

**LAPORAN PRAKTIKUM ALGORITMA
DAN PEMROGRAMAN 1**

**MODUL 13 – REPEAT UNTIL
ALGORITMA DAN PEMOGRAMAN 1**



Disusun oleh:

NAMA : PRIMATAMA SIGALINGGING

NIM : 109082500076

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)

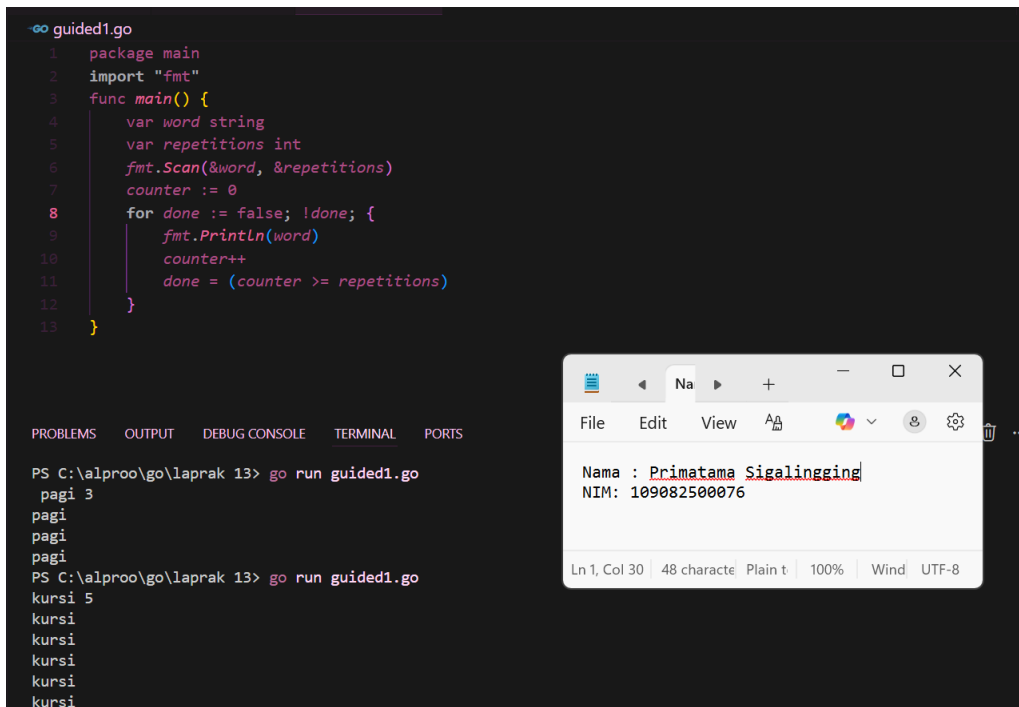
        counter++

        done = (counter >= repetitions)

    }

}
```

Screenshoot program



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

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\alproo\go\laprak 13> go run guided1.go
pagi 3
pagi
pagi
kursi 5
kursi
kursi
kursi
kursi
```

Nama : Primatama Sigalingging
NIM: 109082500076

Ln 1, Col 30 | 48 character Plain text 100% Window UTF-8

Deskripsi program

➤ **Tujuan**

Program ini dibuat untuk menampilkan sebuah kata sebanyak jumlah pengulangan tertentu sesuai dengan input pengguna. Program ini melatih penggunaan perulangan, variabel pencacah (counter), serta logika penghentian perulangan menggunakan kondisi boolean.

➤ **Proses**

1. Program mendeklarasikan variabel word untuk menyimpan kata, repetitions untuk menyimpan jumlah pengulangan, counter sebagai penghitung, dan done sebagai penanda berhenti perulangan.
2. Pengguna memasukkan sebuah kata dan sebuah bilangan bulat yang menyatakan jumlah pengulangan.
3. Program menjalankan perulangan selama kondisi done bernilai false.
4. Pada setiap iterasi:
 - Program menampilkan kata (word) ke layar.
 - Variabel counter ditambahkan satu.
 - Variabel done akan bernilai true jika nilai counter sudah lebih besar atau sama dengan repetitions.
5. Perulangan berhenti ketika jumlah pengulangan yang diminta telah tercapai.

➤ **Kesimpulan**

Program ini menunjukkan bagaimana perulangan dengan kondisi boolean dapat digunakan untuk mengontrol jumlah pengulangan suatu proses. Konsep ini sangat berguna dalam berbagai kasus seperti pencetakan data berulang, simulasi, dan pengendalian alur program berbasis kondisi.

2. Guided 2

Source Code

```
package main

import "fmt"

func main() {

    var number int

    var continueLoop bool

    for continueLoop = true; continueLoop; {

        fmt.Scan(&number)

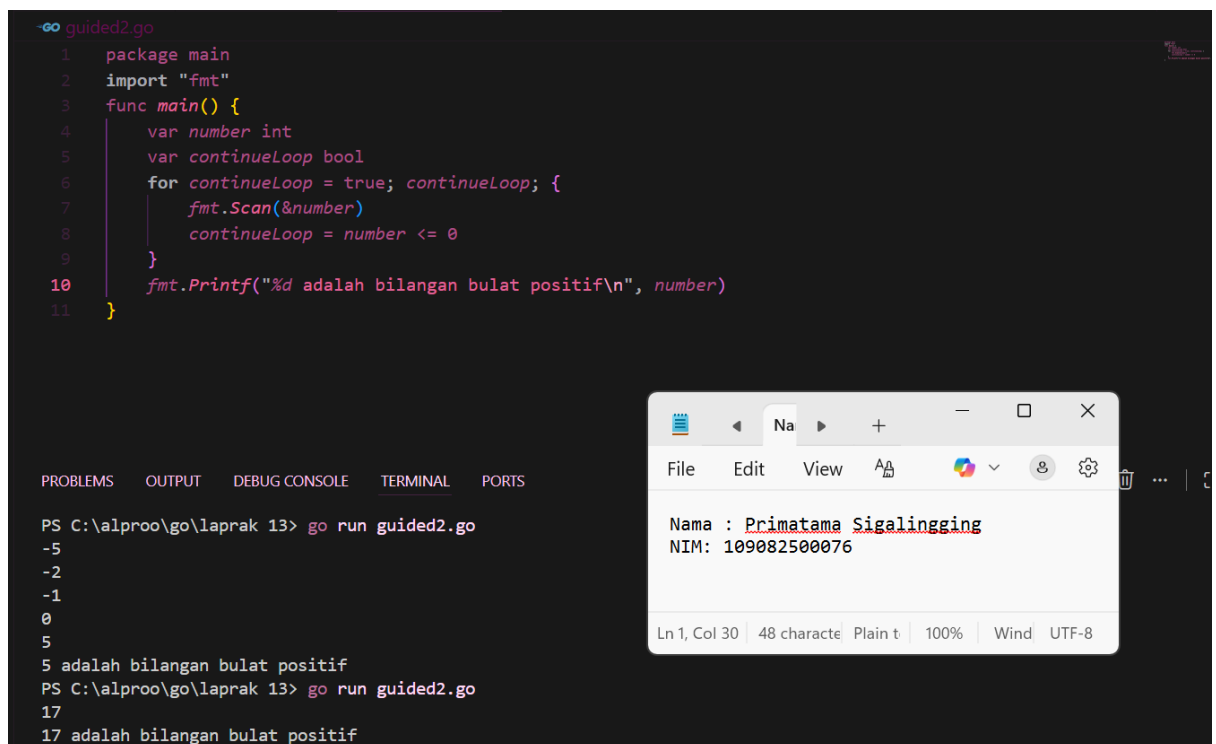
        continueLoop = number <= 0

    }

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

}
```

Screenshoot program



The screenshot displays a Go IDE with a dark theme. The editor window shows the source code for `guided2.go`, which is a Go program that prompts the user to enter an integer. It uses a `for` loop with a boolean condition `continueLoop` to ensure the user enters a positive number. The code is as follows:

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

Below the editor, the **TERMINAL** tab is active, showing the command `go run guided2.go` being executed. The output shows the program prompting for input and accepting the value 5, which is then printed as a positive integer.

```
PS C:\alproo\go\laprak 13> go run guided2.go
-5
-2
-1
0
5
5 adalah bilangan bulat positif
PS C:\alproo\go\laprak 13> go run guided2.go
17
17 adalah bilangan bulat positif
```

Overlaid on the bottom right of the terminal is a small, light-colored window titled "Na" with a standard menu bar (File, Edit, View). It contains the text:

```
Nama : Primatama Sigalingging
NIM: 109082500076
```

The status bar at the bottom of this window indicates the cursor is at line 1, column 30, with 48 characters, in plain text, 100% zoom, and UTF-8 encoding.

Deskripsi program

➤ Tujuan

Program ini dibuat untuk memastikan pengguna memasukkan sebuah bilangan bulat positif. Program ini melatih penggunaan perulangan berbasis kondisi boolean untuk validasi input.

➤ **Proses**

1. Program mendeklarasikan variabel number bertipe integer dan continueLoop bertipe boolean.
2. Program menjalankan perulangan for dengan kondisi continueLoop.
3. Pengguna diminta memasukkan sebuah bilangan.
4. Jika bilangan yang dimasukkan kurang dari atau sama dengan nol, maka perulangan akan terus berjalan.
5. Perulangan berhenti ketika pengguna memasukkan bilangan positif.
6. Program menampilkan bahwa bilangan tersebut adalah bilangan bulat positif.

➤ **Kesimpulan**

Program ini memperlihatkan cara melakukan validasi input menggunakan perulangan dan kondisi logika. Teknik ini penting untuk memastikan data yang diproses oleh program sudah sesuai dengan aturan yang ditentukan.

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

The screenshot shows a Go program named `guided3.go` in a code editor. The program defines a `main` function that declares variables `x` and `y` as integers and `selesai` as a boolean. It uses `fmt.Scan` to read input from the user. A `for` loop runs while `selesai` is `false`, decrementing `x` by `y` and printing the value of `x`. The loop ends when `x` is less than or equal to 0. Finally, it prints a message indicating if `x` is equal to 0.

```
1 package main
2 import "fmt"
3 func main() {
4     var x int
5     var y int
6     var selesai bool
7     fmt.Scan(&x, &y)
8     for selesai = false; !selesai; {
9         x = x - y
10        fmt.Println(x)
11        selesai = x <= 0
12    }
13    fmt.Println(x == 0)
14 }
```

Below the code editor, the terminal shows the command `go run guided3.go` being executed. The output shows the user's input: `5` for `x` and `2` for `y`. The program then prints the values of `x` after each iteration: `3`, `1`, and `-1`. Finally, it prints `false` because `x` is not equal to 0.

Overlaid on the bottom right is a small window titled 'Na' showing the user's name and NIM: `Nama : Primatama Sigalingging` and `NIM: 109082500076`.

Deskripsi program

➤ Tujuan

Program ini dibuat untuk melakukan pengurangan berulang dari sebuah bilangan menggunakan bilangan pengurang tertentu hingga mencapai nol atau kurang. Program ini melatih penggunaan perulangan dan logika penghentian berbasis kondisi.

➤ Proses

1. Program mendeklarasikan variabel `x`, `y`, dan `selesai`.
2. Pengguna memasukkan dua bilangan bulat, yaitu `x` sebagai nilai awal dan `y` sebagai nilai pengurang.
3. Program menjalankan perulangan selama kondisi `selesai` bernilai `false`.
4. Pada setiap iterasi:
 - Nilai `x` dikurangi dengan `y`.
 - Nilai `x` ditampilkan ke layar.

Variabel `selesai` akan bernilai `true` jika `x` kurang dari atau sama dengan nol.

5. Setelah perulangan berhenti, program menampilkan hasil pengecekan apakah nilai akhir `x` sama dengan nol.

➤ Kesimpulan

Program ini menunjukkan bagaimana perulangan dapat digunakan untuk memproses operasi berulang hingga kondisi tertentu tercapai. Konsep ini sering digunakan pada simulasi iteratif dan perhitungan bertahap.

TUGAS

1. Tugas 1

Source code

```
package main

import "fmt"

func main() {
    var n int

    fmt.Print("Masukan bilangan: ")
    fmt.Scan(&n)

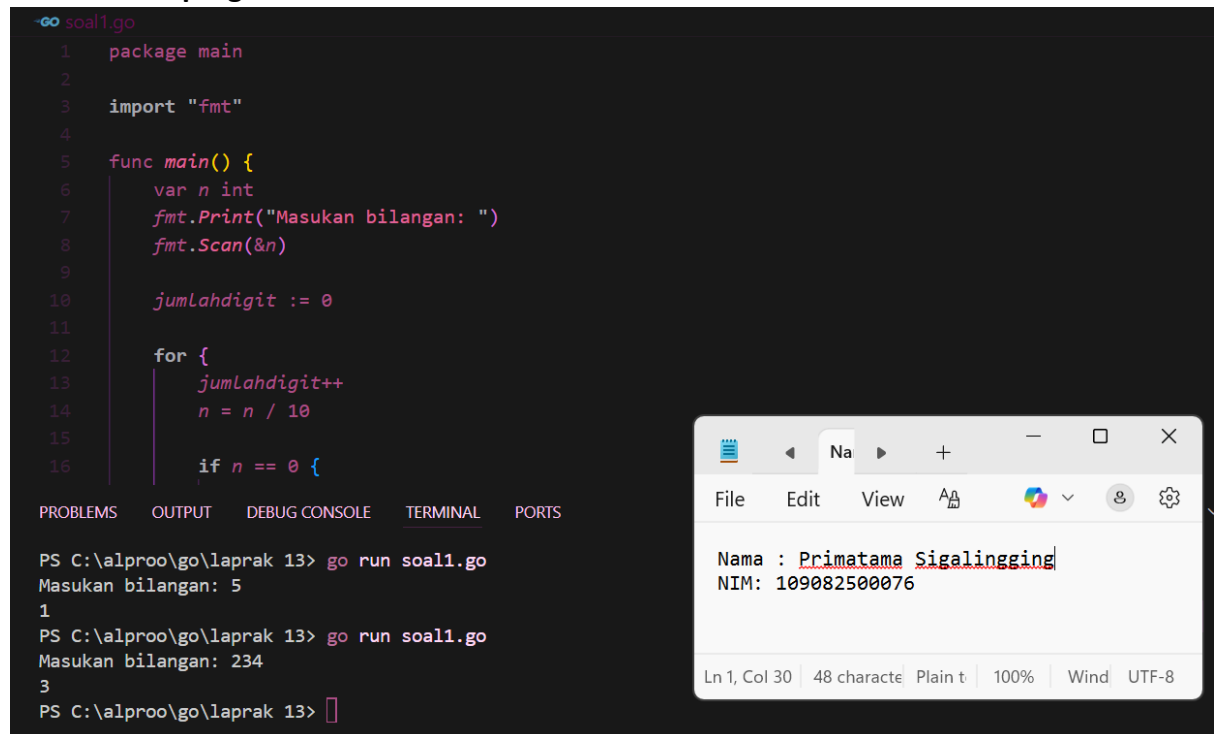
    jumlahdigit := 0

    for {
        jumlahdigit++
        n = n / 10

        if n == 0 {
            break
        }
    }

    fmt.Println(jumlahdigit)
}
```

Screenshoot program



The screenshot shows a Go program in a code editor and its execution in a terminal. The program, named `soal1.go`, is designed to calculate the number of digits in a given integer. It uses a loop to repeatedly divide the number by 10 until it reaches zero, counting each iteration. The terminal shows two test cases: one with the input 5 resulting in 1 digit, and another with the input 234 resulting in 3 digits.

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var n int
7     fmt.Print("Masukan bilangan: ")
8     fmt.Scan(&n)
9
10    jumlahdigit := 0
11
12    for {
13        jumlahdigit++
14        n = n / 10
15
16        if n == 0 {
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\alproo\go\laprak 13> go run soal1.go
Masukan bilangan: 5
1
PS C:\alproo\go\laprak 13> go run soal1.go
Masukan bilangan: 234
3
PS C:\alproo\go\laprak 13>
```

Deskripsi program

➤ Tujuan

Program ini dibuat untuk menghitung jumlah digit dari sebuah bilangan bulat yang dimasukkan oleh pengguna. Program ini melatih penggunaan operasi pembagian bilangan bulat, perulangan tak tentu (infinite loop), serta logika penghentian perulangan berbasis kondisi.

➤ Proses

1. Program mendeklarasikan variabel `n` untuk menyimpan bilangan input dan variabel `jumlahdigit` untuk menghitung jumlah digit.
2. Pengguna memasukkan sebuah bilangan bulat melalui keyboard.
3. Variabel `jumlahdigit` diinisialisasi dengan nilai nol sebagai nilai awal.
4. Program menjalankan perulangan tanpa kondisi awal (perulangan terus-menerus).
5. Pada setiap iterasi:
 - Variabel `jumlahdigit` ditambahkan satu.
 - Nilai `n` dibagi dengan 10 untuk menghilangkan satu digit terakhir.
6. Perulangan akan dihentikan ketika nilai `n` sama dengan nol.
7. Setelah perulangan berhenti, program menampilkan jumlah digit dari bilangan yang dimasukkan.

➤ Kesimpulan

Program ini menunjukkan bagaimana sebuah bilangan dapat diproses secara bertahap dengan menghilangkan digit terakhir menggunakan pembagian bilangan bulat. Teknik ini sangat umum digunakan dalam pengolahan angka, seperti menghitung digit, membalik bilangan, dan analisis digit lainnya.

2. Tugas 2

Source code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var input float64

    fmt.Print("Masukkan bilangan desimal: ")
    fmt.Scan(&input)

    batasAtas := math.Ceil(input)

    for i := input + 0.1; i <= batasAtas; i += 0.1 {
        if i == float64(int64(i)) {
            fmt.Printf("%.0f\n", i)
        } else {
            fmt.Printf("%.1f\n", i)
        }
    }
}
```

Screenshoot program

The screenshot shows a Go program in a code editor and its execution in a terminal. The program, named `soal2.go`, is located at `C:\alproo\go\laprak 13\soal2.go`. It defines a `main` package and imports `fmt` and `math`. The `main` function declares a `float64` variable `input`, prompts the user to enter a decimal number, and scans the input. It then calculates the ceiling of the input using `math.Ceil` and enters a `for` loop that increments the input by 0.1 until it reaches the ceiling. Inside the loop, it checks if the current value is an integer (by comparing it to its integer cast) and prints it with one decimal place if it is not.

```
1 package main
2
3 import (
4     "fmt"
5     "math"
6 )
7 func main() {
8     var input float64
9     fmt.Print("Masukkan bilangan desimal: ")
10    fmt.Scan(&input)
11
12    batasAtas := math.Ceil(input)
13
14    for i := input + 0.1; i <= batasAtas; i += 0.1 {
15        if i == float64(int64(i)) {
16            fmt.Printf("%.0f\n", i)
17        }
18    }
19 }
```

The terminal output shows the program being run with `go run soal2.go`. It prompts for a decimal number, and the user enters `0.2`. The program then prints the sequence of numbers from `0.3` to `1.0` in increments of `0.1`.

```
PS C:\alproo\go\laprak 13> go run soal2.go
Masukkan bilangan desimal: 0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
PS C:\alproo\go\laprak 13>
```

Deskripsi program

➤ Tujuan

Program ini dibuat untuk menampilkan bilangan desimal bertahap dari suatu nilai input hingga mencapai bilangan bulat terdekat di atasnya. Program ini melatih penggunaan perulangan, operasi desimal, serta fungsi pembulatan.

➤ Proses

1. Program mendeklarasikan variabel input bertipe `float64`.
2. Pengguna diminta memasukkan sebuah bilangan desimal.
3. Program menentukan batas atas dengan menggunakan fungsi `math.Ceil`.
4. Program menjalankan perulangan dengan kenaikan 0.1 dari nilai input hingga batas atas.
5. Jika nilai desimal merupakan bilangan bulat, maka nilai tersebut ditampilkan tanpa desimal.
6. Proses ini terus berlanjut hingga batas atas tercapai.

➤ Kesimpulan

Program ini memperlihatkan cara mengolah bilangan desimal menggunakan perulangan dan fungsi matematika. Teknik ini berguna dalam simulasi numerik, iterasi bertahap, dan pengolahan data berbasis nilai pecahan.

3. Tugas 3

Source code

```
package main

import "fmt"

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

    total = 0
    donatur = 0

    for {
        var donasi int
        donatur++

        fmt.Printf("Donatur %d menyumbang: ", donatur)
        fmt.Scan(&donasi)

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

        if total >= target {
            break
        }
    }

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

Screenshoot program

The screenshot shows a Go program named `soal3.go` and its execution output in a terminal window. The program is a simple loop that asks for a target donation amount and then collects donations from three donors until the target is reached.

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var target, total, donatur int
7     fmt.Print("Masukkan target donasi: ")
8     fmt.Scan(&target)
9
10    total = 0
11    donatur = 0
12
13    for {
14        var donasi int
15        donatur++
16
17        fmt.Print("Donatur ")
18        fmt.Print(donatur)
19        fmt.Print(" menyumbang: ")
20        fmt.Scan(&donasi)
21
22        total += donasi
23        fmt.Printf("Total terkumpul: %d\n", total)
24
25        if total >= target {
26            fmt.Printf("Target tercapai! Total donasi: %d dari %d donatur.\n", total, donatur)
27            return
28        }
29    }
30 }
```

Execution output:

```
PS C:\alproo\go\laprak 13> go run soal3.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:\alproo\go\laprak 13>
```

Deskripsi program

➤ Tujuan

Program ini dibuat untuk mensimulasikan proses pengumpulan donasi dari beberapa donatur hingga target donasi tercapai. Program ini melatih penggunaan perulangan tak terbatas (for {}), variabel akumulator, serta kondisi penghentian berdasarkan target tertentu.

➤ Proses

1. Program mendeklarasikan variabel `target`, `total`, dan `donatur` bertipe integer.
2. Pengguna diminta memasukkan nilai target donasi.
3. Nilai awal `total` dan `donatur` diinisialisasi dengan nol.
4. Program menjalankan perulangan `for` tanpa kondisi (loop tak terbatas).
5. Pada setiap iterasi:
 - Variabel `donasi` dideklarasikan untuk menyimpan jumlah donasi dari donatur.
 - Nomor donatur ditambah satu.
 - Pengguna diminta memasukkan jumlah donasi.
 - Nilai donasi ditambahkan ke `total` donasi.
 - `total` donasi sementara ditampilkan.
6. Perulangan akan berhenti ketika `total` donasi telah sama dengan atau melebihi target.
7. Setelah target tercapai, program menampilkan `total` donasi akhir dan jumlah donatur.

➤ **Kesimpulan**

Program ini menunjukkan bagaimana perulangan dan kondisi dapat digunakan untuk menangani proses yang berulang hingga suatu target terpenuhi. Konsep ini sering digunakan dalam simulasi, pengolahan data bertahap, dan sistem akumulasi nilai.