

**LAPORAN PRAKTIKUM**  
**Algoritma Pemrograman**

**MODUL 13**  
**REPEAT-UNTIL**



**Disusun oleh:**  
**EWALDO ARDIANSYAH WIDYADHANA**  
**109082500008**  
**S1IF-13-04**

**PROGRAM STUDI S1 INFORMATIKA**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2025**

## LATIHAN KELAS – GUIDED

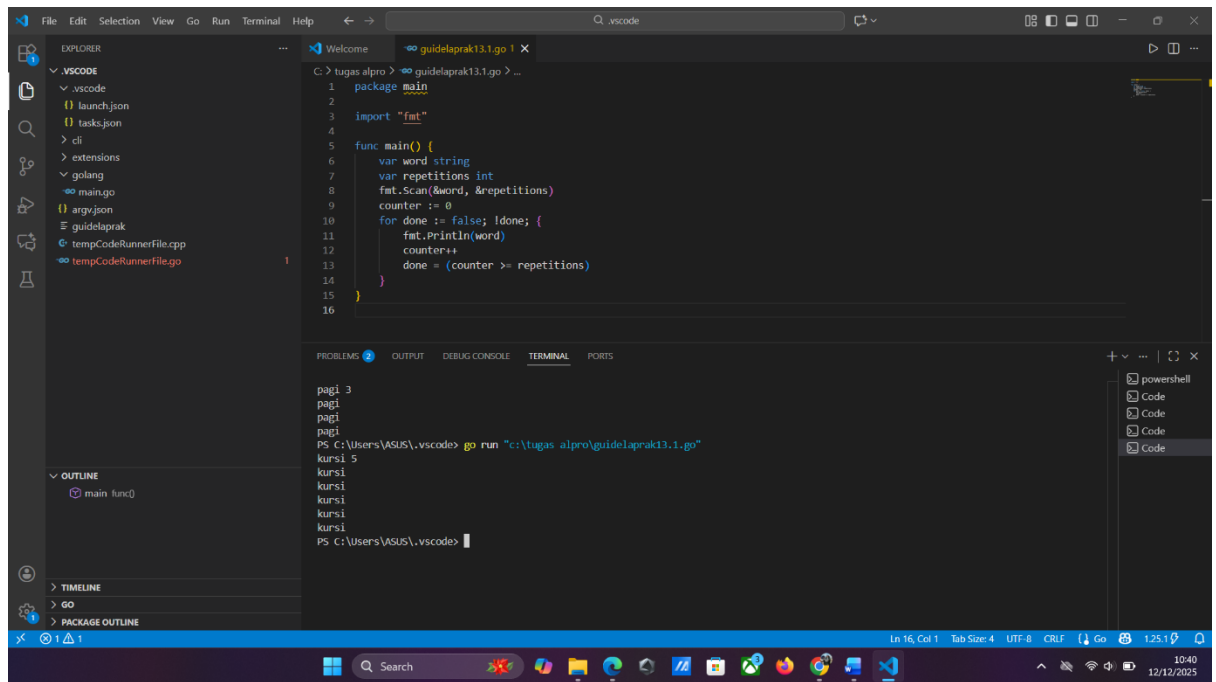
### 1. Guided 1 Source Code

```
package main

import "fmt"

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

### Screenshoot program



### Deskripsi program

Program ini dibuat untuk menampilkan sebuah kata sebanyak jumlah pengulangan yang dimasukkan oleh pengguna. Pada awal program, pengguna diminta memberikan dua input sekaligus, yaitu sebuah kata dan sebuah bilangan bulat yang menyatakan berapa kali kata tersebut harus dicetak. Setelah input diterima, program menjalankan sebuah perulangan menggunakan bentuk *loop* dengan kondisi yang diperbarui di dalam blok perulangan.

Variabel counter digunakan untuk menghitung berapa kali kata telah dicetak. Pada setiap iterasi, program menampilkan kata tersebut menggunakan `fmt.Println()`, lalu menaikkan nilai counter sebanyak satu. Ketika nilai counter telah mencapai jumlah pengulangan yang diminta pengguna, perulangan dihentikan dan program selesai dijalankan.

Secara keseluruhan, program ini berfungsi untuk mencetak sebuah kata berulang-ulang sesuai jumlah yang diinginkan oleh pengguna.

## 2. Guided 2

### Source Code

```
package main

import "fmt"

func main() {
```

```

var number int

var continueLoop bool

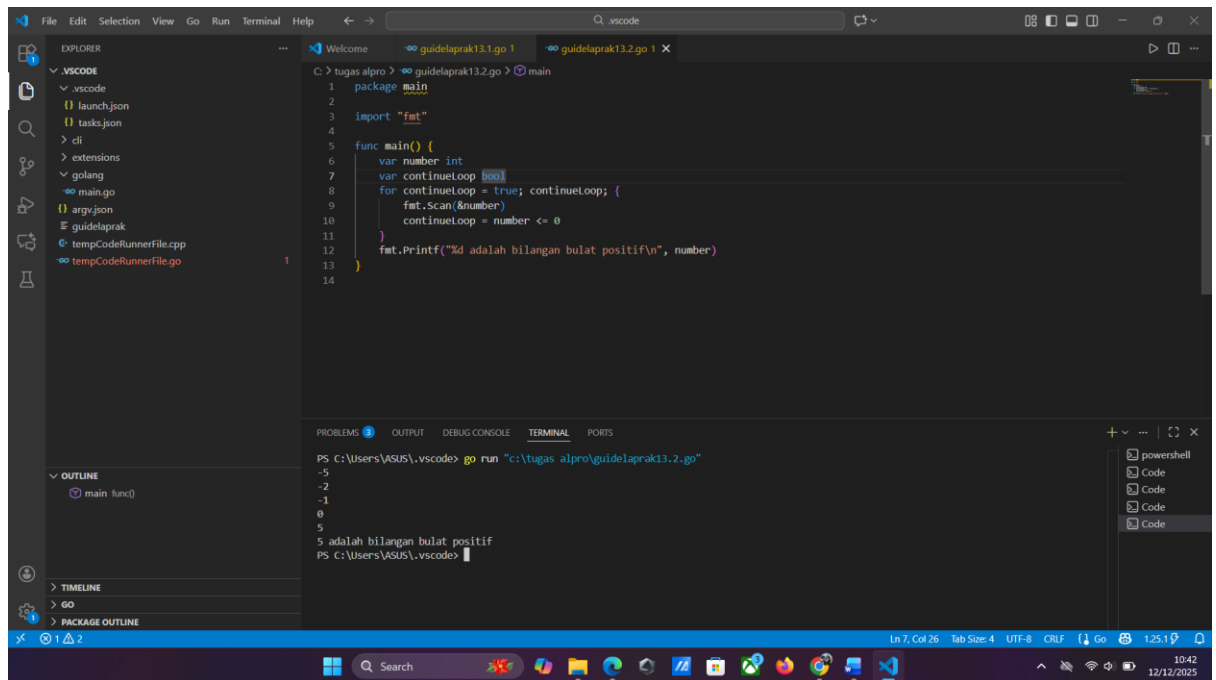
for continueLoop = true; continueLoop; {
    fmt.Scan(&number)

    continueLoop = number <= 0
}

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

```

### Screenshoot program



### Deskripsi program

Program ini dibuat untuk memastikan bahwa pengguna memasukkan sebuah bilangan bulat positif. Ketika program dijalankan, pengguna dapat memasukkan angka berkali-kali. Program menggunakan sebuah *loop* yang terus berjalan selama angka yang dimasukkan tidak memenuhi syarat, yaitu angka tersebut harus lebih besar dari nol.

Di dalam perulangan, program membaca input pengguna menggunakan `fmt.Scan()`. Setelah angka diterima, program memeriksa apakah angka tersebut kurang atau sama dengan nol. Jika iya, maka nilai variabel `continueLoop` tetap bernilai `true` sehingga perulangan akan terus berulang. Dengan demikian, pengguna dipaksa memasukkan angka baru sampai ia memberikan angka yang valid (positif).

Setelah pengguna memasukkan angka yang lebih besar dari nol, kondisi perulangan menjadi false dan loop berhenti. Program kemudian mencetak pesan yang menegaskan bahwa angka tersebut adalah bilangan bulat positif.

Secara keseluruhan, program ini berfungsi sebagai validasi input untuk memastikan bahwa pengguna hanya memasukkan bilangan bulat positif sebelum program dilanjutkan

### 3. Guided 3

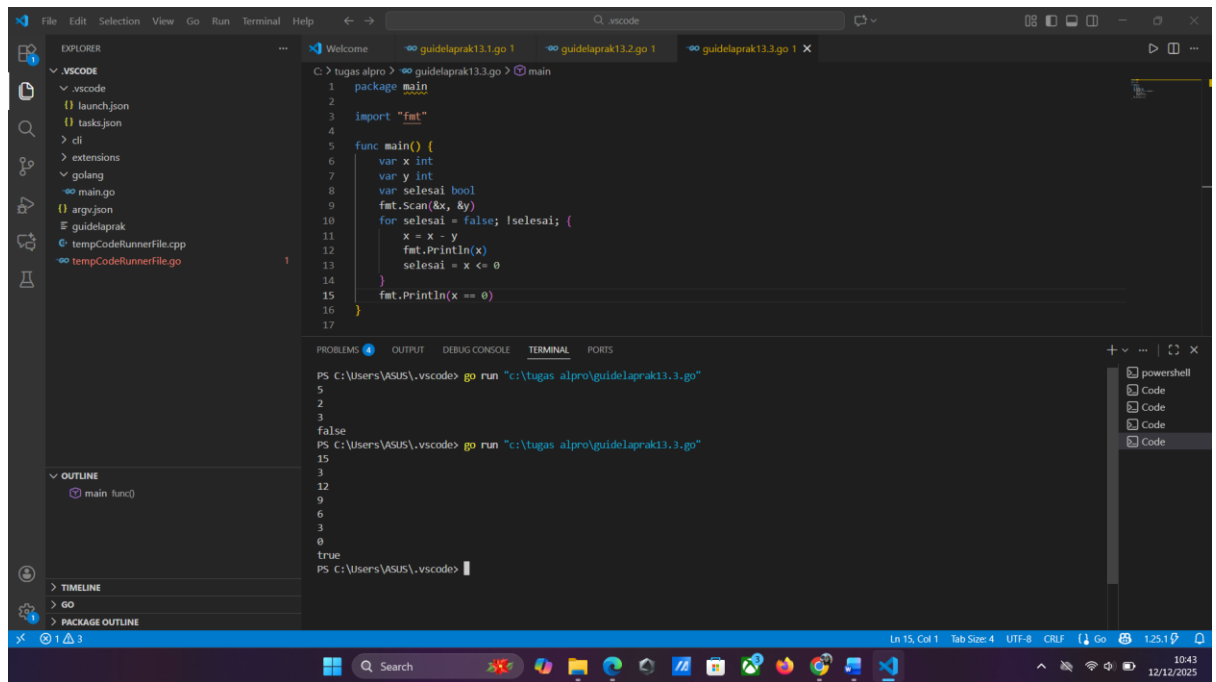
#### Source Code

```
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)
}
```

#### Screenshoot program



### Deskripsi program

Program ini dibuat untuk melakukan proses pengurangan berulang antara dua bilangan yang dimasukkan oleh pengguna. Program menerima dua input awal, yaitu nilai  $x$  dan  $y$ . Setelah kedua nilai tersebut dibaca, program memasuki sebuah perulangan yang bertujuan untuk mengurangi nilai  $x$  secara terus-menerus dengan  $y$ .

Pada setiap iterasi, program melakukan langkah berikut:

1. Mengurangi nilai  $x$  dengan  $y$  ( $x = x - y$ ).
2. Menampilkan nilai  $x$  setelah dikurangi.
3. Memeriksa apakah nilai  $x$  sudah kurang dari atau sama dengan nol.

Jika nilai  $x$  masih lebih besar dari nol, perulangan akan terus dilanjutkan. Namun jika nilai  $x$  sudah mencapai nilai nol atau negatif, perulangan dihentikan. Setelah loop selesai, program mencetak hasil pengecekan berupa nilai boolean true atau false untuk menyatakan apakah nilai akhir  $x$  sama dengan nol.

Secara keseluruhan, program ini digunakan untuk mensimulasikan pengurangan bertahap dan memeriksa apakah pengurangan tersebut dapat menghasilkan nilai tepat nol atau tidak.

## TUGAS

### 1. Tugas 1

#### Source code

```
package main

import "fmt"
```

```

func main() {

    var angka int

    fmt.Print("Masukkan angka: ")

    fmt.Scan(&angka)

    jumlahDigit := 0

    temp := angka

    for temp > 0 {

        temp = temp / 10

        jumlahDigit++

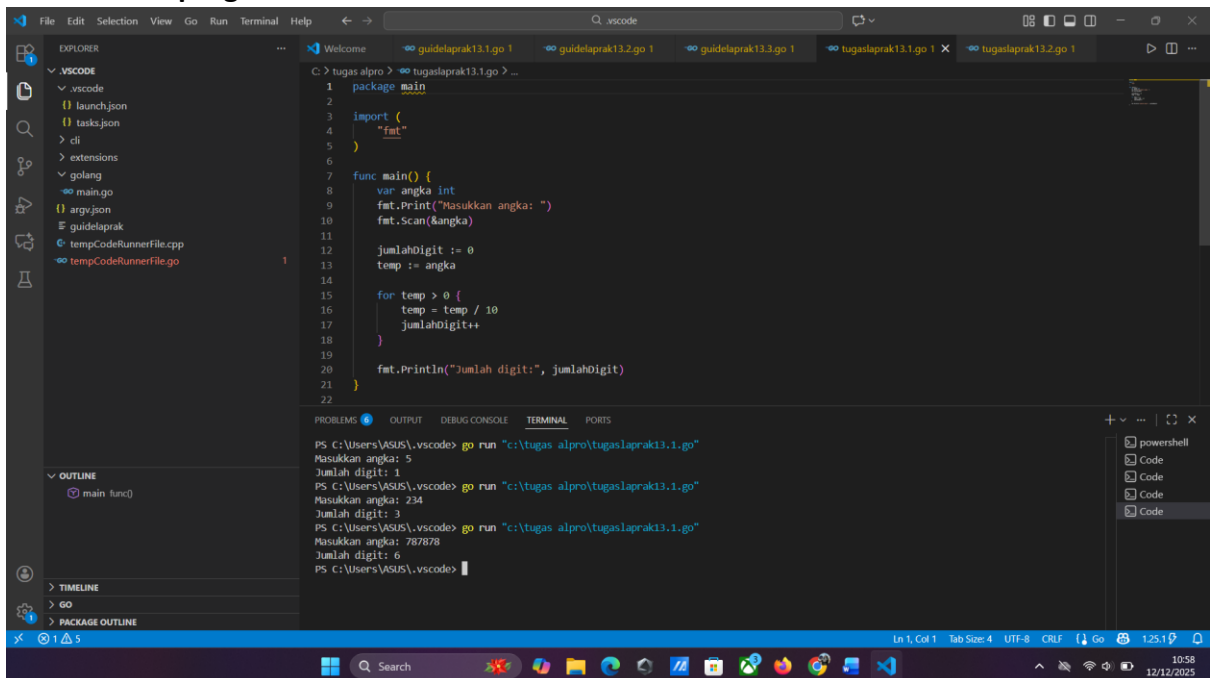
    }

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

}

```

## Screenshoot program



## Deskripsi program

Program ini berfungsi untuk menghitung berapa banyak digit yang terdapat pada sebuah bilangan bulat positif yang dimasukkan oleh pengguna. Program akan membaca satu angka sebagai input, kemudian menggunakan proses pembagian berulang untuk menghitung jumlah digitnya. Setiap pembagian menghilangkan satu digit dari belakang bilangan sampai bilangan menjadi nol. Hasil akhirnya adalah jumlah digit yang telah dihitung, dan nilai tersebut ditampilkan sebagai keluaran program.

## 2. Tugas 2

### Source code

```
package main

import "fmt"

func main() {
    var x float64
    fmt.Print("Masukkan bilangan desimal: ")
    fmt.Scan(&x)

    // Pembulatan ke atas (ceil) secara manual
    batas := float64(int(x))
    if x > batas {
        batas = batas + 1
    }

    hasil := x

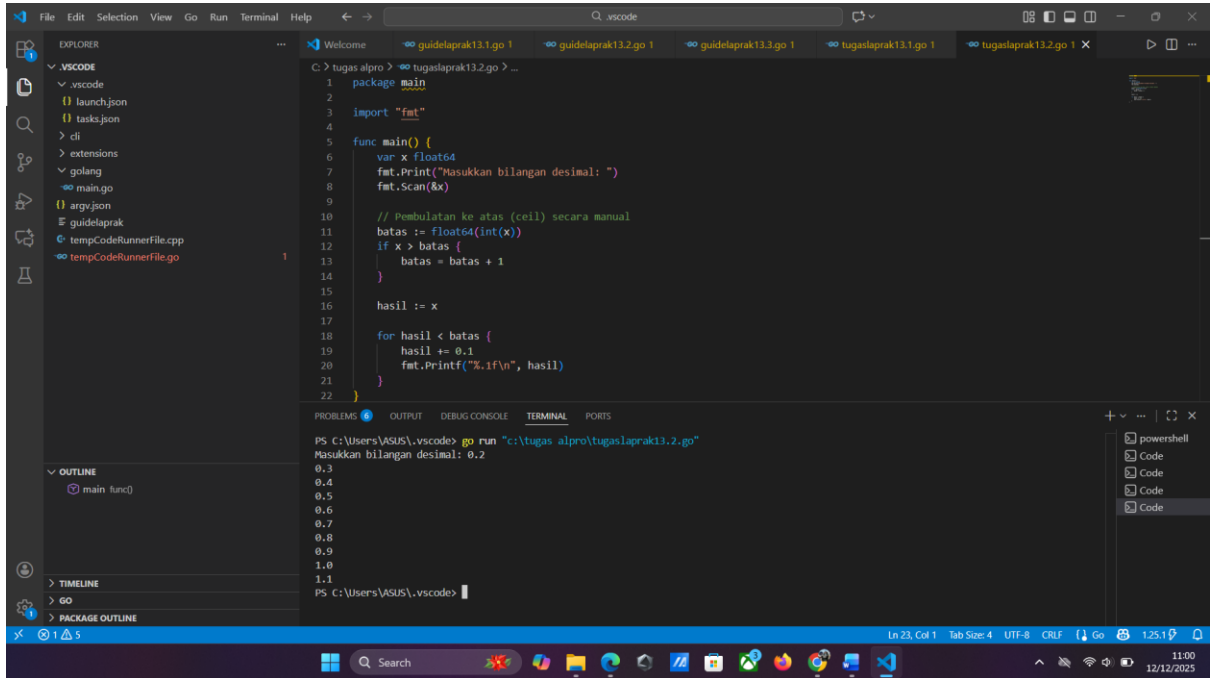
    for hasil < batas {
        hasil += 0.1
        fmt.Printf("%.1f\n", hasil)
    }
}
```



```
}

```

## Screenshoot program



## Deskripsi program

Program ini digunakan untuk mendapatkan nilai bilangan yang meningkat secara bertahap dari sebuah angka desimal hingga mencapai angka pembulatan ke atas (ceil) dari bilangan tersebut. Pengguna memasukkan sebuah bilangan desimal sebagai input. Program kemudian menambahkannya sebesar 0.1 pada setiap langkah perulangan dan menampilkan hasilnya. Proses ini terus dilakukan sampai nilai hasil penjumlahan mencapai atau melebihi angka bulat terdekat di atasnya. Program menampilkan seluruh nilai hasil penjumlahan selama proses berlangsung.

### 3. Tugas 3

#### Source code

```

package main

import "fmt"

func main() {
    var target int
    fmt.Print("Masukkan target donasi: ")
    fmt.Scan(&target)
}

```

```

total := 0
donatur := 1

for total < target {
    var donasi int
    fmt.Printf("Donatur %d menyumbang: ", donatur)
    fmt.Scan(&donasi)

    total += donasi
    fmt.Printf("Donatur %d: Menyumbang %d. Total terkumpul: %d\n",
        donatur, donasi, total)

    // jika sudah mencapai target → berhenti
    if total >= target {
        fmt.Printf("Target tercapai! Total donasi: %d dari %d donatur.\n",
            total, donatur)
        break
    }

    donatur++
}
}

```

## Screenshoot program

The screenshot shows a Go program in VS Code. The source code in the editor is as follows:

```

1 package main
2
3 import "fmt"
4
5 func main() {
6     var target int
7     fmt.Print("Masukkan target donasi: ")
8     fmt.Scan(&target)
9
10    total := 0
11    donatur := 1
12
13    for total < target {
14        var donasi int
15        fmt.Printf("Donatur %d menyumbang: ", donatur)
16        fmt.Scan(&donasi)
17
18        total += donasi
19        fmt.Printf("Donatur %d: Menyumbang %d. Total terkumpul: %d\n",
20            donatur, donasi, total)
21
22        // jika sudah mencapai target + berhenti
23    }
24
25    donatur++
26 }

```

The terminal output shows the program's execution:

```

PS C:\Users\ASUS\.vscode> go run "c:\tugas alpro\tugaslaprak13.3.go"
Masukkan target donasi: 300
Donatur 1 menyumbang: 100
Donatur 1: Menyumbang 100. Total terkumpul: 100
Donatur 2 menyumbang: 50
Donatur 2: Menyumbang 50. Total terkumpul: 150
Donatur 3 menyumbang: 200
Donatur 3: Menyumbang 200. Total terkumpul: 350
Target tercapai! Total donasi: 350 dari 3 donatur.
PS C:\Users\ASUS\.vscode>

```

## Deskripsi program

Program ini mensimulasikan proses pengumpulan donasi oleh sebuah organisasi amal. Pada awal program, pengguna memasukkan target donasi yang ingin dicapai.

Setelah itu, program akan menerima input jumlah donasi dari setiap donatur secara berurutan. Setiap kali donasi diterima, program menampilkan nomor donatur, jumlah sumbangan yang diberikan, serta total donasi yang sudah terkumpul. Proses ini berlanjut hingga total donasi mencapai atau melebihi target. Setelah target tercapai, program menampilkan total donasi yang terkumpul dan jumlah donatur yang berpartisipasi. Program ini membantu memantau donasi secara dinamis hingga target terpenuhi.