

**LAPORAN PRAKTIKUM**  
**ALGORITMA DAN PEMROGRAMAN 1**  
**MODUL 13**  
**“REPEAT UNTIL”**



**DISUSUN OLEH:**  
**SAVILA NUR FADILLA**  
**103112400031**  
**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

### Karakteristik Repeat-Until

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

- a.) 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.
- b.) Kondisi/berhenti, merupakan kondisi berhenti dari perulangan, harus bernilai true selama perulangan dilakukan.

### 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.

- Struktur repeat until :

```
repeat  
    // aksi;  
until condition;
```

## CONTOH SOAL

- 1.) Buatlah program menggunakan bahasa Go yang menerima input kata dan mencetaknya sebanyak jumlah pengulangan yang diinginkan oleh pengguna. Program akan dihentikan ketika jumlah kata yang dicetak mencapai jumlah yang diinginkan oleh pengguna.

Masukan berupa suatu kata dan jumlah pengulangan yang diinginkan oleh pengguna. Keluaran berupa kata yang diinputkan pengguna dan dicetak sebanyak jumlah pengulangan yang diinginkan oleh pengguna.

Source Code:

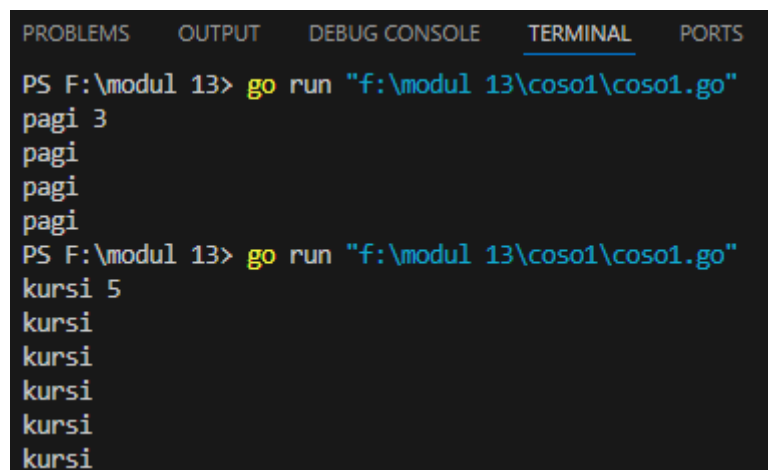
```
package main

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:



The screenshot shows a terminal window with the following content:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS F:\modul 13> go run "f:\modul 13\coso1\coso1.go"
pagi 3
pagi
pagi
pagi
PS F:\modul 13> go run "f:\modul 13\coso1\coso1.go"
kursi 5
kursi
kursi
kursi
kursi
kursi
```

#### Deskripsi Program:

Program ini merupakan program sederhana bahasa Go yang bertujuan untuk menerima input kata dan mencetaknya sebanyak jumlah pengulangan yang diinginkan oleh pengguna. Program akan dihentikan ketika jumlah kata yang dicetak mencapai jumlah yang diinginkan oleh pengguna. Program meminta kita untuk memasukkan word dan repetitions. Word adalah variabel untuk kata dan repetitions adalah variabel untuk jumlah pengulangan kata. Untuk counter := 0, dimulai dari 0 (diinisialisasi dengan 0 untuk menghitung iterasi atau perulangan). Kemudian kita masuk ke struktur repeat until. For done := false; !done; perulangan akan terus berjalan selama nilai done adalah false. Variable done adalah boolean (mengatur kapan perulangan berhenti). Kemudian masuk ke isi perulangan, program akan mencetak word. Variabel counter++, nilai counter bertambah 1 di setiap iterasi. Untuk done = (counter >= repetitions), perulangan akan berhenti saat counter sudah mencapai repetitions.

- 2.) Buatlah program dalam bahasa Go yang meminta pengguna untuk memasukkan bilangan bulat positif. Program akan terus meminta input hingga pengguna memasukkan bilangan bulat positif.

Masukan berupa bilangan bulat positif, apabila bukan maka program akan terus meminta masukan hingga bilangan yang diberikan adalah bilangan bulat positif.

Keluaran berupa satu baris keluaran yang menunjukkan n bilangan adalah bilangan bulat positif."

Source Code:

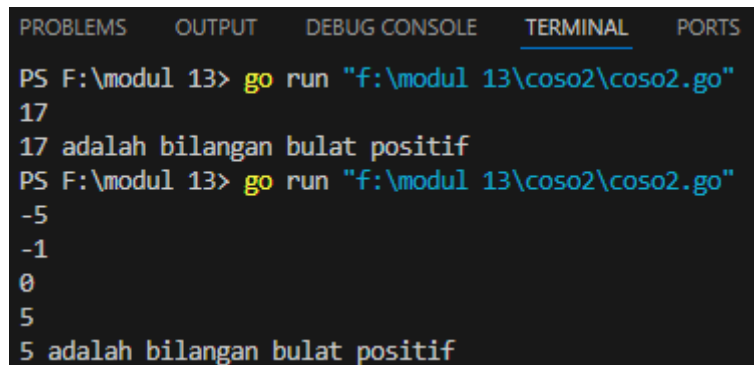
```
package main

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
PS F:\modul 13> go run "f:\modul 13\coso2\coso2.go"
17
17 adalah bilangan bulat positif
PS F:\modul 13> go run "f:\modul 13\coso2\coso2.go"
-5
-1
0
5
5 adalah bilangan bulat positif
```

#### Deskripsi Program:

Program ini merupakan program sederhana bahasa Go yang bertujuan untuk meminta pengguna memasukkan bilangan bulat positif. Program akan terus meminta input hingga pengguna memasukkan bilangan bulat positif. Variabel `number` digunakan untuk bilangan bulat (integer) dan `continueLoop` (bool) untuk menentukan apakah perulangan akan berlanjut. Kemudian kita masuk ke struktur `repeat until`. `For continueLoop = true; continueLoop;` perulangan akan terus berjalan selama nilai `continueLoop` adalah `true`. Kemudian masuk ke isi perulangan, program meminta kita memasukkan number (bilangan bulat). Untuk `continueLoop = number <= 0`, jika number kurang dari atau sama dengan 0 bernilai `true`, maka perulangan berlanjut. Sebaliknya jika bernilai `false` maka perulangan dihentikan. Keluaran berupa satu baris yang menunjukkan number adalah bilangan bulat positif, `fmt.Printf("%d adalah bilangan bulat positif\n", number)`.

- 3.) Buatlah program yang digunakan untuk melakukan pengecekan apakah suatu bilangan merupakan kelipatan dari bilangan lainnya.  
Masukan terdiri dari dua buah bilangan bulat positif X dan Y.  
Keluaran terdiri dari perulangan pengurangan kelipatan dengan hasil akhir Boolean yang menyatakan apakah bilangan X merupakan kelipatan dari Y.

Source Code:

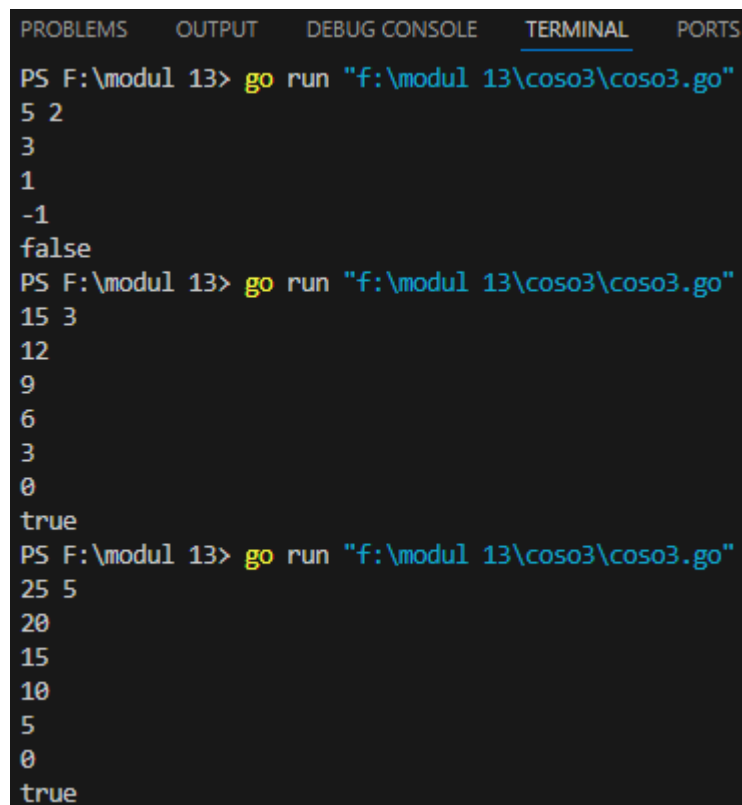
```
package main

import "fmt"

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

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

Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS F:\modul 13> go run "f:\modul 13\coso3\coso3.go"
5 2
3
1
-1
false
PS F:\modul 13> go run "f:\modul 13\coso3\coso3.go"
15 3
12
9
6
3
0
true
PS F:\modul 13> go run "f:\modul 13\coso3\coso3.go"
25 5
20
15
10
5
0
true
```

#### Deskripsi Program:

Program ini merupakan program sederhana bahasa Go yang bertujuan untuk melakukan pengecekan apakah suatu bilangan merupakan kelipatan dari bilangan lainnya. Program ini meminta kita untuk memasukkan bilangan bulat  $x$  dan  $y$ . Kemudian kita masuk ke struktur repeat until. For  $\text{done} = \text{false}; !\text{done};$  perulangan akan terus berjalan selama nilai  $\text{done}$  adalah  $\text{false}$ . Kemudian masuk ke isi perulangan, program akan menghitung operasi pengurangan  $x = x - y$  lalu program akan mencetak nilai  $x$  (setelah dikurangi). Untuk  $\text{done} = x \leq 0$ , perulangan akan berhenti jika nilai  $x$  kurang dari atau sama dengan 0. Keluaran berupa perulangan pengurangan kelipatan dengan hasil akhir boolean yang menyatakan apakah bilangan  $x$  merupakan kelipatan dari  $y$ , `fmt.Println(x == 0)`. Jika  $x == 0$  maka `true`, selain itu maka `false`.



## SOAL LATIHAN

- 1.) Buatlah program yang digunakan untuk menghitung banyaknya digit dari suatu bilangan.

Masukan berupa bilangan bulat positif.

Keluaran berupa bilangan bulat yang menyatakan banyaknya digit dari bilangan yang diberikan pada masukan.

Source Code:

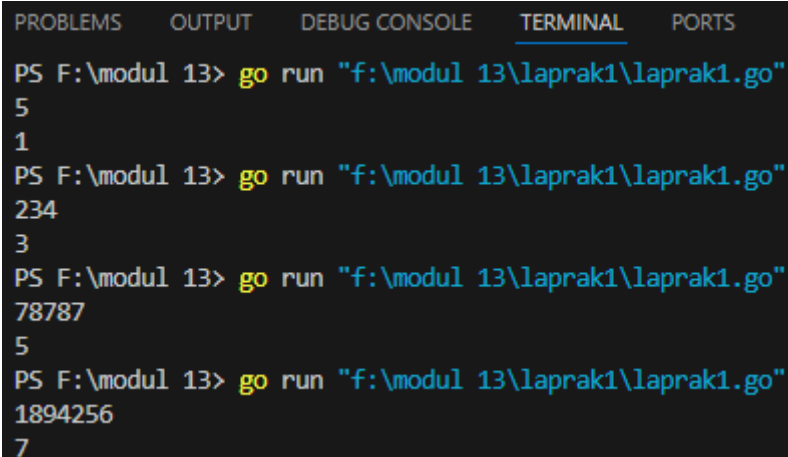
```
package main

import "fmt"

func main() {
    var angka, digit, i int
    fmt.Scan(&angka)
    digit = 1

    for proses := true; proses; {
        digit = digit * 10
        proses = digit < angka
        i++
    }
    fmt.Println(i)
}
```

Output:



The screenshot shows a terminal window with the following content:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS F:\modul 13> go run "f:\modul 13\laprak1\laprak1.go"
5
1
PS F:\modul 13> go run "f:\modul 13\laprak1\laprak1.go"
234
3
PS F:\modul 13> go run "f:\modul 13\laprak1\laprak1.go"
78787
5
PS F:\modul 13> go run "f:\modul 13\laprak1\laprak1.go"
1894256
7
```

#### Deskripsi Program:

Program ini merupakan program sederhana bahasa Go yang bertujuan untuk menghitung banyaknya digit dari suatu bilangan. Program ini meminta kita untuk memasukkan angka. Untuk digit = 1, dimulai dari 1. Kemudian kita masuk ke struktur repeat until. For proses := true; proses; perulangan akan terus berjalan selama nilai proses adalah true. Kemudian masuk ke isi perulangan. Program akan menghitung digit = digit \* 10. Untuk proses = digit < angka, jika digit kurang dari angka maka proses akan bernilai true dan perulangan akan berlanjut, sebaliknya jika kondisi ini bernilai false maka perulangan akan berhenti dan mencetak nilai i. Variabel i++, nilai i bertambah 1 di setiap iterasi. Keluaran berupa bilangan bulat yang menyatakan banyaknya digit dari bilangan yang diberikan pada masukan, fmt.Println(i).

- 2.) Buatlah program yang digunakan untuk mendapatkan bilangan bulat optimal dari bilangan yang telah diinputkan. Melakukan penjumlahan tiap perulangan mencapai pembulatan keatas dari bilangan yang diinputkan.  
Masukan berupa bilangan desimal.  
Keluaran terdiri dari bilangan hasil penjumlahan tiap perulangannya sampai pembulatan keatas dari bilangan yang diinputkan.

Source Code:

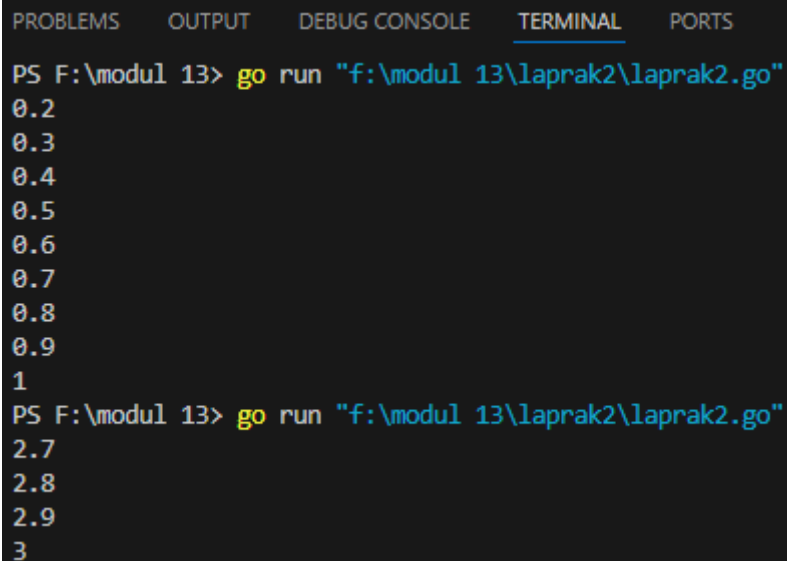
```
package main

import "fmt"

func main() {
    var des float64
    var n int
    fmt.Scan(&des)
    n = int(des) + 1

    for done := false; !done; {
        des += 0.1
        fmt.Printf("%.1f\n", des)
        done = des > float64(n) - 0.11
    }
    fmt.Print(n)
}
```

Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS F:\modul 13> go run "f:\modul 13\laprak2\laprak2.go"
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
PS F:\modul 13> go run "f:\modul 13\laprak2\laprak2.go"
2.7
2.8
2.9
3
```

#### Deskripsi Program:

Program ini merupakan program sederhana bahasa Go yang bertujuan untuk mendapatkan bilangan bulat optimal dari bilangan yang telah diinputkan dengan melakukan penjumlahan tiap perulangan mencapai pembulatan keatas dari bilangan yang diinputkan. Program ini meminta kita untuk memasukkan bilangan desimal. Untuk  $n = \text{int}(\text{des}) + 1$ , mengambil hasil angka bulat dari bilangan desimal. Kemudian kita masuk ke struktur repeat until. For `done := false; !done;` perulangan akan terus berjalan selama nilai `done` adalah `false`. Kemudian masuk ke isi perulangan, `des += 0.1`, menambahkan 0.1 ke bilangan desimal. Program akan mencetak desimal dengan format satu angka di belakang koma, `fmt.Printf("%.1f\n", des)`. Untuk `done = des > float64(n) - 0.11`, perulangan akan berhenti jika desimal lebih dari `float64(n)` atau desimal mendekati nilai integer `n` bernilai `true`. Keluaran terdiri dari bilangan hasil penjumlahan tiap perulangannya sampai pembulatan keatas dari bilangan yang diinputkan, `fmt.Print(n)`.

- 3.) Sebuah organisasi amal sedang mengumpulkan donasi untuk mendukung kegiatan sosial mereka. Setiap donatur dapat memberikan sumbangan dalam jumlah tertentu. Program ini akan terus meminta input dari pengguna untuk jumlah donasi hingga total donasi mencapai atau melebihi target yang telah ditentukan.
- Masukan pada baris pertama berupa bilangan bulat yang merupakan target donasi yang harus dicapai. Masukan pada baris berikut dan seterusnya merupakan bilangan bulat yang menyatakan donasi oleh setiap donatur, masukan terus diterima hingga target tercapai.
- Keluaran berupa bilangan hasil total penjumlahan tiap perulangannya serta jumlah donatur.

Source Code:

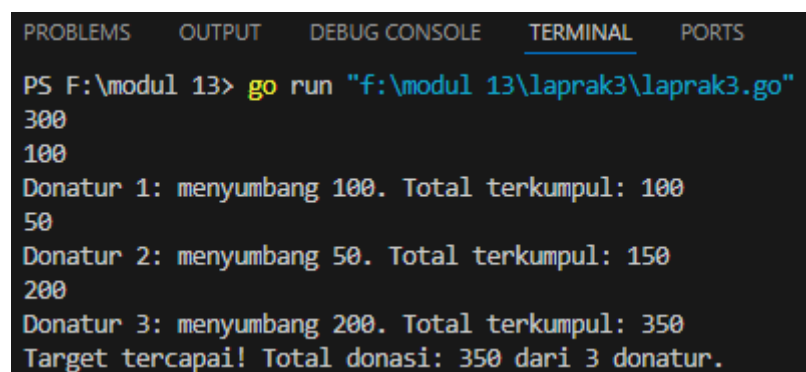
```
package main

import "fmt"

func main() {
    var target, donasi, total int
    fmt.Scan(&target)
    donatur := 0

    for done := false; !done; {
        fmt.Scan(&donasi)
        total = total + donasi
        donatur++
        fmt.Printf("Donatur %v: menyumbang %v. Total terkumpul: %v\n",
donatur, donasi, total)
        done = total >= target
    }
    fmt.Printf("Target tercapai! Total donasi: %v dari %v donatur.", total, donatur)
}
```

Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS F:\modul 13> go run "f:\modul 13\laprak3\laprak3.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.
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS F:\modul 13> go run "f:\modul 13\laprak3\laprak3.go"
500
150
Donatur 1: menyumbang 150. Total terkumpul: 150
100
Donatur 2: menyumbang 100. Total terkumpul: 250
50
Donatur 3: menyumbang 50. Total terkumpul: 300
300
Donatur 4: menyumbang 300. Total terkumpul: 600
Target tercapai! Total donasi: 600 dari 4 donatur.
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS F:\modul 13> go run "f:\modul 13\laprak3\laprak3.go"
200
300
Donatur 1: menyumbang 300. Total terkumpul: 300
Target tercapai! Total donasi: 300 dari 1 donatur.
```

#### Deskripsi Program:

Program ini merupakan program sederhana bahasa Go yang bertujuan untuk menghitung total donasi dan jumlah donator serta donasi yang diberikan oleh masing-masing donatur. Program meminta kita untuk memasukkan target. Untuk donator := 0, dimulai dari 0 (diinisialisasi dengan 0 untuk menghitung iterasi atau perulangan). Kemudian kita masuk ke struktur repeat until. For done := false; !done; perulangan akan terus berjalan selama nilai done adalah false. Kemudian masuk ke isi perulangan, program meminta kita memasukkan donasi (jumlah donasi yang diberikan donator). Lalu program menghitung total = total + donasi, menambahkan total dengan donasi. Variabel donator++, nilai donator bertambah 1 di setiap iterasi. Kemudian program akan mencetak informasi mengenai donator ke berapa dan jumlah donasi yang disumbangkan serta total donasi yang terkumpul, `fmt.Printf("Donatur %v: menyumbang %v. Total terkumpul: %v\n", donatur, donasi, total)`. Untuk `done = total >= target`, perulangan akan berhenti jika total telah mencapai atau lebih dari sama dengan target bernilai true. Keluaran berupa total donasi dan jumlah donator, `fmt.Printf("Target tercapai! Total donasi: %v dari %v donatur.", total, donatur)`.

.

## DAFTAR PUSTAKA

- Izzuddin, D. (2024). *Perulangan dengan REPEAT UNTIL loop pada Pascal*.  
<https://invasikode.com/belajar/pascal/struktur-kontrol/perulangan-dengan-repeat-until-loop-pada-pascal>
- Prasti Eko Yunanto, S. T. , M. Kom. (2004). *MODUL PRAKTIKUM 13 – REPEAT-UNTIL ALGORITMA DAN PEMROGRAMAN 1 S1 INFORMATIKA* .