

**LAPORAN PRAKTIKUM ALGORITMA  
DAN PEMROGRAMAN 1**

**MODUL 13  
REPEAT-UNTIL**



**Disusun oleh:**

**DYAH IMANSARI**

**109082500130**

**S1IF-13-02**

**Asisten Praktikum**

Adithana dharma putra

Alfin Ilham Berlianto

**PROGRAM STUDI S1 INFORMATIKA  
FAKULTAS INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO**

**2025**

## LATIHAN KELAS – GUIDED

### 1. Guided 1 Source Code

```
package main

import "fmt"

func main() {

    var kata string

    var jumlah int

    fmt.Scan(&kata, &jumlah)

    counter := 0

    for done := false; !done; {

        fmt.Println(kata)

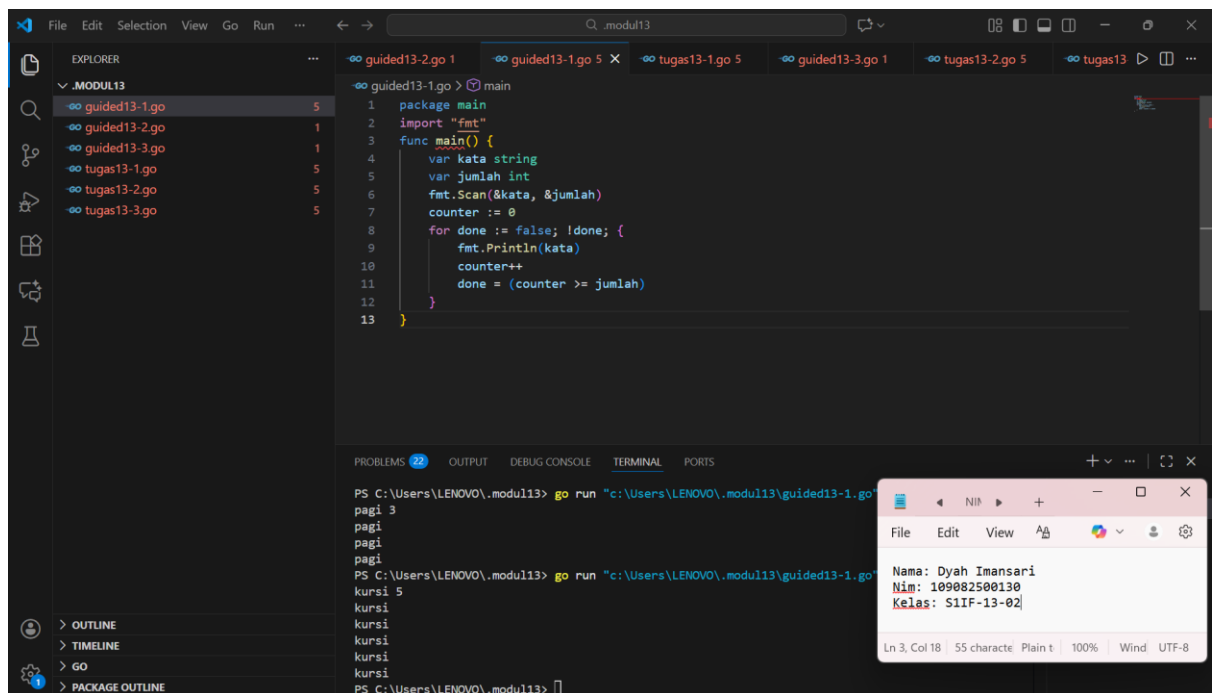
        counter++

        done = (counter >= jumlah)

    }

}
```

**Screenshoot program**



## Deskripsi program

Program ini digunakan 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. Setelah pengguna memasukkan kata dan jumlah pengulangan kata yang diinginkan, program akan langsung mengeksekusi keduanya. Sebelumnya counter sudah diinisialisasikan dengan 0 sebagai penghitung jumlah kata yang sudah dicetak. Struktur `for done := false; !done; {...}` berarti loop akan terus berjalan selama done masih false (artinya proses belum selesai). Di dalam loop, program mencetak kata yang diinput, kemudian `counter++` menaikkan jumlah hitungan kata yang sudah dicetak. Kondisi `done = (counter >= jumlah)` menentukan kapan loop berhenti. Jika nilai counter sudah mencapai atau melebihi nilai jumlah, maka done menjadi true dan perulangan selesai.

## 2. Guided 2

### Source Code

```
package main

import "fmt"

func main() {

    var angka int

    var done bool
```

```

        for done = false; !done; {

            fmt.Scan(&angka)

            done = (angka > 0)

        }

        fmt.Printf("%d adalah bilangan bulat positif\n",
angka)

    }

```

### Screenshoot program

```

1 package main
2 import "fmt"
3 func main() {
4     var angka int
5     var done bool
6     for done = false; !done; {
7         fmt.Scan(&angka)
8         done = (angka > 0)
9     }
10    fmt.Printf("%d adalah bilangan bulat positif\n", angka)
11 }

```

PS C:\Users\LENOVO\.modul13> go run "c:\Users\LENOVO\.modul13\guided13-2.go"  
 -5  
 -2  
 -1  
 0  
 5  
 5 adalah bilangan bulat positif  
 PS C:\Users\LENOVO\.modul13> go run "c:\Users\LENOVO\.modul13\guided13-2.go"  
 17  
 17 adalah bilangan bulat positif  
 PS C:\Users\LENOVO\.modul13>

Nama: Dyah Imansari  
 Nim: 109082500130  
 Kelas: S1IF-13-02

### Deskripsi program

Program ini meminta pengguna untuk memasukkan bilangan bulat positif. Program akan terus meminta input hingga pengguna memasukkan bilangan bulat positif. Struktur `for done = false; !done; {...}` berarti loop akan terus berjalan selama `done` masih `false` (artinya proses belum selesai). Di dalam loop, program membaca angka yang dimasukkan pengguna. Jika angka tersebut lebih besar dari nol, maka kondisi `done = (angka > 0)` akan bernilai `true`, sehingga loop berhenti. Jika tidak, loop akan berlanjut dan meminta input lagi. Setelah pengguna memasukkan bilangan bulat positif, program akan mencetak angka tersebut dan kemudian selesai.

### 3. Guided 3

#### Source Code

```
package main

import "fmt"

func main() {

    var X, Y int

    var done bool

    fmt.Print("Masukkan bilangan X dan Y: ")

    fmt.Scan(&X, &Y)

    fmt.Println("Proses pengurangan: ")

    for done = false; !done; {

        X = X - Y

        fmt.Println(X)

        done = (X <= 0)

    }

    fmt.Println("Apakah X kelipatan dari Y?")

    fmt.Println(X == 0)

}
```

**Screenshoot program**

```
package main
import "fmt"
func main() {
    var X, Y int
    var done bool
    fmt.Print("Masukkan bilangan X dan Y: ")
    fmt.Scan(&X, &Y)
    fmt.Println("Proses pengurangan: ")
    for done = false; !done; {
        X = X - Y
        fmt.Println(X)
        done = (X <= 0)
    }
    fmt.Println("Apakah X kelipatan dari Y?")
    fmt.Println(X == 0)
}
```

Masukkan bilangan X dan Y: 5 2  
Proses pengurangan:  
3  
1  
-1  
Apakah X kelipatan dari Y?  
false  
PS C:\Users\LENOVO\.modul13> go run "c:\Users\LENOVO\.modul13\guided13-3.go"  
Masukkan bilangan X dan Y: 15 3  
Proses pengurangan:  
12  
9  
6  
3  
0  
Apakah X kelipatan dari Y?  
true

```
package main
import "fmt"
func main() {
    var X, Y int
    var done bool
    fmt.Print("Masukkan bilangan X dan Y: ")
    fmt.Scan(&X, &Y)
    fmt.Println("Proses pengurangan: ")
    for done = false; !done; {
        X = X - Y
        fmt.Println(X)
        done = (X <= 0)
    }
    fmt.Println("Apakah X kelipatan dari Y?")
    fmt.Println(X == 0)
}
```

Masukkan bilangan X dan Y: 25 5  
Proses pengurangan:  
20  
15  
10  
5  
0  
Apakah X kelipatan dari Y?  
true  
PS C:\Users\LENOVO\.modul13>

## Deskripsi program

Program ini digunakan untuk melakukan pengecekan apakah suatu bilangan (X) merupakan kelipatan dari bilangan lainnya (Y). Program ini mengurangi X dengan Y berulang kali untuk melihat apakah X bisa habis dibagi oleh Y. Jika akhirnya X menjadi 0, berarti X adalah kelipatan dari Y. Jika tidak, maka bukan. Setelah pengguna memasukkan bilangan X dan Y, program akan langsung mengeksekusi. Struktur `for done = false; !done; {...}` berarti loop akan terus berjalan selama done masih false (artinya proses belum selesai). Di dalam loop, X akan terus mengurangi Y ( $X = X - Y$ ) kemudian mencetak X. Loop akan berhenti ketika angka tersebut (X) kurang dari atau sama dengan nol (kondisi `done = (X <= 0)` bernilai true). Setelah itu, program akan mencetak apakah (true/false) X kelipatan dari Y dan kemudian program selesai.

## TUGAS

### 1. Tugas 1

#### Source code

```
package main

import "fmt"

func main() {
```

```

var x int

fmt.Print("Masukkan bilangan bulat positif: ")

fmt.Scan(&x)

counter := 0

for x != 0 {

    counter++

    x = x/10

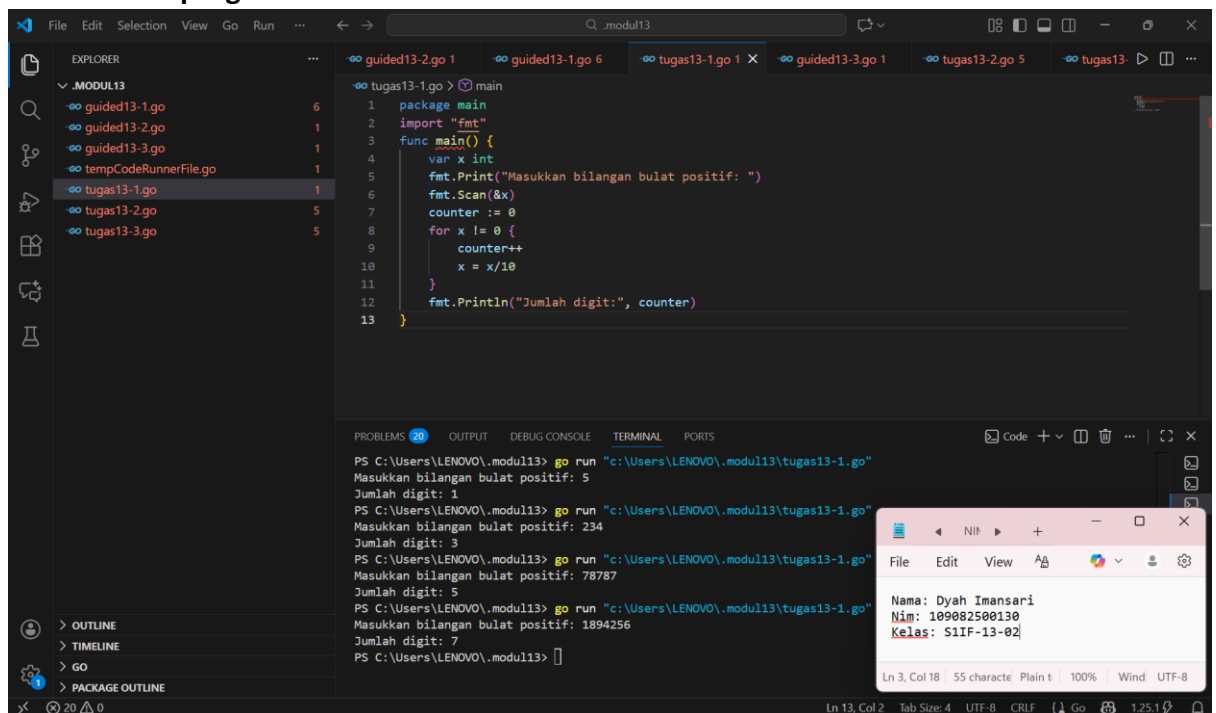
}

fmt.Println("Jumlah digit:", counter)

}

```

### Screenshoot program



### Deskripsi program

Program ini digunakan untuk menghitung banyaknya digit dari suatu bilangan bulat positif. Program ini bekerja dengan membaca sebuah bilangan bulat positif dari pengguna. Counter diinisialisasikan dengan 0 sebagai penghitung jumlah digit. Setelah menerima input dari pengguna, program akan langsung mengeksekusinya. Struktur `for x != 0 { ... }` berarti loop akan terus berjalan selama `x` tidak sama dengan 0. Selama `x` tidak sama dengan 0, program akan menghitung jumlah digitnya dengan cara

membagi bilangan tersebut berulang-ulang dengan 10 ( $x = x/10$ ) hingga nilainya menjadi 0. Setiap kali pembagian dilakukan, penghitung (counter) ditambah satu (counter++) sehingga pada akhir proses nilai counter menunjukkan berapa banyak digit yang dimiliki bilangan tersebut.

## 2. Tugas 2

### Source code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var x float64

    fmt.Print("Masukkan bilangan desimal: ")

    fmt.Scan(&x)

    batas := math.Ceil(x)

    hasil := x

    fmt.Println("Hasil penjumlahan:")

    for {
        hasil += 0.1

        fmt.Printf("%.1f\n", hasil)

        if math.Round(hasil*10)/10 >= batas {
            break
        }
    }

    fmt.Printf("Hasil Pembulatan ke atas: %.0f\n",
        hasil)
```



```
}
```

## Screenshoot program

The screenshot shows a Go program in VS Code. The Explorer pane on the left shows a project named .MODUL13 with files guided13-1.go, guided13-2.go, guided13-3.go, tempCodeRunnerFile.go, tugas13-1.go, tugas13-2.go, and tugas13-3.go. The main editor displays tugas13-2.go with the following code:

```
1 package main
2 import (
3     "fmt"
4     "math"
5 )
6 func main() {
7     var x float64
8     fmt.Print("Masukkan bilangan desimal: ")
9     fmt.Scan(&x)
10    batas := math.Ceil(x)
11    hasil := x
12    fmt.Println("Hasil penjumlahan:")
13    for {
14        hasil += 0.1
15        fmt.Printf("%.1f\n", hasil)
16        if math.Round(hasil*10)/10 >= batas {
17            break
18        }
19    }
20    fmt.Printf("Hasil Pembulatan ke atas: %.0f\n", hasil)
21 }
```

The TERMINAL pane shows the output of the program:

```
Masukkan bilangan desimal: 0.2
Hasil penjumlahan:
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Hasil Pembulatan ke atas: 1
```

A second screenshot shows the program being run from a command prompt. The command is `PS C:\Users\LENOVO\.modul13> go run "c:\Users\LENOVO\.modul13\tugas13-2.go"`. The output is:

```
Masukkan bilangan desimal: 2.7
Hasil penjumlahan:
2.8
2.9
3.0
Hasil Pembulatan ke atas: 3
PS C:\Users\LENOVO\.modul13>
```

A small window in the bottom right corner displays student information:

```
Nama: Dyah Imansari
Nim: 109082500130
Kelas: SIIF-13-02
```

## Deskripsi program

Program ini digunakan untuk memperoleh bilangan bulat terdekat ke atas dari nilai yang diinputkan dengan melakukan penjumlahan secara berulang hingga mencapai hasil pembulatan ke atas tersebut. Selain mengimpor paket `fmt`, program juga menggunakan paket `math` untuk memanfaatkan fungsi `math.Ceil(x)` sebagai penentu batas pembulatan ke atas. Setelah pengguna memasukkan sebuah bilangan desimal, nilai `batas := math.Ceil(x)` digunakan sebagai acuan, sementara `hasil := x` berfungsi sebagai inisialisasi nilai awal. Perulangan dilakukan menggunakan struktur `for { ... }` tanpa kondisi, yang berarti akan terus berjalan hingga perintah `break` dijalankan. Di dalam loop, operasi `hasil += 0.1` menambah nilai hasil sebesar 0.1 pada setiap iterasi untuk mendekatkannya secara bertahap ke nilai pembulatan ke atas, dan setiap kenaikan ditampilkan dengan satu angka di belakang koma menggunakan `fmt.Printf("%.1f\n", hasil)`. Kondisi penghentian menggunakan `math.Round(hasil*10)/10 >= batas`, yang berfungsi mengatasi kesalahan presisi bilangan pecahan dan memastikan bahwa nilai yang telah dibulatkan ke satu desimal benar-benar mencapai atau melampaui batas pembulatan. Ketika kondisi tersebut terpenuhi, loop dihentikan. Pada akhir, program akan mencetak hasil bilangan bulat tanpa koma.

### 3. Tugas 3

#### Source code

```
package main

import "fmt"

func main() {

    var target, donasi int

    fmt.Print("Masukkan target donasi: ")
    fmt.Scanln(&target)

    fmt.Println("Masukkan donasi dari tiap donatur: ")

    counter := 0
    total := 0
    for {

        counter++

        fmt.Scan(&donasi)

        fmt.Printf("Donatur %d: Menyumbang %d. ",
counter, donasi)

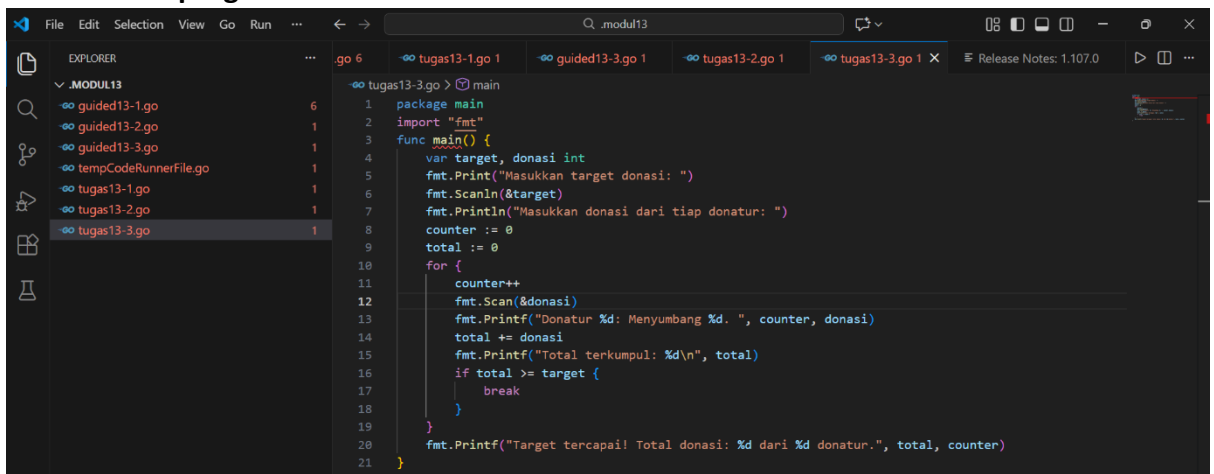
        total += donasi

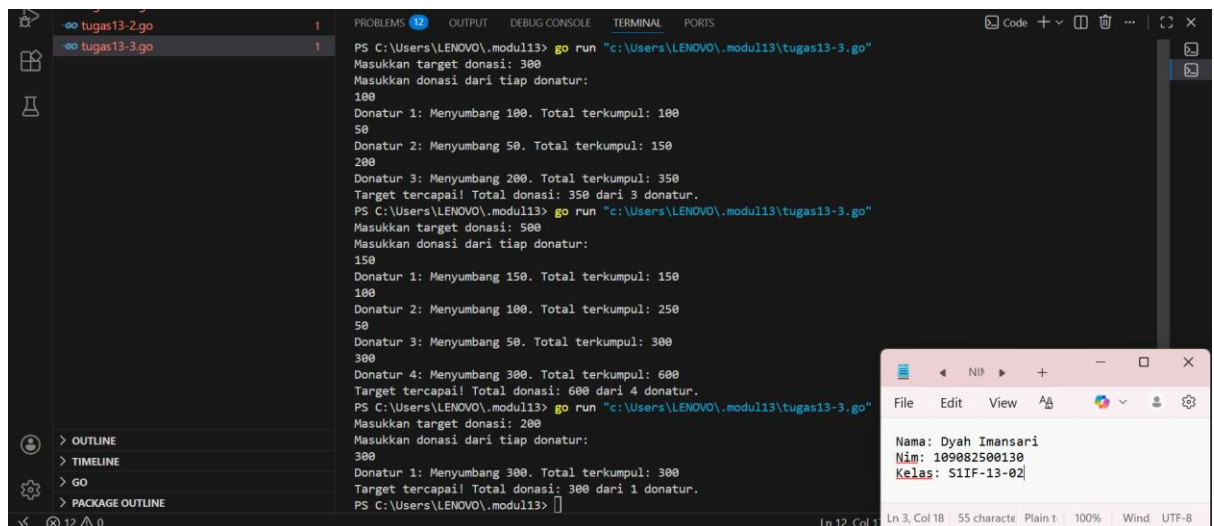
        fmt.Printf("Total terkumpul: %d\n", total)
        if total >= target {
            break
        }

    }

    fmt.Printf("Target tercapai! Total donasi: %d dari
%d donatur.", total, counter)
}
```

#### Screenshoot program





```
PS C:\Users\LENOVO\.modul13> go run "c:\Users\LENOVO\.modul13\tugas13-3.go"
Masukkan target donasi: 300
Masukkan donasi dari tiap donatur:
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 C:\Users\LENOVO\.modul13> go run "c:\Users\LENOVO\.modul13\tugas13-3.go"
Masukkan target donasi: 500
Masukkan donasi dari tiap donatur:
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.
PS C:\Users\LENOVO\.modul13> go run "c:\Users\LENOVO\.modul13\tugas13-3.go"
Masukkan target donasi: 200
Masukkan donasi dari tiap donatur:
300
Donatur 1: Menyumbang 300. Total terkumpul: 300
Target tercapai! Total donasi: 300 dari 1 donatur.
PS C:\Users\LENOVO\.modul13>
```

## Deskripsi program

Program ini bertujuan untuk membantu sebuah organisasi amal dalam mengumpulkan donasi dengan cara menerima input jumlah sumbangan dari setiap donatur. Program akan terus meminta donasi secara berulang dan menambahkan setiap sumbangan ke total donasi yang terkumpul. Proses ini berlangsung hingga total donasi mencapai atau melebihi target yang telah ditentukan sebelumnya.

Program ini dimulai dengan meminta pengguna memasukkan target donasi yang ingin dicapai, lalu masuk ke dalam sebuah perulangan yang terus berjalan hingga total donasi mencapai atau melebihi target tersebut. Pada setiap iterasi, program menambah penghitung donatur (melalui `counter++` yang sebelumnya bernilai 0), lalu meminta input jumlah donasi dari donatur saat itu. Nilai donasi tersebut kemudian ditambahkan ke total donasi menggunakan operasi `total += donasi`. Setelah setiap donasi diterima, program menampilkan informasi berupa nomor donatur, jumlah sumbangan, serta total donasi yang telah terkumpul menggunakan fungsi `fmt.Printf`. Ketika total donasi telah memenuhi atau melampaui target (`total >= target`), perulangan dihentikan menggunakan `break`, dan program menampilkan pesan akhir yang berisi total donasi yang terkumpul dan jumlah donatur yang terlibat.