year, "bukan tahun kabisat (false)")

    }

}
```

SCREENSHOOT OUTPUT

```
go run "/Users/ariffrach/Documents/go lang/guided.go"
● ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Masukkan tahun: 2078
2078 bukan tahun kabisat (false)
○ ariffrach@Arifs-MacBook-Air go lang %
```

Deskripsi program

- Program ini meminta pengguna memasukkan sebuah tahun, lalu memeriksa apakah tahun tersebut adalah tahun kabisat. Caranya adalah dengan menggunakan aturan bahwa tahun kabisat harus habis dibagi 400, atau habis dibagi 4 tetapi tidak habis dibagi 100. Setelah memeriksa kondisi ini, program akan mencetak pesan yang menyatakan apakah tahun tersebut tahun kabisat (true) atau bukan tahun kabisat (false).

3. SOURCE CODE

```
package main

import "fmt"

func main() {

    //Temperatur

    var temperaturCelsius float64

    fmt.Print("Temperatur Celsius: ")

    fmt.Scanln(&temperaturCelsius)


    var temperaturFahrenheit float64 = (temperaturCelsius * 9 / 5) + 32

    var temperaturReamur float64 = temperaturCelsius * 4 / 5

    var temperaturKelvin float64 = temperaturCelsius + 273.15

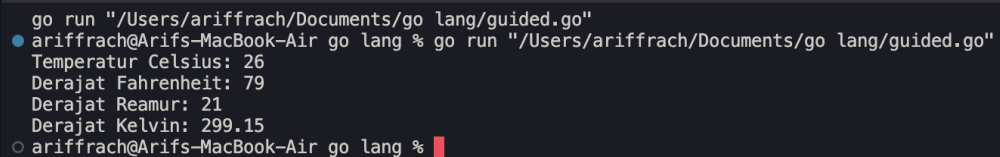

    fmt.Printf("Derajat Fahrenheit: %.0f\n", temperaturFahrenheit)

    fmt.Printf("Derajat Reamur: %.0f\n", temperaturReamur)

    fmt.Printf("Derajat Kelvin: %.2f\n", temperaturKelvin)


}
```

SCREENSHOOT OUTPUT



```
go run "/Users/ariffrach/Documents/go lang/guided.go"
● ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Temperatur Celsius: 26
Derajat Fahrenheit: 79
Derajat Reamur: 21
Derajat Kelvin: 299.15
○ ariffrach@Arifs-MacBook-Air go lang %
```

Deskripsi program

Program ini meminta pengguna untuk memasukkan suhu dalam Celsius, lalu mengonversinya ke tiga skala suhu lainnya: Fahrenheit, Reamur, dan Kelvin. Rumus yang digunakan adalah:

- Fahrenheit = $(\text{Celsius} \times 9/5) + 32$

- Reamur = $\text{Celsius} \times 4/5$

- Kelvin = $\text{Celsius} + 273.15$

4. 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)

}
```

SCREENSHOOT OUTPUT

```
go run "/Users/ariffrach/Documents/go lang/guided.go"
● ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Masukan input string: arip
Masukan input string: informatika
Masukan input string: maba
Output awal = arip informatika maba
Output akhir = informatika maba arip
○ ariffrach@Arifs-MacBook-Air go lang %
```

Deskripsi Program

Program ini meminta pengguna untuk memasukkan tiga string secara berurutan, kemudian menampilkan ketiga string tersebut dalam satu baris. Setelah itu, program melakukan pertukaran nilai di antara ketiga string: string pertama menjadi string kedua, string kedua menjadi string ketiga, dan string ketiga menjadi string pertama. Hasil dari pertukaran ini kemudian ditampilkan kembali kepada pengguna.

III. UNGUIDED

2B.

No 1. Source Code

```
package main
```

```
import (  
    "bufio"  
    "fmt"  
    "os"  
    "strings"  
)
```

```
func main() {  
    expectedColors := []string{"merah", "kuning", "hijau", "ungu"}  
  
    scanner := bufio.NewScanner(os.Stdin)  
  
    success := true  
  
    for i := 1; i <= 5; i++ {  
        fmt.Printf("Percobaan %d: ", i)  
  
        scanner.Scan()  
        colors := strings.Fields(scanner.Text())
```

```
if len(colors) != len(expectedColors) {  
    success = false  
    break  
}
```

```
for j, color := range colors {  
    if color != expectedColors[j] {  
        success = false  
        break  
    }  
}
```

```
if !success {  
    break  
}  
}
```

```
fmt.Printf("BERHASIL: %v\n", success)  
}
```

Screenshoot Output

```
go run "/Users/ariffrach/Documents/go lang/guided.go"
● ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: merah kuning hijau ungu
Percobaan 5: merah kuning hijau ungu
BERHASIL: true
○ ariffrach@Arifs-MacBook-Air go lang %
```

Deskripsi Program

Program ini meminta pengguna untuk memasukkan serangkaian warna sebanyak 5 kali percobaan, dan akan memeriksa apakah urutan warna yang dimasukkan sesuai dengan urutan yang diharapkan: "merah", "kuning", "hijau", dan "ungu". Setiap kali pengguna memasukkan input, program memecah input menjadi kata-kata dan membandingkannya dengan daftar warna yang diharapkan. Jika urutannya salah, program menghentikan pengecekan dan menganggap percobaan gagal. Pada akhirnya, program mencetak apakah semua percobaan berhasil atau tidak dengan mengeluarkan nilai `true` jika berhasil dan `false` jika gagal.

No 2. Source Code

```
package main
```

```
import (
```

```
    "fmt"
```

```
    "strings"
```

```
)
```

```
func main() {
```

```
    var n int
```

```
    var bunga, pita string
```

```
    var count int
```

```
fmt.Print("Masukkan jumlah bunga (N): ")
```

```
fmt.Scan(&n)
```

```
if n == 0 {
```

```
    fmt.Println("Pita: ")
```

```
    fmt.Println("Bunga: 0")
```

```
    return
```

```
}
```

```
for count = 1; count <= n; count++ {
```

```
    fmt.Printf("Bunga %d: ", count)
```

```
    fmt.Scan(&bunga)
```

```
if strings.ToUpper(bunga) == "SELESAI" {
```

```
    count--
```

```
    break
```

```
}
```

```
if count > 1 {
```

```
    pita += " - "
```

```
}
```

```
pita += bunga
```

```
}
```

```

    fmt.Println("Pita:", pita)

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

```

Screenshoot Output

```

go run "/Users/ariffrach/Documents/go lang/guided.go"
ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Masukkan jumlah bunga (N): 5
Bunga 1: 2
Bunga 2: 4
Bunga 3: 5
Bunga 4: 2
Bunga 5: 4
Pita: 2 - 4 - 5 - 2 - 4
Bunga: 6
ariffrach@Arifs-MacBook-Air go lang %

```

Deskripsi Program

Program ini meminta pengguna untuk memasukkan sejumlah nama bunga, lalu menggabungkannya menjadi satu string yang dipisahkan oleh tanda hubung (" - "). Pengguna diminta untuk menentukan berapa banyak bunga (N) yang akan dimasukkan, tetapi proses input dapat dihentikan kapan saja dengan mengetikkan "SELESAI". Jika pengguna memasukkan 0 bunga, program akan langsung menampilkan pesan bahwa tidak ada bunga yang diinputkan. Pada akhir program, akan ditampilkan string bunga yang telah diinputkan beserta jumlah bunga yang dimasukkan sebelum berhenti.

No 3. Source Code

```
package main
```

```

import (
    "fmt"
    "math"
)

```

```
func main() {
```



```
var beratKantong1, beratKantong2, totalBerat float64
```

```
for {
```

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

```
    _, err := fmt.Scanf("%f %f", &beratKantong1, &beratKantong2)
```

```
    if err != nil {
```

```
        fmt.Println("Input tidak valid. Silakan coba lagi.")
```

```
        continue
```

```
    }
```

```
    if beratKantong1 < 0 || beratKantong2 < 0 {
```

```
        fmt.Println("Proses selesai.")
```

```
        break
```

```
    }
```

```
    totalBerat = beratKantong1 + beratKantong2
```

```
    if totalBerat > 150 {
```

```
        fmt.Println("Proses selesai.")
```

```
        break
```

```
    }
```

```
    selisih := math.Abs(beratKantong1 - beratKantong2)
```

```

    if selisih >= 9 {

        fmt.Println("Sepeda motor pak Andi akan oleng: true")

    } else {

        fmt.Println("Sepeda motor pak Andi akan oleng: false")

    }

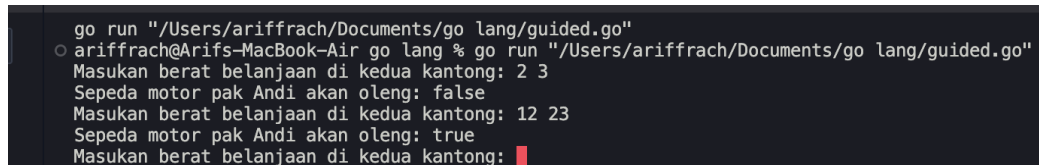
}

}

}

```

Screenshoot Output



```

go run "/Users/ariffrach/Documents/go lang/guided.go"
○ ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Masukan berat belanjaan di kedua kantong: 2 3
Sepeda motor pak Andi akan oleng: false
Masukan berat belanjaan di kedua kantong: 12 23
Sepeda motor pak Andi akan oleng: true
Masukan berat belanjaan di kedua kantong: 

```

Deskripsi Program

Program ini meminta pengguna memasukkan berat belanjaan di dua kantong. Jika selisih berat antara kedua kantong lebih dari 9 kg, program akan menyatakan bahwa sepeda motor Pak Andi akan oleng. Program berhenti jika total berat belanjaan lebih dari 150 kg atau jika ada berat yang negatif.

No 4. Source Code

```

package main

```

```

import (

    "fmt"

    "math"

)

```

```

func f(k int) float64 {

```

```

    numerator := math.Pow(float64(4*k+2), 2)

    denominator := float64((4*k + 1) * (4*k + 3))

    return numerator / denominator
}

```

```

func sqrt2Approximation(K int) float64 {
    product := 1.0

    for k := 0; k <= K; k++ {
        product *= f(k)
    }

    return product
}

```

```

func main() {
    var K int

    fmt.Print("Masukkan nilai K: ")

    fmt.Scan(&K)

    fmt.Printf("Nilai f(K) = %.10f\n", f(K))

    fmt.Print("Masukkan nilai K untuk akar 2: ")

    fmt.Scan(&K)

    fmt.Printf("Nilai akar 2 = %.10f\n", sqrt2Approximation(K))
}

```

Screenshoot Output

```
go run "/Users/ariffrach/Documents/go lang/guided.go"
● ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Masukkan nilai K: 33
Nilai f(K) = 1.0000556948
Masukkan nilai K untuk akar 2: 21
Nilai akar 2 = 1.4102022148
○ ariffrach@Arifs-MacBook-Air go lang %
```

Deskripsi Program

Program ini menghitung nilai fungsi matematis $f(K)$ dan memberikan perkiraan nilai akar kuadrat dari 2 menggunakan pendekatan produk matematika. Fungsi $f(k)$ menghitung sebuah nilai berdasarkan rumus yang melibatkan pangkat dan pembagian, di mana hasil dari fungsi ini digunakan dalam perhitungan berikutnya. Fungsi $\text{sqrt2Approximation}(K)$ menghitung perkiraan nilai akar 2 dengan mengalikan nilai-nilai dari fungsi $f(k)$ mulai dari $k=0$ hingga K . Pengguna diminta memasukkan nilai K dua kali, pertama untuk menghitung nilai $f(K)$, dan kedua untuk menghitung perkiraan akar 2.

2C

No 1. Source Code

```
package main
```

```
import "fmt"
```

```
func main() {
```

```
    var berat, kg, gr, biayaKirim, tambahanBiaya, totalBiaya int
```

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

```
    fmt.Scan(&berat)
```

```
    kg = berat / 1000
```

```
    gr = berat % 1000
```

```
    if kg > 10 {
```

```
        tambahanBiaya = 0
```

```
    } else if gr >= 500 {
```

```
        tambahanBiaya = gr * 5
```

```
    } else {
```

```
        tambahanBiaya = gr * 15
```

```
    }
```

```
    biayaKirim = kg * 10000
```

```
    totalBiaya = biayaKirim + tambahanBiaya
```

```

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

    fmt.Printf("Detail biaya: Rp. %d + Rp. %d\n", biayaKirim, tambahanBiaya)

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

```

Screenshoot Output

```

go run "/Users/ariffnach/Documents/go lang/guided.go"
● ariffnach@Arifs-MacBook-Air go lang % go run "/Users/ariffnach/Documents/go lang/guided.go"
Berat parcel (gram): 20
Detail berat: 0 kg + 20 gr
Detail biaya: Rp. 0 + Rp. 300
Total biaya: Rp. 300
○ ariffnach@Arifs-MacBook-Air go lang %

```

Deskripsi Program

Program ini menghitung biaya pengiriman parcel berdasarkan berat yang dimasukkan pengguna dalam gram. Berat tersebut diubah menjadi kilogram dan sisa gram. Jika berat parcel lebih dari 10 kg, tidak ada biaya tambahan; jika sisa gram 500 atau lebih, biaya tambahan dihitung dengan tarif 5 per gram; dan jika kurang dari 500 gram, tarifnya adalah 15 per gram. Biaya pengiriman dasar dihitung berdasarkan berat dalam kilogram dengan tarif 10.000 per kg. Program kemudian menjumlahkan biaya pengiriman dan biaya tambahan untuk menampilkan total biaya yang harus dibayar.

No 2. Source Code

```
package main
```

```
import "fmt"
```

```
func main() {
```

```
    var nam float64
```

```
    var nmk string
```

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

```

```

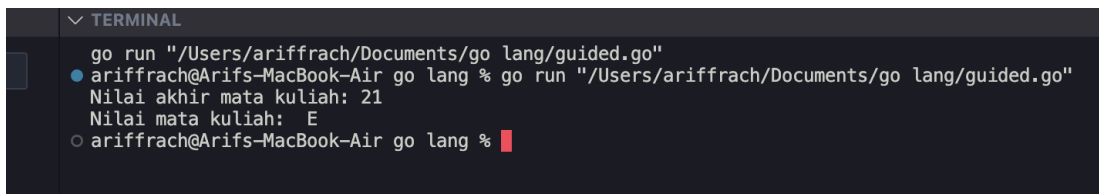
fmt.Scanln(&nam)

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)
}

```

Screenshoot Output



```

▼ TERMINAL
go run "/Users/ariffrach/Documents/go lang/guided.go"
● ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Nilai akhir mata kuliah: 21
Nilai mata kuliah: E
○ ariffrach@Arifs-MacBook-Air go lang %

```

Deskripsi Program

Program ini menentukan nilai huruf berdasarkan nilai akhir mata kuliah yang dimasukkan oleh pengguna. Pengguna diminta untuk memasukkan nilai akhir,

lalu program mengevaluasi nilai tersebut dengan menggunakan serangkaian kondisi. Jika nilai lebih dari 80, akan diberikan nilai "A"; jika lebih dari 72.5, "AB"; lebih dari 65, "B"; lebih dari 57.5, "BC"; lebih dari 50, "C"; lebih dari 40, "D"; dan jika tidak memenuhi syarat di atas, nilai akan menjadi "E". Program kemudian menampilkan nilai huruf yang sesuai dengan input pengguna.

No 3. Source Code

```
package main
```

```
import "fmt"
```

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

```
func isPrime(b int) bool {  
    factors := findFactors(b)  
    return len(factors) == 2  
}
```

```
func main() {
```



```

var b int

fmt.Print("Bilangan: ")

fmt.Scan(&b)

factors := findFactors(b)

fmt.Print("Faktor: ")

for _, factor := range factors {

    fmt.Printf("%d ", factor)

}

fmt.Println()

primeStatus := isPrime(b)

fmt.Printf("Prima: %t\n", primeStatus)

}

```

Screenshoot Output

```

go run "/Users/ariffrach/Documents/go lang/guided.go"
● ariffrach@Arifs-MacBook-Air go lang % go run "/Users/ariffrach/Documents/go lang/guided.go"
Bilangan: 23
Faktor: 1 23
Prima: true
○ ariffrach@Arifs-MacBook-Air go lang %

```

Deskripsi Program

Program ini berfungsi untuk menemukan faktor-faktor dari sebuah bilangan dan menentukan apakah bilangan tersebut adalah bilangan prima. Pengguna diminta untuk memasukkan sebuah bilangan. Program pertama-tama menggunakan fungsi `findFactors` untuk mencari semua faktor dari bilangan tersebut, yaitu angka-angka yang dapat membagi bilangan tersebut tanpa sisa. Hasil faktor-faktor ini kemudian ditampilkan. Selanjutnya, program memeriksa apakah bilangan tersebut adalah bilangan prima dengan menggunakan fungsi `isPrime`, yang memeriksa jumlah faktor yang ditemukan. Jika jumlah faktor hanya dua (1 dan bilangan itu sendiri),

maka bilangan tersebut dianggap prima. Program akan menampilkan hasil status prima dari bilangan yang dimasukkan.