

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL: 1
MATERI: REVIEW STRUKTUR KONTROL



DISUSUN OLEH:

NAMA: JESIKA METANIA RAHMA ARIFIN

NIM: 103112400080

KELAS: 12 IF 01

DOSEN:

Dimas Fanny Hebrisianto Permadi S.ST, M.Kom

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025/2026

1. DASAR TEORI

1. Perulangan

Perulangan adalah proses mengulang-ulang eksekusi blok kode tanpa henti, selama kondisi yang dijadikan acuan terpenuhi. Biasanya disiapkan variabel untuk iterasi atau variabel penanda kapan perulangan akan diberhentikan. Di Go keyword perulangan hanya **for** saja, tetapi meski demikian, kemampuannya merupakan gabungan for, foreach, dan while ibarat bahasa pemrograman lain. Ada beberapa cara standar menggunakan for. Cara pertama dengan memasukkan variabel counter perulangan beserta kondisinya setelah keyword.

```
for i := 0; i < 5; i++ {  
    fmt.Println("Angka", i)  
}
```

Perulangan di atas hanya akan berjalan ketika variabel *i* bernilai di bawah 5, dengan ketentuan setiap kali perulangan, nilai variabel *i* akan di-iterasi atau ditambahkan 1 (*i++* artinya ditambah satu, sama seperti *i = i + 1*). Karena *i* pada awalnya bernilai 0, maka perulangan akan berlangsung 5 kali, yaitu ketika *i* bernilai 0, 1, 2, 3, dan 4.

```
var i = 0  
  
for i < 5 {  
    fmt.Println("Angka", i)  
    i++  
}
```

```
var i = 0  
  
for {  
    fmt.Println("Angka", i)  
  
    i++  
    if i == 5 {  
        break  
    }  
}
```

```
for i := 0; i < 5; i++ {  
    for j := i; j < 5; j++ {  
        fmt.Print(j, " ")  
    }  
  
    fmt.Println()  
}
```

2. Percabangan

Seleksi kondisi digunakan untuk mengontrol alur eksekusi flow program. Analoginya mirip seperti fungsi rambu lalu lintas di jalan raya. Kapan kendaraan diperbolehkan melaju dan kapan harus berhenti diatur oleh rambu tersebut. Seleksi kondisi pada program juga kurang lebih sama, kapan sebuah blok kode dieksekusi dikontrol. Yang dijadikan acuan oleh seleksi kondisi adalah nilai bertipe bool, bisa berasal dari variabel, ataupun hasil operasi perbandingan. Nilai tersebut menentukan blok kode mana yang akan dieksekusi. Go memiliki 2 macam keyword untuk seleksi kondisi, yaitu **if else** dan **switch**.

Cara penerapan if-else di Go sama seperti pada bahasa pemrograman lain. Yang membedakan hanya tanda kurungnya (*parentheses*), di Go tidak perlu ditulis. Kode berikut merupakan contoh penerapan seleksi kondisi if else, dengan jumlah kondisi 4 buah.

```
var point = 8

if point == 10 {
    fmt.Println("lulus dengan nilai sempurna")
} else if point > 5 {
    fmt.Println("lulus")
} else if point == 4 {
    fmt.Println("hampir lulus")
} else {
    fmt.Printf("tidak lulus. nilai anda %d\n", point)
}
```

Variabel temporary adalah variabel yang hanya bisa digunakan pada deretan blok seleksi kondisi di mana ia ditempatkan. Penggunaan variabel ini membawa beberapa manfaat, antara lain:

- Scope atau cakupan variabel jelas, hanya bisa digunakan pada blok seleksi kondisi itu saja
- Kode menjadi lebih rapi
- Ketika nilai variabel tersebut didapat dari sebuah komputasi, perhitungan tidak perlu dilakukan di dalam blok masing-masing kondisi

```
var point = 8840.0

if percent := point / 100; percent >= 100 {
    fmt.Printf("%.1f%s perfect!\n", percent, "%")
} else if percent >= 70 {
    fmt.Printf("%.1f%s good\n", percent, "%")
} else {
    fmt.Printf("%.1f%s not bad\n", percent, "%")
}
```

Switch merupakan seleksi kondisi yang sifatnya fokus pada satu variabel, lalu kemudian di-cek nilainya. Contoh sederhananya seperti penentuan apakah nilai variabel x adalah: 1, 2, 3, atau lainnya.

```

var point = 6

switch point {
case 8:
    fmt.Println("perfect")
case 7:
    fmt.Println("awesome")
default:
    fmt.Println("not bad")
}

```

Pada kode di atas, tidak ada kondisi atau case yang terpenuhi karena nilai variabel point tetap 6. Ketika hal seperti ini terjadi, blok kondisi default dipanggil. Bisa dibilang bahwa default merupakan else dalam sebuah switch.

Perlu diketahui, switch pada pemrograman Go memiliki perbedaan dibanding bahasa lain. Di Go, ketika sebuah case terpenuhi, tidak akan dilanjutkan ke pengecekan case selanjutnya, meskipun tidak ada keyword break di situ. Konsep ini berkebalikan dengan switch pada umumnya pemrograman lain (yang ketika sebuah case terpenuhi, maka akan tetap dilanjut mengecek case selanjutnya kecuali ada keyword break).

2. GUIDED



```

package main

import "fmt"

func main() {
    for i := 1; i <= 5; i++ {
        fmt.Println("Iterasi ke-", i)
    }
}

```

SOURCE CODE:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER 2\MODUL 2\GuidedM2.go"
Iterasi ke- 1
Iterasi ke- 2
Iterasi ke- 3
Iterasi ke- 4
Iterasi ke- 5
PS C:\ALPRO SEMESTER 2>

```

Program di atas merupakan program yang menjalankan perulangan sebanyak 5 kali , mulai dari iterasi ke-1 sampai dengan iterasi ke-5.

```

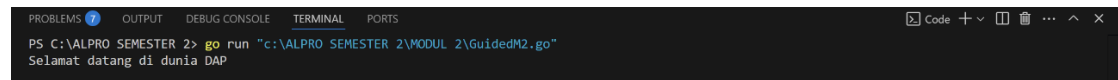
package main

import "fmt"

```

```
func main() {
    var greetings = "Selamat datang di dunia DAP"
    var a, b int
    fmt.Println(greetings)
    fmt.Scanln(&a, &b)
    fmt.Printf("%v + %v = %v\n", a, b, a+b)
}
```

SOURCE CODE:



```
PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER 2\MODUL 2\GuidedM2.go"
Selamat datang di dunia DAP
```

Program di atas merupakan program yang dibuat untuk menghasilkan output string yang berupa kalimat “ Selamat dating di dunia DAP”.

3. UNGUIDED

1.

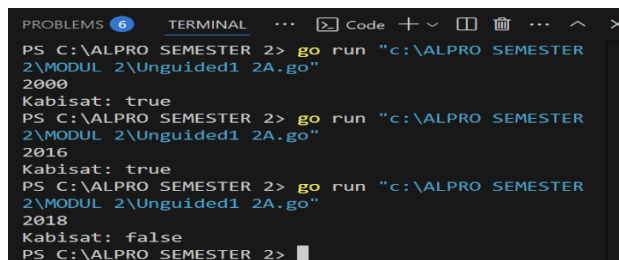
```
//JESIKA METANIA
package main

import "fmt"

func main() {
    var tahun int
    fmt.Scan(&tahun)

    if tahun%400 == 0 {
        fmt.Println("Kabisat: true")
    } else if tahun%100 == 0 {
        fmt.Println("Kabisat: false")
    } else if tahun%4 == 0 {
        fmt.Println("Kabisat: true")
    } else {
        fmt.Println("Kabisat: false")
    }
}
```

SOURCE CODE:



```
PROBLEMS 6 TERMINAL ... Code + - - - - -
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER 2\MODUL 2\Unguided1 2A.go"
2000
Kabisat: true
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER 2\MODUL 2\Unguided1 2A.go"
2016
Kabisat: true
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER 2\MODUL 2\Unguided1 2A.go"
2018
Kabisat: false
PS C:\ALPRO SEMESTER 2> 
```

Program di atas merupakan program yang di buat untuk menentukan “true” or “false” tahun kabisat.

2.

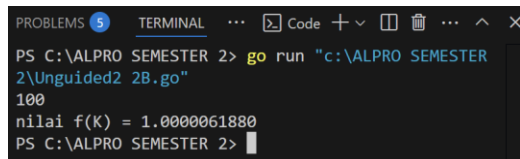
```
//JESIKA METANIA
package main

import "fmt"

func main() {
    var k float64
    fmt.Scan(&k)

    fK := (4*k+2) * (4*k+2) / ((4*k +1) * (4*k +3))
    fmt.Printf("nilai f(K) = %.10f\n", fK)
}
```

SOURCE CODE:



```
PROBLEMS 5 TERMINAL ... Code + - □ □ □ ... ^ ×
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER
2\Unguided2 28.go"
100
nilai f(K) = 1.0000061880
PS C:\ALPRO SEMESTER 2> |
```

MODIFIKASI :

```
// JESIKA METANIA
package main

import "fmt"

func main() {
    var k int
    fmt.Scan(&k)

    result := 1.0
    for i := 0; i <= k; i++ {
        pembilang := (4*i + 2) * (4*i + 2)
        penyebut := (4*i + 1) * (4*i + 3)
        result *= float64(pembilang) / float64(penyebut)
    }
    fmt.Printf("nilai √2 = %.10f\n", result)
}
```

SOURCE CODE:

```
PROBLEMS 6 TERMINAL ... Code + - □ □ ... ^ ×
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER
2\modif Unguided2 2B.go"
10
nilai √2 = 1.4062058441
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER
2\modif Unguided2 2B.go"
100
nilai √2 = 1.4133387072
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER
2\modif Unguided2 2B.go"
1000
nilai √2 = 1.4141252651
PS C:\ALPRO SEMESTER 2> █
```

Program di atas merupakan program untuk menghitung nilai $f(K)$ dengan nilai $f(K)$ diketahui merupakan bilangan decimal. Lalu program kedua diminta untuk memodifikasi code untuk mencari hasil nilai dari $f(K)$ jika diketahui jika $f(K)$ adalah $\sqrt{2}$

3.

```
// JESIKA METANIA R.A
package main

import "fmt"

func main() {
    var berat int
    fmt.Print("Berat parsel(gram): ")
    fmt.Scan(&berat)

    kg := berat / 1000
    sisa_gram := berat % 1000

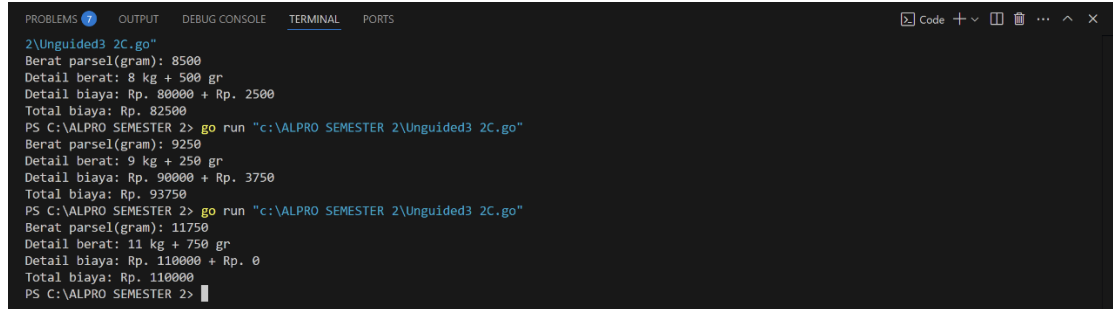
    biaya := kg * 10000
    biaya_tambahan := 0

    switch {
    case sisa_gram == 0:
        biaya_tambahan = 0
    case sisa_gram >= 500:
        biaya_tambahan = sisa_gram * 5
    case sisa_gram < 500:
        biaya_tambahan = sisa_gram * 15
    }
    if kg >= 10 {
        biaya_tambahan = 0
    }

    totalBiaya := biaya + biaya_tambahan
}
```

```
fmt.Println("Detail berat:", kg, "kg +", sisa_gram, "gr")
fmt.Println("Detail biaya: Rp.", biaya, "+ Rp.", biaya_tambahan)
fmt.Println("Total biaya: Rp.", totalBiaya)
}
```

SOURCE CODE:

A screenshot of a terminal window showing the execution of a Go program. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The output shows three runs of the program. Each run takes a weight in grams as input and calculates the total cost based on a base price per kg and a price for the remaining grams. The first run takes 8500g and outputs a total cost of Rp. 82500. The second run takes 9250g and outputs a total cost of Rp. 93750. The third run takes 11750g and outputs a total cost of Rp. 110000.

```
2\Unguided3 2C.go"
Berat parcel(gram): 8500
Detail berat: 8 kg + 500 gr
Detail biaya: Rp. 80000 + Rp. 2500
Total biaya: Rp. 82500
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER 2\Unguided3 2C.go"
Berat parcel(gram): 9250
Detail berat: 9 kg + 250 gr
Detail biaya: Rp. 90000 + Rp. 3750
Total biaya: Rp. 93750
PS C:\ALPRO SEMESTER 2> go run "c:\ALPRO SEMESTER 2\Unguided3 2C.go"
Berat parcel(gram): 11750
Detail berat: 11 kg + 750 gr
Detail biaya: Rp. 110000 + Rp. 0
Total biaya: Rp. 110000
PS C:\ALPRO SEMESTER 2> |
```

Program ini dibuat untuk menghitung total biaya, jika diketahui berat parcel nya.

4. KESIMPULAN:

Kombinasi keduanya sangat penting untuk membuat program yang efisien, dinamis, dan fleksibel sesuai dengan kebutuhan pengguna. Perulangan membantu dalam otomatisasi tugas yang berulang, sedangkan percabangan memungkinkan pengambilan keputusan dalam program.

5. REFERENSI

<https://dasarpemrogramangolang.novalagung.com/A-perulangan.html>