

**LAPORAN PRAKTIKUM ALGORITMA  
DAN PEMROGRAMAN 1**

**MODUL 13  
REPEAT-UNTIL**



**Disusun oleh:**

**M MAHDAN ARGYA SYARIF**

**109082500059**

**S1IF-13-07**

**Asisten Praktikum**

Adithana dharma putra

Apri pandu wicaksono

**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)

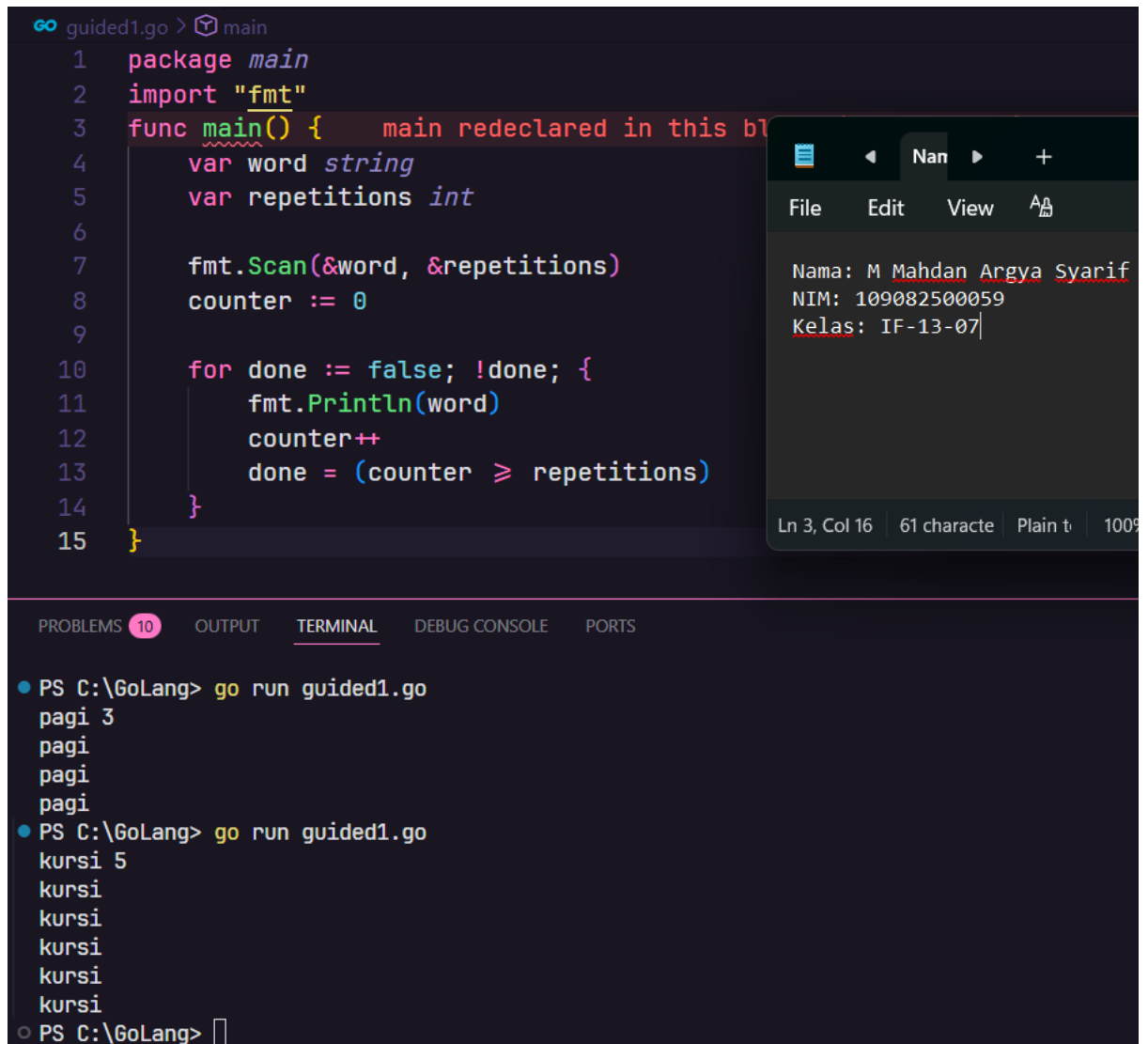
        counter++

        done = (counter >= repetitions)

    }

}
```

## Screenshoot program



```
go guided1.go > main
1 package main
2 import "fmt"
3 func main() { main redeclared in this block
4     var word string
5     var repetitions int
6
7     fmt.Scan(&word, &repetitions)
8     counter := 0
9
10    for done := false; !done; {
11        fmt.Println(word)
12        counter++
13        done = (counter ≥ repetitions)
14    }
15 }
```

File Edit View A

Nama: M Mahdan Argya Syarif  
NIM: 109082500059  
Kelas: IF-13-07

Ln 3, Col 16 61 character Plain text 100%

PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

```
● PS C:\GoLang> go run guided1.go
pagi 3
pagi
pagi
pagi
● PS C:\GoLang> go run guided1.go
kursi 5
kursi
kursi
kursi
kursi
kursi
● PS C:\GoLang> 
```

## Deskripsi program

Program ini adalah program untuk menampilkan kata yang sama sebanyak angka yang diinputkan oleh user, program ini menggunakan konsep perulangan dengan penanda *boolean* yang logikanya menyerupai struktur *repeat-until* dimana program akan terus mencetak kata selama variabel penanda (*done*) bernilai *false* dan akan berhenti ketika kondisinya berubah menjadi *true*, program ini menggunakan 3 tipe data yaitu string, integer, dan boolean, dimana logika iterasinya menggunakan *increment* (penambahan) pada variabel counter yang dimulai dari 0, perulangan akan berhenti apabila counter sudah bernilai sama atau lebih besar dari jumlah repetisi yang diminta ( $\text{counter} \geq \text{repetitions}$ ), sehingga output yang dihasilkan adalah kata yang tercetak ulang sejumlah input user, contohnya disini user menginput 'pagi 3', maka komputer mencetak 'pagi' pertama, counter bertambah jadi 1, lalu dicek apakah  $1 \geq 3$ ? jawabannya Tidak (*done* tetap *false*), maka ulang lagi, cetak 'pagi' kedua, counter jadi 2, apakah  $2 \geq 3$ ? Tidak, ulang lagi, cetak 'pagi' ketiga, counter jadi 3, apakah  $3 \geq 3$ ?

Ya, maka nilai `done` berubah menjadi `true` dan perulangan pun berhenti, menghasilkan kata 'pagi' yang tercetak sebanyak 3 kali.

## 2. Guided 2

### Source Code

```
package main

import "fmt"

func main() {

    var number int

    var continueLoop bool

    for continueLoop = true; continueLoop; {

        fmt.Scan(&number)

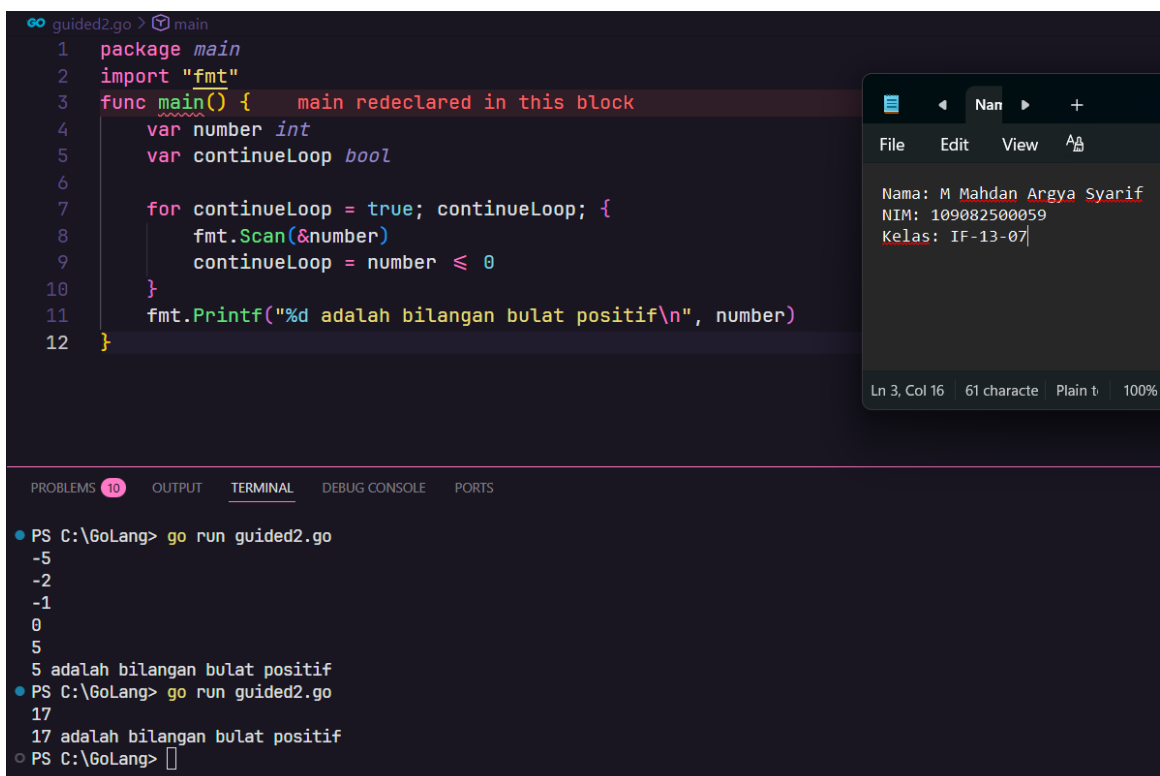
        continueLoop = number <= 0

    }

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

}
```

### Screenshoot program



The screenshot shows a Go IDE with the following components:

- Editor:** Displays the source code for `guided2.go`. A red error message "main redeclared in this block" is visible on line 3, indicating a conflict with the `main` package name.
- Terminal:** Shows the execution of the program. The output is:

```
PS C:\GoLang> go run guided2.go
-5
-2
-1
0
5
5 adalah bilangan bulat positif
PS C:\GoLang> go run guided2.go
17
17 adalah bilangan bulat positif
PS C:\GoLang>
```
- Right Panel:** Displays a sidebar with the following information:
  - File: `Nan`
  - Edit: `View`
  - File Name: `Nama: M Mahdan Argya Syarif`
  - NIM: `109082500059`
  - Kelas: `IF-13-07`

### Deskripsi program

Program ini adalah program untuk memvalidasi input agar user memasukkan bilangan bulat positif, dimana program akan terus melakukan perulangan atau meminta input ulang selama angka yang dimasukkan bukan bilangan positif, program ini menggunakan dua tipe data yaitu integer dan boolean, untuk algoritmanya menggunakan logika perulangan yang dikendalikan oleh variabel `continueLoop`, dimana nilai variabel ini ditentukan oleh kondisi `number <= 0`, artinya jika user menginput angka yang kurang dari atau sama dengan nol maka `continueLoop` bernilai *true* dan program akan berulang kembali, sebagai contoh disini user menginputkan -5, karena `-5 <= 0` adalah *true*, maka program lanjut meminta input, begitu juga saat diinput -2 atau 0, namun ketika user menginputkan 5, logika `5 <= 0` hasilnya adalah *false* (karena 5 itu lebih besar dari 0), sehingga perulangan berhenti dan program akhirnya mencetak bahwa 5 adalah bilangan bulat positif.

### 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)

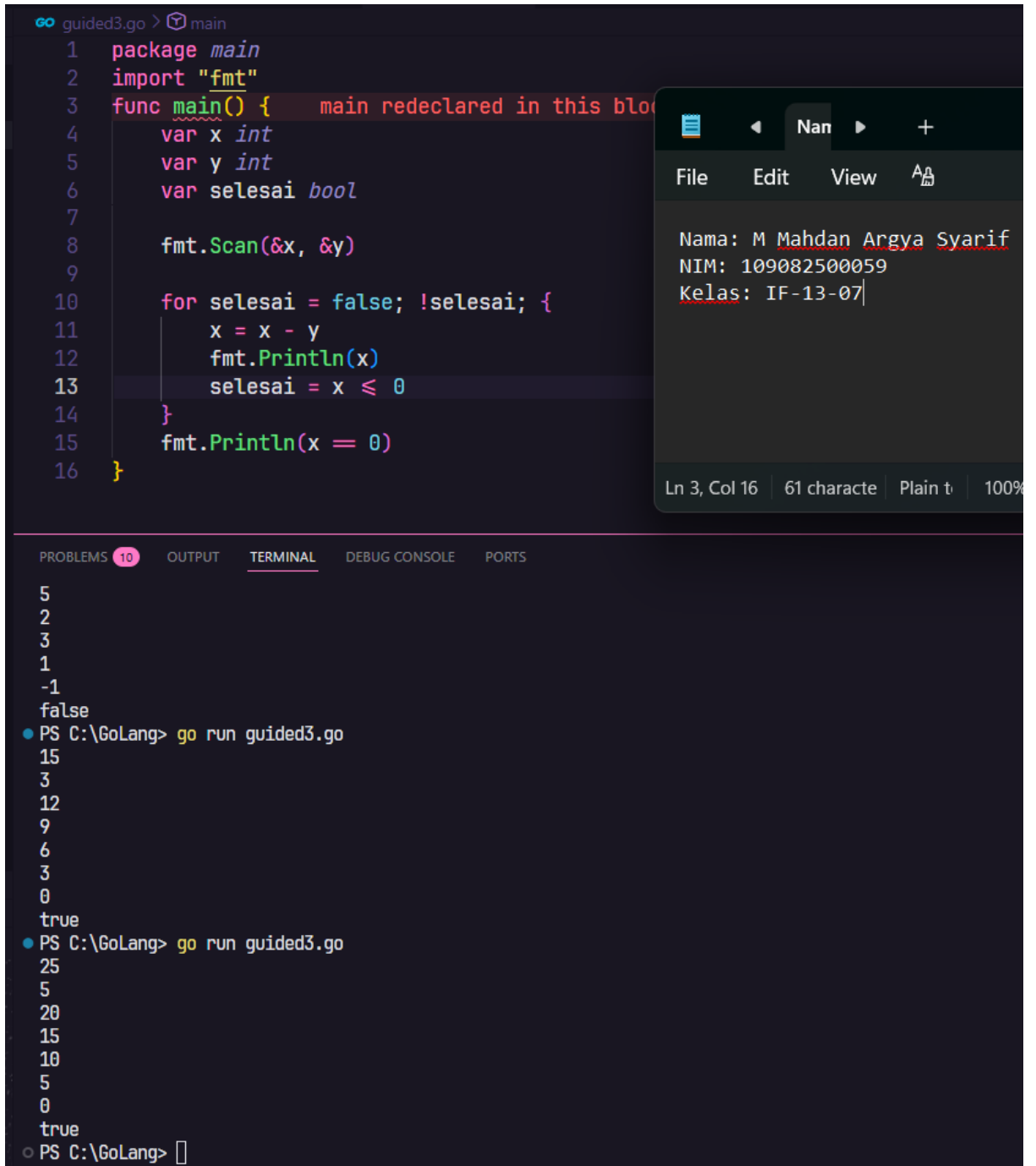
        selesai = x <= 0

    }

    fmt.Println(x == 0)
```

```
}
```

## Screenshoot program



```
guided3.go > main
1 package main
2 import "fmt"
3 func main() { main redeclared in this block
4     var x int
5     var y int
6     var selesai bool
7
8     fmt.Scan(&x, &y)
9
10    for selesai = false; !selesai; {
11        x = x - y
12        fmt.Println(x)
13        selesai = x ≤ 0
14    }
15    fmt.Println(x == 0)
16 }
```

Nama: M Mahdan Argya Syarif  
NIM: 109082500059  
Kelas: IF-13-07

Ln 3, Col 16 | 61 character | Plain text | 100%

PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

```
5
2
3
1
-1
false
● PS C:\GoLang> go run guided3.go
15
3
12
9
6
3
0
true
● PS C:\GoLang> go run guided3.go
25
5
20
15
10
5
0
true
○ PS C:\GoLang> 
```

## Deskripsi program

Program ini adalah program untuk menentukan apakah bilangan x merupakan kelipatan dari bilangan y dengan menggunakan metode pengurangan berulang, program ini menggunakan 2 tipe data yaitu integer untuk bilangan dan Boolean untuk kondisi berhenti, dimana algoritmanya adalah user menginputkan dua angka

kemudian program akan melakukan perulangan `for` yang terus mengurangi nilai `x` dengan `y` ( $x = x - y$ ) sampai kondisi variabel selesai menjadi `true`, yaitu ketika nilai `x` sudah kurang dari atau sama dengan 0, jika hasil pengurangan terakhirnya tepat di angka 0 maka output akhirnya adalah `true` yang artinya valid sebagai kelipatan, namun jika hasilnya minus maka `false`, sebagai contoh pada gambar user menginput `x=15` dan `y=3`, maka  $15-3=12$  apakah  $\leq 0$ ? Tidak, dikurangi lagi  $12-3=9$  apakah  $\leq 0$ ? Tidak, lanjut terus sampai  $3-3=0$  apakah  $\leq 0$ ? Iya (`true`), maka perulangan berhenti dan karena nilai akhirnya 0 maka dicetak `true`, sedangkan jika kita ambil contoh lain misal `x=5` dan `y=2`, maka  $5-2=3$ , lalu  $3-2=1$ , lalu  $1-2 = -1$ , karena  $-1 \leq 0$  maka loop berhenti, tapi karena -1 tidak sama dengan 0 maka output yang dihasilkan adalah `false`.

## TUGAS

### 1. Tugas 1

#### Source code

```
package main

import "fmt"

func main() {

    var bil, digit int

    digit = 0

    fmt.Print("Masukkan bilangan: ")

    fmt.Scan(&bil)

    for bil > 0 {

        bil = bil / 10

        digit++

    }

    fmt.Print("jumlah digit: ", digit)

}
```



## Screenshoot program

```
tugas1.go > main
1 package main
2 import "fmt"
3
4 func main() { main redeclared in this file
5     var bil, digit int
6     digit = 0
7     fmt.Print("Masukkan bilangan: ")
8     fmt.Scan(&bil)
9
10    for bil > 0 {
11        bil = bil / 10
12        digit++
13    }
14    fmt.Print("jumlah digit: ", digit)
15 }
```

File Edit View

Nama: M Mahdan Argya Syarif  
NIM: 109082500059  
Kelas: IF-13-07

Ln 3, Col 16 | 61 character | Plain text | 100%

PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

- PS C:\GoLang> go run tugas1.go  
Masukkan bilangan: 5  
jumlah digit: 1
- PS C:\GoLang> go run tugas1.go  
Masukkan bilangan: 234  
jumlah digit: 3
- PS C:\GoLang> go run tugas1.go  
Masukkan bilangan: 78787  
jumlah digit: 5
- PS C:\GoLang> go run tugas1.go  
Masukkan bilangan: 1894256  
jumlah digit: 7
- PS C:\GoLang>

## Deskripsi program

Program ini adalah program untuk menghitung jumlah digit dari bilangan bulat yang diinputkan oleh user, program ini menggunakan tipe data integer untuk variabel bilangan dan digit, dimana logika utamanya menggunakan perulangan while (dalam Go ditulis for bil > 0) yang akan terus membagi bilangan input dengan angka 10 hingga bilangan tersebut bernilai 0, setiap kali pembagian dilakukan maka variabel digit akan bertambah 1 sebagai penanda jumlah digitnya, sebagai contoh user menginput angka 5, cek kondisi 5 > 0? Ya, maka 5 dibagi 10 hasilnya 0 (karena integer division) dan digit menjadi 1, lalu cek lagi 0 > 0? Tidak, maka loop berhenti dan outputnya 1, contoh lain user input 234, cek 234 > 0? Ya, maka 234 dibagi 10 hasilnya 23 dan digit jadi 1, cek 23 > 0? Ya, maka 23 dibagi 10 hasilnya 2 dan digit jadi 2, cek 2 > 0? Ya, maka 2 dibagi 10 hasilnya 0 dan digit jadi 3, cek 0 > 0? Tidak, maka perulangan berhenti dan hasil akhirnya menampilkan jumlah digit: 3.

## 2. Tugas 2

### Source code

```
package main

import (

    "fmt"

    "math"

)


func main() {

    var bil float64

    fmt.Print("Masukkan bilangan: ")

    fmt.Scan(&bil)


    batasAtas := math.Ceil(bil)


    nilaiAwal := int(bil * 10)

    nilaiLanjutan := int(batasAtas * 10)


    for i := nilaiAwal + 1; i <= nilaiLanjutan; i++){

        if i%10 == 0 {

            fmt.Println(i / 10)

        } else {

            fmt.Printf("%.1f\n", float64(i) / 10.0)

        }

    }

}
```

```
}
```

## Screenshoot program

The screenshot shows a Go program in a code editor and its execution in a terminal. The code defines a `main` function that takes a float64 input, rounds it up to the nearest integer using `math.Ceil`, and then prints a sequence of decimal values from the rounded value down to the next integer below it. For example, if the input is 0.2, it rounds up to 1 and prints 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, and 1.0. If the input is 2.7, it rounds up to 3 and prints 2.8, 2.9, and 3.0.

```
1 package main
2 import (
3     "fmt"
4     "math"
5 )
6
7 func main() {
8     var bil float64
9     fmt.Print("Masukkan bilangan: ")
10    fmt.Scan(&bil)
11
12    batasAtas := math.Ceil(bil)
13
14    nilaiAwal := int(bil * 10)
15    nilaiLanjutan := int(batasAtas * 10)
16
17    for i := nilaiAwal + 1; i <= nilaiLanjutan; i++{
18        if i%10 == 0 {
19            fmt.Println(i / 10)
20        } else {
21            fmt.Printf("%.1f\n", float64(i) / 10.0)
22        }
23    }
24 }
25 }
```

Terminal Output:

```
PS C:\GoLang> go run tugas2.go
Masukkan bilangan: 0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
PS C:\GoLang> go run tugas2.go
Masukkan bilangan: 2.7
2.8
2.9
3
PS C:\GoLang>
```

## Deskripsi program

Program ini adalah program untuk mencetak urutan bilangan desimal dari angka yang diinputkan user hingga mencapai bilangan bulat terdekat di atasnya, program ini menggunakan tipe data `float64` dan library `math.Ceil` untuk membulatkan bilangan ke atas, dimana disini logikanya bilangan input dikalikan 10 terlebih dahulu dan diubah ke integer agar bisa dilakukan perulangan (karena loop integer lebih presisi), variabel `nilaiAwal` didapat dari input dikali 10 dan `nilaiLanjutan` didapat dari hasil `ceil` dikali 10, kemudian perulangan dimulai dari `i` sama dengan `nilaiAwal + 1` hingga `i` kurang dari

atau sama dengan nilaiLanjutan, di dalam loop terdapat kondisi if-else, dimana jika  $i \% 10 == 0$  maka angka tersebut ditampilkan sebagai bilangan bulat, dan jika tidak maka ditampilkan sebagai desimal dengan dibagi 10.0, sebagai contoh user menginput 2.7, maka  $\text{math.Ceil}(2.7)$  adalah 3, lalu dikonversi jadi integer dimana nilaiAwal=27 dan nilaiLanjutan=30, iterasi dimulai dari 28 (27+1), cek apakah  $28 \leq 30$ ? Ya, karena 28 modulus 10 sisanya bukan 0 maka masuk else dan dicetak 2.8, lanjut i jadi 29 dicetak 2.9, lanjut i jadi 30, karena 30 modulus 10 hasilnya 0 (habis dibagi) maka masuk blok if dan dicetak angka 3 (hasil dari  $30/10$ ), sehingga output akhirnya berurutan: 2.8, 2.9, 3.

### 3. Tugas 3

#### Source code

```
package main
import "fmt"

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

    total = 0
    jumlahDonatur = 0

    fmt.Println("Masukkan jumlah uang untuk di
donasikan: ")

    for total <= target {
        var donasi int
        fmt.Scan(&donasi)
        jumlahDonatur += 1

        total += donasi

        fmt.Printf("Donatur %d: menyumbang %d. Total
terkumpul: %d\n ", jumlahDonatur, donasi, total)
    }

    fmt.Printf("Target tercapai! total donasi: %d dari
%d donatur. \n", total, jumlahDonatur)
```

```
}
```

## Screenshoot program

```
1 package main
2 import "fmt"
3
4 func main() {
5     var target, total, jumlahDonatur int
6     fmt.Print("Masukkan target donasi: ")
7     fmt.Scan(&target)
8
9     total = 0
10    jumlahDonatur = 0
11
12    fmt.Println("Masukkan jumlah uang untuk di donasikan: ")
13
14    for total <= target {
15        var donasi int
16        fmt.Scan(&donasi)
17        jumlahDonatur += 1
18
19        total += donasi
20
21        fmt.Printf("Donatur %d: menyumbang %d. Total terkumpul: %d\n", jumlahDonatur, donasi, total)
22    }
23    fmt.Printf("Target tercapai! total donasi: %d dari %d donatur. \n", total, jumlahDonatur)
24 }
```

Terminal Output:

```
PS C:\GoLang> go run tugas3.go
Masukkan target donasi: 300
Masukkan jumlah uang untuk di donasikan:
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:\GoLang> go run tugas3.go
Masukkan target donasi: 500
Masukkan jumlah uang untuk di donasikan:
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:\GoLang> go run tugas3.go
Masukkan target donasi: 200
```

## Deskripsi program

Program ini adalah program untuk mengumpulkan dana donasi hingga mencapai atau melampaui target yang ditentukan, program ini menggunakan tipe data integer untuk variabel target, total, dan jumlahDonatur, dimana nilai awal total dan jumlahDonatur diinisialisasi 0 sebelum masuk ke proses perulangan, algoritma perulangannya menggunakan kondisi  $total \leq target$  yang artinya program akan terus meminta input donasi selama total uang yang terkumpul masih di bawah atau sama dengan target, di dalam loop nilai jumlahDonatur akan selalu bertambah 1 setiap kali ada inputan dan nilai donasi tersebut ditambahkan ke variabel total, sebagai contoh berdasarkan output di layar, user menginput target 300, lalu donatur pertama menyumbang 100, total menjadi 100, apakah  $100 \leq 300$ ? Ya, maka lanjut, lalu donatur kedua menyumbang 50, total menjadi 150, apakah  $150 \leq 300$ ? Ya, lanjut lagi, lalu donatur ketiga menyumbang 200, sehingga total menjadi 350, apakah  $350 \leq 300$ ? Tidak, kondisi bernilai *false* karena total sudah melebihi target, maka perulangan berhenti

dan program menampilkan pesan akhir bahwa target tercapai dengan total donasi 350 dari 3 donatur.