

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 1
MODUL XIII
“REPEAT-UNTIL”



Disusun oleh :
NAMA : Felix Pedrosa Valentino
NIM : 103112400056
S1 IF – 12 - 01

Dosen Pengampu :
Yohani Setiya Rafika Nur, M. Kom.

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

DASAR TEORI

Paradigma Perulangan

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. Pada modul 12 sebelumnya telah dipelajari terkait penggunaan struktur kontrol perulangan dengan while-loop, selanjutnya perulangan juga dapat dilakukan menggunakan repeat-until.

Penggunaan repeat-until pada dasarnya sama dengan while-loop di mana perulangan berdasarkan kondisi. Perbedaan terletak pada kondisi yang digunakan, pada while-loop kondisi yang harus didefinisikan adalah kondisi perulangannya, atau kapan perulangan itu terjadi, sedangkan pada repeat-until kondisi yang harus didefinisikan merupakan kondisi berhenti, atau kapan perulangan tersebut harus dihentikan. Kondisi perulangan dan kondisi berhenti memiliki keterhubungan sifat komplemen, sehingga apabila kita mengetahui kondisi perulangannya, maka cukup dengan menambahkan operator negasi atau not untuk mengubah menjadi kondisi berhenti. Hal ini berlaku juga sebaliknya, komplemen dari kondisi berhenti adalah kondisi perulangan.

Pahami beberapa contoh yang diberikan berikut ini:

- Statement while-loop: *"Menulis teks tertentu selama tinta pena masih ada"*.

Statement repeat-until: *"Menulis teks tertentu sampai tinta pena habis"*.

Komplemen dari kondisi "tinta pena masih ada" adalah "tinta pena habis".

- Statement while-loop: *"Saya makan suap demi suap selama saya masih lapar"*.

Statement repeat-until: *"Saya makan suap demi suap sampai saya merasa kenyang"*.

Komplemen dari kondisi "saya masih lapar" adalah. *"saya merasa kenyang"*.

Bentuk Repeat-Until

- Berbeda dengan while-loop, di mana <action> akan dieksekusi secara berulang sampai <condition> bernilai true.

- Jumlah iterasi tidak dapat ditentukan, karena bergantung dengan perubahan nilai pada <condition>.
- Pada bahasa Go, tidak ada penulisan secara spesifik untuk repeat-until, jadi alternatifnya bisa menggunakan bentuk while-loop.

Karakteristik Repeat-Until

Komponen dari repeat-until sama dengan while-loop, yaitu terdapat kondisi dan aksi, hanya struktur penulisannya saja yang berbeda.

1) Aksi, merupakan kumpulan instruksi yang akan dilakukan perulangan. Aksi minimal dijalankan sekali, baru dilakukan pengecekan kondisi berhenti setelahnya. Apabila kondisi bernilai true, maka perulangan dihentikan.

2) Kondisi/berhenti, merupakan kondisi berhenti dari perulangan, harus bernilai false selama perulangan dilakukan.

Notasi repeat-until memiliki banyak sekali keragaman kata kunci di dalam bahasa

pemrograman. Penggunaan repeat-until sebenarnya berasal dari keluarga bahasa pemrograman Pascal. Pada keluarga bahasa pemrograman C/C++ digunakan do-while, sedangkan pada bahasa Go tidak ada instruksi eksplisit untuk repeat-until.

CONTOH SOAL

1.) Contoh Soal 1

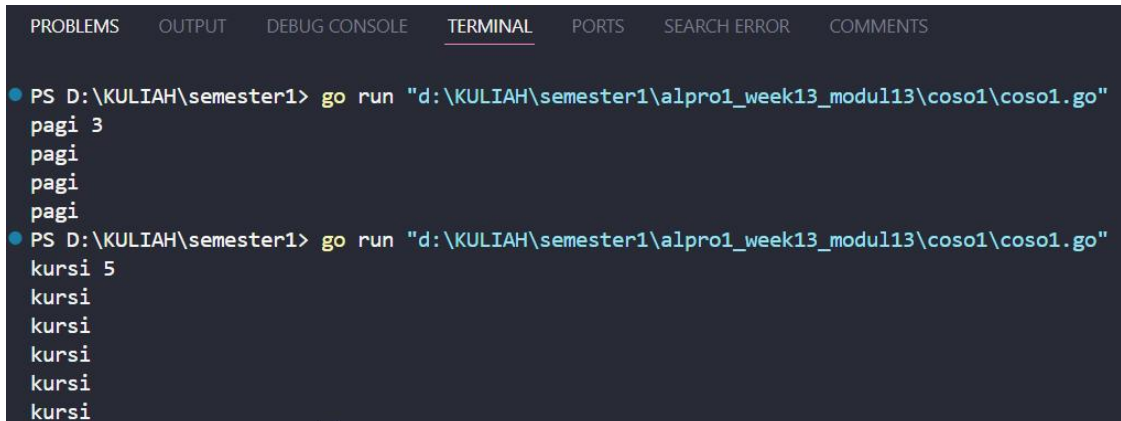
Source Code :

```
package main

import "fmt"

func main() {
    var word string
    var repetitions int
    fmt.Scan(&word, &repetitions)
    counter := 0
    for done := false; !done; {
        fmt.Println(word)
        counter++
        done = (counter >= repetitions)
    }
}
```

Output :



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR  COMMENTS

● PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\coso1\coso1.go"
pagi 3
pagi
pagi
pagi
● PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\coso1\coso1.go"
kursi 5
kursi
kursi
kursi
kursi
kursi
```

Deskripsi Program :

Program di atas berfungsi untuk mencetak sebuah kata yang dimasukkan oleh pengguna sebanyak jumlah pengulangan yang ditentukan. Program dimulai dengan mendeklarasikan dua variabel; variabel *word* bertipe *string* untuk menyimpan kata yang akan dicetak dan variabel *repetitions* bertipe *integer* untuk menyimpan jumlah pengulangan. Setelah itu, program meminta pengguna untuk memasukkan nilai untuk kedua variabel tersebut menggunakan *fmt.Scan*. Selanjutnya, program menginisialisasi variabel *counter* dengan nilai 0, yang akan digunakan untuk menghitung berapa kali kata telah dicetak. Program kemudian memasuki sebuah *loop* yang akan terus berjalan selama kondisi *done* bernilai *false*. Di dalam *loop*, program mencetak kata yang disimpan dalam variabel *word* dan kemudian meningkatkan nilai *counter* sebanyak 1. Setelah mencetak, program memeriksa apakah *counter* telah mencapai atau melebihi nilai *repetitions*. Jika ya, maka *done* akan diatur menjadi *true*, yang akan menghentikan *loop*.

2.) Contoh Soal 2

Source Code :

```
package main

import "fmt"

func main() {
    var number int
    var continueLoop bool
    for continueLoop = true; continueLoop; {
        fmt.Scan(&number)
        continueLoop = number <= 0
    }
    fmt.Printf("%d adalah bilangan bulat positif\n", number)
}
```

Output :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR  COMMENTS

PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\coso2\coso2.go"
-5
-2
-1
0
5
5 adalah bilangan bulat positif
PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\coso2\coso2.go"
17
17 adalah bilangan bulat positif
```

Deskripsi Program :

Program di atas berfungsi untuk meminta pengguna memasukkan sebuah bilangan bulat positif. Program dimulai dengan mendeklarasikan variabel `number` bertipe `integer` untuk menyimpan input dari pengguna dan variabel `continueLoop` bertipe `boolean` sebagai penanda untuk mengontrol perulangan. Program kemudian memasuki sebuah loop yang akan terus berjalan selama `continueLoop` bernilai `true`. Di dalam loop, program meminta pengguna untuk memasukkan sebuah bilangan bulat dengan menggunakan `fmt.Scan`. Setelah menerima input, program memeriksa apakah nilai `number` yang dimasukkan kurang dari atau sama dengan 0. Jika kondisi tersebut terpenuhi, `continueLoop` akan tetap bernilai `true`, dan loop akan meminta input lagi. Sebaliknya, jika pengguna memasukkan bilangan bulat positif, `continueLoop` akan diatur menjadi `false`, yang akan menghentikan loop. Setelah keluar dari loop, program mencetak pesan yang menyatakan bahwa bilangan yang dimasukkan adalah bilangan bulat positif.

3.) Contoh Soal 3

Source Code :

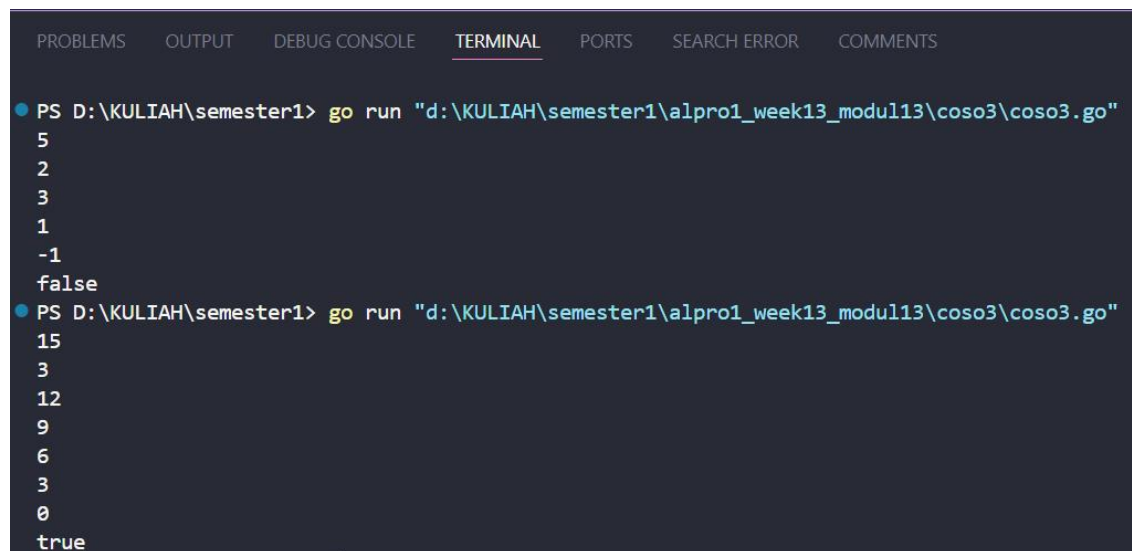
```
package main

import "fmt"

func main() {
    var x int
    var y int
    var selesai bool
    fmt.Scan(&x, &y)
```

```
for selesai = false; !selesai; {  
    x = x - y  
    fmt.Println(x)  
    selesai = x <= 0  
}  
fmt.Println(x == 0)  
}
```

Output :



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR  COMMENTS  
● PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\coso3\coso3.go"  
5  
2  
3  
1  
-1  
false  
● PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\coso3\coso3.go"  
15  
3  
12  
9  
6  
3  
0  
true
```

Deskripsi Program :

Program di atas berfungsi untuk melakukan pengurangan berulang antara dua bilangan bulat yang dimasukkan oleh pengguna, yaitu x dan y. Program dimulai dengan mendeklarasikan dua variabel x dan y bertipe integer untuk menyimpan input dari pengguna, serta variabel selesai bertipe boolean yang digunakan untuk mengontrol perulangan. Setelah meminta pengguna untuk memasukkan nilai untuk x dan y menggunakan fmt.Scan, program memasuki sebuah loop yang akan terus berjalan selama selesai bernilai false. Di dalam loop, program mengurangi nilai x dengan y dan mencetak hasil pengurangan tersebut. Kemudian, program memeriksa apakah nilai x sudah kurang dari atau sama dengan 0. Jika ya, maka selesai akan diatur menjadi true, yang akan menghentikan loop. Setelah keluar dari loop, program mencetak hasil evaluasi dari ekspresi `x == 0`, yang akan menghasilkan true jika x sama dengan 0 dan false jika tidak.

Latihan Soal

1.) Latihan Soal 1

Source Code :

```
package main

import "fmt"

func main() {
    var bilangan int
    fmt.Scan(&bilangan)
    jumlahDigit := 0

    for {
        jumlahDigit++
        bilangan /= 10
        if bilangan == 0 {
            break
        }
    }
    fmt.Print(jumlahDigit)
}
```

Output :


```

PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\latsoal1\latsoal1.go"
5
1
PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\latsoal1\latsoal1.go"
234
3
PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\latsoal1\latsoal1.go"
78787
5
PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\latsoal1\latsoal1.go"
1894256
7

```

Deskripsi Program :

Program di atas berfungsi untuk menghitung jumlah digit dari sebuah bilangan bulat yang diinputkan oleh pengguna. Pertama, program mendeklarasikan sebuah variabel bilangan bertipe integer untuk menyimpan input dari pengguna. Dengan menggunakan fungsi `fmt.Scan`, program meminta pengguna untuk memasukkan sebuah bilangan. Selanjutnya, program menginisialisasi variabel `jumlahDigit` dengan nilai 0, yang akan digunakan untuk menghitung jumlah digit. Program kemudian memasuki sebuah loop tak terbatas yang akan terus berjalan hingga kondisi tertentu terpenuhi. Di dalam loop, setiap iterasi akan menambah nilai `jumlahDigit` sebesar 1 dan membagi bilangan dengan 10 untuk menghilangkan digit terakhir. Jika setelah pembagian bilangan menjadi 0, loop akan dihentikan dengan perintah `break`. Akhirnya, program mencetak jumlah digit yang telah dihitung ke layar.

2.) Latihan Soal 2

Source Code :

```

package main

import "fmt"

func main() {
    var number, jumlah float64
    fmt.Scan(&number)
    jumlah = number + 0.1
    for jumlah <= float64(int(number)+1) {
        fmt.Printf("%.1f\n", jumlah)
        jumlah += 0.1
    }
}

```

```
}  
}
```

Output :

```
PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\latsoal2\latsoal2.go"  
0.2  
0.3  
0.4  
0.5  
0.6  
0.7  
0.8  
0.9  
1.0
```

Deskripsi Program :

Program ini berfungsi untuk membaca sebuah angka desimal (float) dari input pengguna, kemudian mencetak angka-angka desimal dengan kenaikan 0.1 hingga mencapai nilai integer yang lebih besar dari angka awal tersebut.

Program dimulai dengan mendeklarasikan dua variabel bertipe float64, yaitu *number* untuk menampung input dari pengguna dan *jumlah* yang akan digunakan sebagai variabel iterasi. Setelah pengguna memasukkan angka desimal melalui fungsi `fmt.Scan`, program menghitung nilai awal *jumlah* dengan menambahkan 0.1 pada angka input. Program kemudian menggunakan perulangan `for` untuk mencetak nilai *jumlah* dengan format desimal satu angka di belakang koma (`%.1f`). Iterasi berlangsung dengan menambahkan 0.1 ke *jumlah* di setiap langkah hingga nilainya melebihi bilangan bulat terbesar yang lebih kecil atau sama dengan input awal yang ditambah satu.

3.) Latihan Soal 3

Source Code :

```
package main  
  
import "fmt"  
  
func main() {  
    var target, perdonasi, donatur, akumulasi int
```

```

    fmt.Scan(&target)
    for akumulasi = perdonasi; akumulasi < target; {
        fmt.Scan(&perdonasi)
        akumulasi += perdonasi
        donatur++
        fmt.Printf("Donatur  %d: Menyumbang  %d. Total
    terkumpul: %d\n", donatur, perdonasi, akumulasi)
    }
    fmt.Printf("Target tercapai! Total donasi: %d dari %d donatur.",
    akumulasi, donatur)
}

```

Output :

```

PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\latsoal3\latsoal3.go"
300
100
Donatur 1: Menyumbang 100. Total terkumpul: 100
50
Donatur 2: Menyumbang 50. Total terkumpul: 150
200
Donatur 3: Menyumbang 200. Total terkumpul: 350
Target tercapai! Total donasi: 350 dari 3 donatur.
PS D:\KULIAH\semester1> go run "d:\KULIAH\semester1\alpro1_week13_modul13\latsoal3\latsoal3.go"
200
300
Donatur 1: Menyumbang 300. Total terkumpul: 300
Target tercapai! Total donasi: 300 dari 1 donatur.

```

Deskripsi Program :

Program ini berfungsi untuk menghitung banyak donasi hingga mencapai target tertentu. Dalam program ini, pengguna diminta untuk memasukkan jumlah target donasi yang ingin dicapai. Setelah itu, program akan terus meminta input dari pengguna berupa jumlah donasi yang diberikan oleh setiap donatur. Setiap kali donatur menyumbang, program akan mencatat jumlah sumbangan tersebut dan menampilkan informasi tentang donatur, termasuk nomor urut donatur, jumlah sumbangan yang diberikan, dan total akumulasi donasi yang telah terkumpul hingga saat itu. Proses ini akan berlanjut hingga total akumulasi donasi mencapai atau melebihi target yang telah ditentukan. Setelah target tercapai, program akan menampilkan pesan yang menyatakan bahwa target telah tercapai, beserta total donasi yang terkumpul dan jumlah donatur yang berpartisipasi.

DAFTAR PUSTAKA

Prasti Eko Yunanto, S.T., M.Kom. (2024). MODUL PRAKTIKUM 13 –
REPEAT-UNTIL
ALGORITMA DAN PEMROGRAMAN 1 S1 INFORMATIKA (MODUL 13)