

**LAPORAN PRAKTIKUM**  
**ALGORITMA DAN PEMROGRAMAN 1**  
**MODUL 12**  
**“WHILE-LOOP”**



**DISUSUN OLEH:**  
**ANGGUN WAHYU WIDIYANA**  
**103112480280**  
**S1 IF-12-01**  
**DOSEN:**  
**Yohani Setiya Rafika Nur, M. Kom.**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2024/2025**

## DASAR TEORI

### Paradigma Percabangan

Perulangan merupakan salah satu struktur kontrol yang memungkinkan suatu instruksi yang sama dilakukan berulang kali dalam waktu atau jumlah yang lama. Tanpa instruksi perulangan, maka suatu instruksi akan ditulis dalam jumlah yang sangat banyak. Sebelumnya pada modul ke-5 dan 6 telah dipelajari instruksi perulangan dengan for-loop. Instruksi for-loop memungkinkan kita melakukan berulang sebanyak  $n$  iterasi, akan tetapi pada banyak kasus yang melibatkan perulangan, **tidak semua perulangan diketahui jumlah iterasinya di awal**. Perulangan seperti ini disebut juga dengan istilah **perulangan dengan kondisi**.

**Penting!** Pastikan bahwa instruksi perulangan yang digunakan pasti bisa membuat proses perulangan berhenti, apabila tidak maka program akan terus berjalan mengulangi instruksi tanpa akan pernah berhenti.

Terdapat dua bentuk perulangan dengan kondisi ini, yaitu **while-loop** dan **repeat-until**.

### Karakteristik While-Loop

Struktur kontrol perulangan menggunakan while-loop memiliki bentuk yang hampir serupa dengan penulisan if-then pada percabangan, yaitu memiliki kondisi dan aksi. Hal yang membedakan adalah aksi akan dilakukan secara berulang-ulang selama kondisi bernilai true.

- a) **Kondisi**, merupakan nilai atau operasi tipe data yang menghasilkan tipe data boolean.

Kondisi ini merupakan syarat terjadinya perulangan. Artinya perulangan terjadi apabila kondisi bernilai true operasi tipe data yang menghasilkan **nilai** selain tipe data boolean,

- b) **Aksi**, merupakan kumpulan instruksi yang akan dieksekusi secara berulang-ulang selama kondisi bernilai true. Salah satu instruksi dari aksi harus bisa membuat kondisi yang awalnya bernilai true menjadi false, tujuannya adalah untuk membuat perulangan berhenti.

Pada penulisan notasinya secara umum bahasa pemrograman menggunakan kata kunci while, tetapi khusus di bahasa pemrograman Go, kata kunci yang digunakan adalah for. Walaupun berbeda dari kata kunci yang digunakan, secara struktur penulisannya tetap sama, sehingga tetap mudah untuk membedakan instruksi for yang digunakan adalah for-loop atau while-loop.

Berikut adalah notasi dari while-loop:

Notasi dalam pseduecode	Notasi dalam bahasa Go
<pre>while kondisi do     // aksi endwhile</pre>	<pre>for kondisi {     // aksi }</pre>

## Implementasi menggunakan Go

Contoh persoalan yang melibatkan perulangan dengan kondisi ini adalah: Program yang akan menampilkan hasil penjumlahan dua bilangan yang diterima dari masukan pengguna secara terus menerus. Proses ini akan berlangsung terus selama hasil penjumlahan adalah genap.

```
1 // filename: whileloop1.go
2 package main
3 import "fmt"
4
5 func main() {
6     var a int
7     var b int
8     fmt.Scan(&a, &b)
9     for (a + b) % 2 == 0 {
10         fmt.Println("Hasil penjumlahan", a+b)
11         fmt.Scan(&a, &b)
12     }
13     fmt.Println("Program selesai")
14 }
```

```
C:\users\go\src\hello>go build whileloop1.go
```

```
C:\users\go\src\hello>whileloop1
```

```
10 6
```

```
Hasil penjumlahan 16
```

```
5 5
```

```
Hasil penjumlahan 10
```

```
3 9
```

```
Hasil penjumlahan 12
```

```
9 2
```

```
Program selesai
```

```
C:\users\go\src\hello>whileloop1
```

```
4 5
```

```
Program selesai
```

Pada contoh di atas, instruksi di baris ke-10 dan 11 akan berhenti dieksekusi berulang ketika kondisi di baris ke-9 bernilai false, yaitu ketika a dan b bernilai 9 dan 2 (  $9 + 2 = 11$ , yaitu ganjil).

Contoh penggunaan bentuk while-loop untuk menghitung  $y = \sqrt{x}$  adalah sebagai berikut:

Notasi algoritma		Penulisan dalam bahasa
1	e <- 0.0000001	e = 0.0000001
2	x <- 2.0	x = 2.0
3	y <- 0.0	y = 0.0
4	y1 <- x	y1 = x
5	<b>while</b> y1-y > e <b>or</b> y1-y < -e <b>do</b>	<b>for</b> y1-y > e    y1-y < -e {
6	y <- y1	y = y1
7	y1 <- 0.5*y + 0.5*(x/y)	y1 = 0.5*y + 0.5*(x/y)
8	<b>endwhile</b>	}
9	<b>output</b> ("sqrt(", x, ")=", y)	fmt.Printf("sqrt(%v)=%v\n", x, y)

## CONTOH SOAL

### Latihan1

Buatlah program yang digunakan untuk menampilkan deret bilangan Faktorial dari suatu bilangan.

**Masukan** terdiri dari sebuah bilangan bulat non negatif  $n$ .

**Keluaran** berupa deret bilangan dari Faktorial  $n$ . Perhatikan contoh masukan dan keluaran yang diberikan.

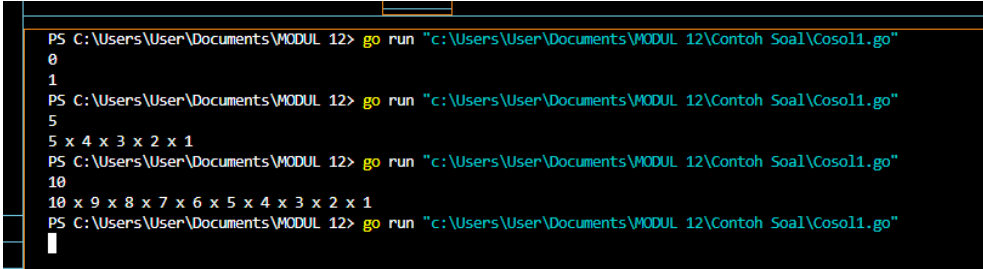
**Contoh masukan dan keluaran:**

No	Masukan	Keluaran
1	0	1
2	5	5 x 4 x 3 x 2 x 1
3	10	10 x 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1
4	1	1

Source Code:

```
package main
import "fmt"
func main(){
    var n, j int
    fmt.Scan(&n)
    j = n
    for j > 1 {
        fmt.Print(j, " x ")
        j = j - 1
    }
    fmt.Println(1)
}
```

Output:



```
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Cosol1.go"
0
1
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Cosol1.go"
5
5 x 4 x 3 x 2 x 1
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Cosol1.go"
10
10 x 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Cosol1.go"
```

Deskripsi Program:

Program Go ini bertujuan untuk untuk menampilkan deret bilangan faktorial dari sebuah bilangan bulat non-negatif yang dimasukkan oleh pengguna. Setelah pengguna memasukkan nilai  $n$ , program akan mencetak deret angka mulai dari  $n$  hingga 1, dipisahkan oleh tanda "x".

## Latihan2

Buatlah program Go yang digunakan untuk login ke dalam suatu aplikasi. Asumsi token untuk yang valid adalah "12345abcde".

**Masukan** terdiri dari suatu token. Selama token yang diberikan salah, maka program akan meminta token secara terus menerus hingga token yang diberikan benar.

**Keluaran** adalah teks yang menyatakan "Selamat Anda berhasil login".

**Contoh masukan dan keluaran:**

No	Masukan	Keluaran
1	Qwe12312 231234 13213 123lijwe 12345abcde	Selamat Anda berhasil login
2	12345abcde	Selamat Anda berhasil login

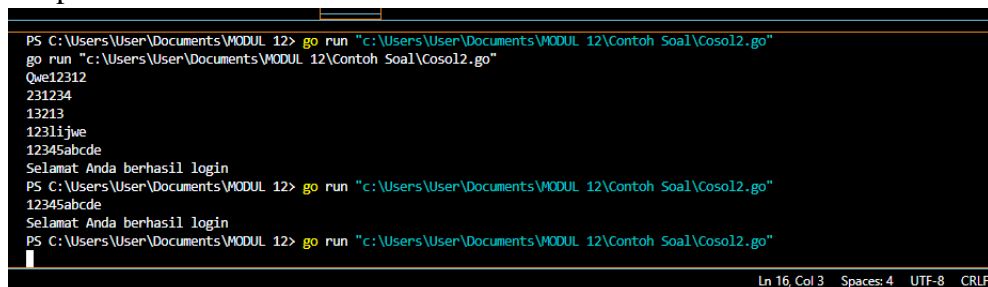
Source Code:

```
package main

import "fmt"

func main(){
    var token string
    fmt.Scan(&token)
    for token != "12345abcde" {
        fmt.Scan(&token)
    }
    fmt.Println("Selamat Anda berhasil login")
}
```

Output:



```
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Coso12.go"
go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Coso12.go"
Qwe12312
231234
13213
123lijwe
12345abcde
Selamat Anda berhasil login
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Coso12.go"
12345abcde
Selamat Anda berhasil login
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Coso12.go"
```

Deskripsi Program:

Program ini adalah sistem login sederhana yang meminta pengguna memasukkan token "12345abcde". Jika token salah, program terus meminta input hingga benar. Setelah berhasil, program menampilkan pesan "Selamat Anda berhasil login".

### Latihan3

Buatlah program dalam bahasa Go yang digunakan untuk mencetak N bilangan pertama dalam deret Fibonacci.

Masukan terdiri dari bilangan bulat positif N dengan nilai besar atau sama dengan 2.

Keluaran terdiri dari sejumlah N bilangan yang menyatakan N deret bilangan Fibonacci yang pertama.

#### Contoh masukan dan keluaran

No	Masukan	Keluaran
1	5	0 1 1 2 3
2	2	0 1
3	10	0 1 1 2 3 5 8 13 21 34

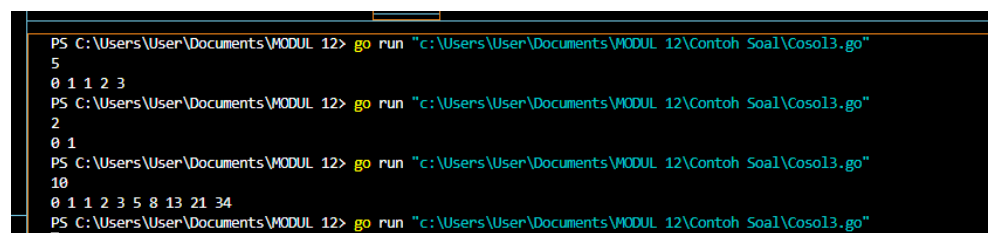
#### Source Code:

```
package main

import "fmt"

func main(){
    var N, s1, s2, j, temp int
    fmt.Scan(&N)
    s1 = 0
    s2 = 1
    j = 0
    for j < N {
        fmt.Print(s1, " ")
        temp = s1 + s2
        s1 = s2
        s2 = temp
        j = j + 1
    }
}
```

#### Output:



```
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Cosol3.go"
5
0 1 1 2 3
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Cosol3.go"
2
0 1
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Cosol3.go"
10
0 1 1 2 3 5 8 13 21 34
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Contoh Soal\Cosol3.go"
```

#### Deskripsi Program:

Program Go ini berfungsi untuk Program ini ditulis dalam bahasa pemrograman Go dan berfungsi untuk mencetak N bilangan pertama dalam deret Fibonacci. Pengguna diminta untuk memasukkan bilangan bulat positif

N dengan nilai minimal 2. Program kemudian menghitung dan mencetak deret Fibonacci mulai dari 0 dan 1, di mana setiap bilangan berikutnya merupakan penjumlahan dari dua bilangan sebelumnya.

Program ini menggunakan variabel s1 dan s2 untuk menyimpan dua bilangan terakhir dalam deret Fibonacci, dan variabel j untuk menghitung jumlah bilangan yang telah dicetak. Setelah N bilangan dicetak, program selesai.

---

## SOAL LATIHAN

### Latihan1

Buatlah sebuah program dalam bahasa Go yang digunakan untuk menghitung berapa banyak seseorang pengguna gagal melakukan login, karena kesalahan memberikan username dan password.

**Masukan** terdiri dari dua teks yang berisi username dan password, apabila username dan password salah, maka program akan meminta masukan ulang. Apabila username dan password sudah benar maka program akan menampilkan informasi berapa kali percobaan login yang gagal dilakukan. Asumsi username dan password yang benar adalah "Admin" dan "Admin" tanpa tanda petik.

### Contoh masukan dan keluaran

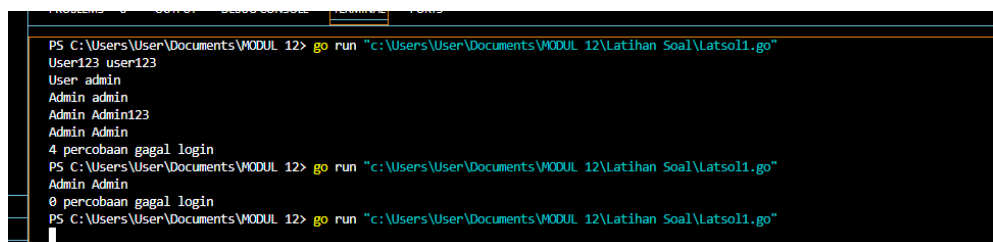
No	Masukan	Keluaran
1	User123 user123 User admin Admin admin Admin Admin123 Admin Admin	4 percobaan gagal login
2	Admin Admin	0 percobaan gagal login

### Source Code:

```
package main
import "fmt"
func main(){
    var username, password string
    perc_gagal := 0

    for {
        fmt.Scan(&username, &password)
        if username == "Admin" && password == "Admin" {
            break
        }
        perc_gagal++
    }
    fmt.Println(perc_gagal, "percobaan gagal login")
}
```

### Output:



```
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latsol1.go"
User123 user123
User admin
Admin admin
Admin Admin123
Admin Admin
4 percobaan gagal login
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latsol1.go"
Admin Admin
0 percobaan gagal login
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latsol1.go"
```



### Deskripsi Program:

Program Go ini menghitung jumlah percobaan gagal login pengguna. Pengguna diminta untuk memasukkan username dan password. Jika username dan password salah, program akan meminta masukan ulang. Ketika pengguna berhasil memasukkan username dan password yang benar, yaitu "Admin" dan "Admin", program akan menampilkan jumlah percobaan gagal login yang dilakukan. Variabel `perc_gagal` digunakan untuk menyimpan jumlah percobaan yang gagal, yang akan diincrement setiap kali input salah. Program ini berfungsi dalam loop hingga login berhasil.

### Latihan2

Buatlah program yang digunakan untuk mencacah setiap digit yang terdapat di dalam suatu bilangan bulat positif.

**Masukan** terdiri dari suatu bilangan bulat positif.

**Keluaran** terdiri dari nilai digit pada bilangan tersebut. Tampilkan dari digit terakhir (paling kanan) sampai dengan digit pertama (paling kiri).

### Contoh masukan dan keluaran

No	Masukan	Keluaran
1	2	2
2	2544	4 4 5 2
3	3423554654	4 5 6 4 5 5 3 2 4 3

### Source Code:

```
package main
import "fmt"
func main(){
    var bilangan, digit int
    fmt.Scan(&bilangan)

    for bilangan > 0 {
        digit = bilangan % 10
```

```
        fmt.Println(digit)
        bilangan = bilangan / 10
    }
}
```

Output:

```
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latso12.go"
2
2
2544
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latso12.go"
4
4
5
3423554654
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latso12.go"
4
5
6
4
5
5
3
2
4
3
3
```

Deskripsi Program:

Program Go berfungsi untuk mencacah setiap digit dari suatu bilangan bulat positif. Pengguna diminta untuk memasukkan bilangan bulat positif, yang kemudian disimpan dalam variabel bilangan. Program menggunakan loop untuk mengambil digit terakhir dari bilangan dengan operasi modulus (%) dan menampilkannya. Setelah menampilkan digit, program menghapus digit terakhir dengan membagi bilangan tersebut dengan 10. Proses ini diulang hingga semua digit ditampilkan dari yang terakhir (paling kanan) ke yang pertama (paling kiri). Program ini menghasilkan output yang mencetak setiap digit pada baris terpisah.

### Latihan3

Buatlah program untuk mencari hasil integer division dari dua bilangan. Gunakan perulangan dan tidak diperbolehkan menggunakan operator pembagian.

**Masukan** terdiri dari dua bilangan bulat positif  $x$  dan  $y$ , yang mana  $x \geq y$ .

**Keluaran** terdiri dari hasil dari operasi  $x$  div dengan  $y$ .

**Contoh masukan dan keluaran:**

No	Masukan	Keluaran
1	5 2	2
2	10 7	1
3	120	30

#### Source Code:

```
package main
import "fmt"
func main(){
    var x, y, hasil int
    fmt.Scan(&x, &y)

    if x < y {
        fmt.Println(0)
        return
    }

    for x >= y {
        x -= y
        hasil++
    }

    fmt.Println(hasil)
}
```

#### Output:

```
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latsol3.go"
5 2
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latsol3.go"
10 7
1
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latsol3.go"
120 4
30
PS C:\Users\User\Documents\MODUL 12> go run "c:\Users\User\Documents\MODUL 12\Latihan Soal\Latsol3.go"
```

#### Deskripsi Program:

Program Go ini untuk menghitung hasil integer division dari dua bilangan bulat positif x dan y tanpa menggunakan operator pembagian. Pengguna diminta untuk memasukkan dua bilangan, dan program memeriksa apakah x lebih besar atau sama dengan y. Jika tidak, program akan mencetak 0.

Program kemudian menggunakan perulangan untuk mengurangi x dengan y secara berulang, sambil menghitung berapa kali pengurangan tersebut dilakukan. Hasil dari penghitungan ini disimpan dalam variabel hasil, yang kemudian dicetak sebagai output.

## **DAFTAR PUSTAKA**

School of Computing. *Modul Praktikum 12 – While-Loop Algoritma dan Pemrograman 1 SI Informatika*.2024