

LAPORAN PRAKTIKUM
Algoritma Pemrograman

MODUL 13
REPEAT-UNTIL



Disusun oleh:

LEONARDO FARRIZ GARCYA

109082530036

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 kata string

    var jumlah int

    var kondisi bool

    fmt.Scan(&kata, &jumlah)

    i:= 0

    for kondisi = false; !kondisi;{

        fmt.Println(kata)

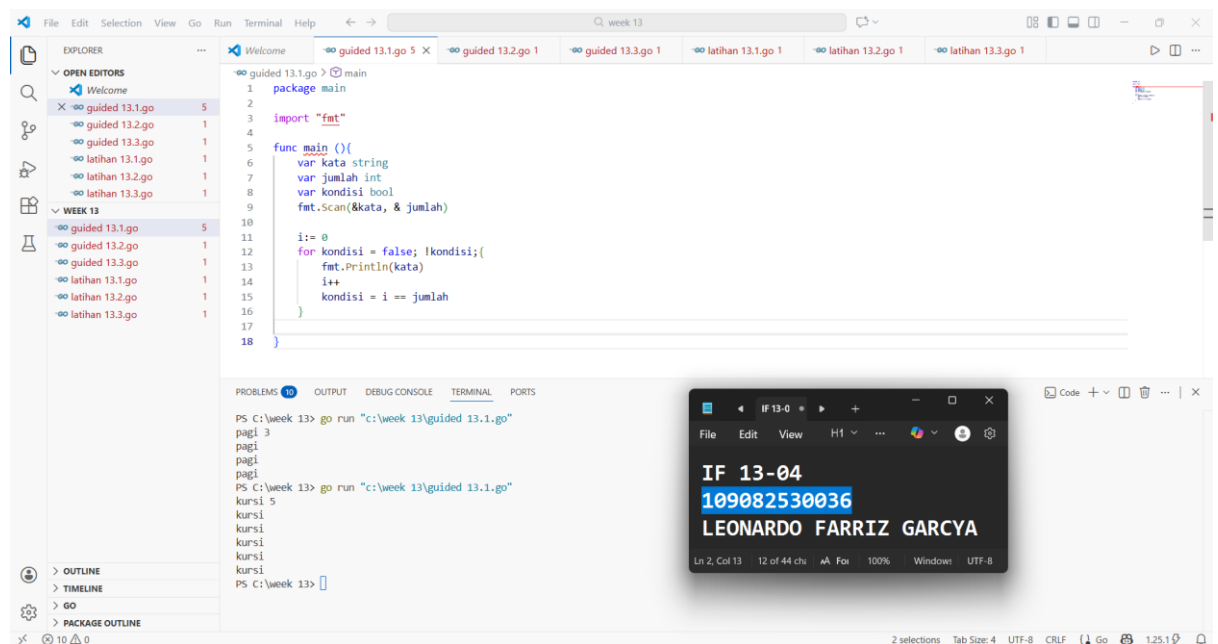
        i++

        kondisi = i == jumlah

    }

}
```

Screenshoot program



Deskripsi program

1.package main

- Bahwa program ini adalah program utama yang bisa langsung dijalankan.
- Tanpa main, kode tidak bisa dieksekusi sebagai aplikasi.

2.import "fmt"

- Format (fmt), digunakan untuk input/output standar.
- Seperti "fmt.Scanln" : untuk membaca input dari user.
- "fmt.Println" : menampilkan output ke layar.

3.func main () { ... }

- Fungsi main adalah untuk titik awal eksekusi program Go.
- Semua intruksi akan berjalan dari fungsi ini

4. var kata string

var jumlah int

var kondisi bool

- Menyatakan variabel kata bertipe string untuk menyimpan teks yang akan di input, variabel jumlah bertipe integer untuk menyimpan angka yang akan di input, variabel kondisi bertipe boolean untuk syarat perulangan (true/false).

5. fmt.Scan(&kata, &jumlah)

- Membaca input dari pengguna dan menyimpannya ke variabel kata dan jumlah.

6. i:= 0

- Menyatakan variabel counter i dan dengan nilai awal 0.

7. for kondisi = false; !kondisi;

- Memulai perulangan for tanpa kondisi umum tetapi berbasis boolean:
- kondisi = false, untuk inialisasi awal kondisi loop bernilai false.
- !kondisi ,selama kondisi bukan true, perulangan terus berjalan.

8. fmt.Println(kata)

- Untuk Mencetak nilai kata setiap kali perulangan.

9. i++

- Untuk menambah nilai i satu per satu setiap loop.

10. kondisi = i == jumlah

- Mengecek apakah jumlah perulangan sudah sama dengan input jumlah. Jika i == jumlah , kondisi jadi true, maka perulangan berhenti.

2. Guided 2

Source Code

```
package main

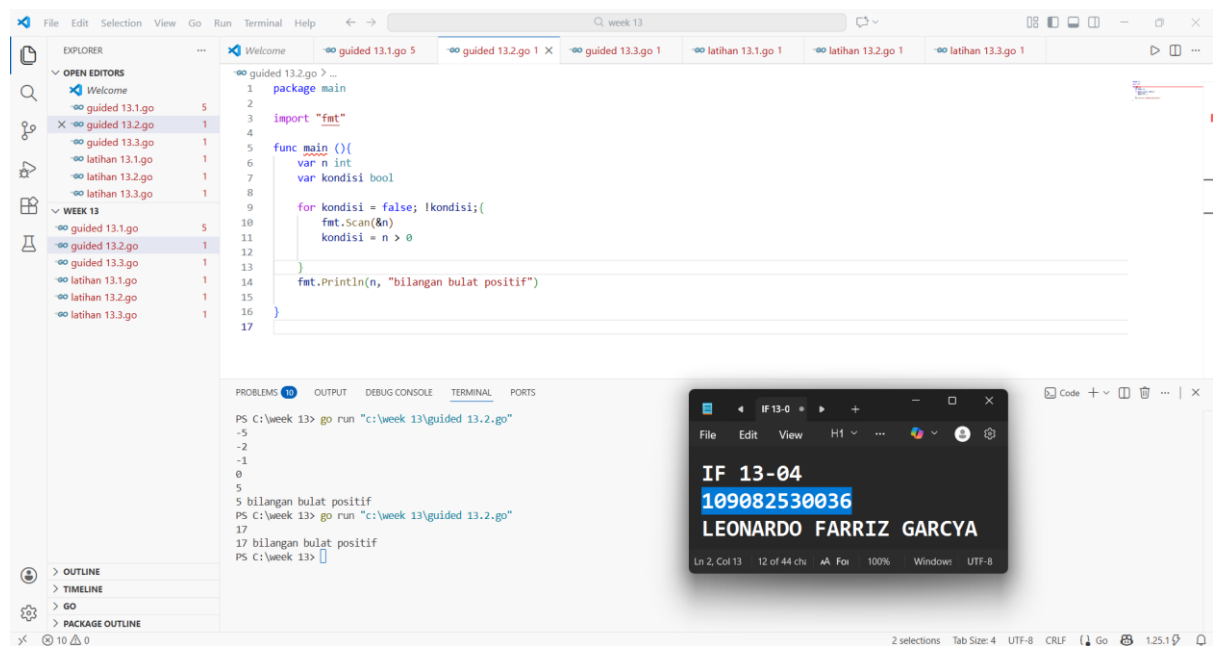
import "fmt"

func main () {
    var n int
    var kondisi bool

    for kondisi = false; !kondisi; {
        fmt.Scan(&n)
        kondisi = n > 0
    }

    fmt.Println(n, "bilangan bulat positif")
}
```

Screenshoot program



Deskripsi program

1.package main

- Bahwa program ini adalah program utama yang bisa langsung dijalankan.
- Tanpa main, kode tidak bisa dieksekusi sebagai aplikasi.

2.import "fmt"

- Format (fmt), digunakan untuk input/output standar.
- Seperti "fmt.Scanln" : untuk membaca input dari user.
- "fmt.Println" : menampilkan output ke layar.

3.func main () { ... }

- Fungsi main adalah untuk titik awal eksekusi program Go.

4. var n int

- Menyatakan variabel n bertipe integer.

5. var kondisi bool

- Menyatakan variabel kondisi bertipe Boolean (true/false).

6. for kondisi = false; !kondisi;

- Jadi kondisi = false adalah nilai awal kondisi yaitu false.
- Dan perulangan akan berjalan selama kondisi bukan true. Karena kondisi = false, !kondisi = true ,maka perulangan dimulai.

7. fmt.Scan(&n)

- Meminta inputan secara terus menerus jika false.

8. kondisi = $n > 0$

- Untuk mengecek jika n lebih besar dari 0, maka kondisi menjadi true.
- Jika input ≤ 0 , maka kondisi tetap false, perulangan berlanjut.

9. `fmt.Println(n, "bilangan bulat positif")`

- Setelah perulangan berhenti (karena pengguna memasukkan angka positif), maka program akan mencetak ke layar.

3. Guided 3

Source Code

```
package main

import "fmt"

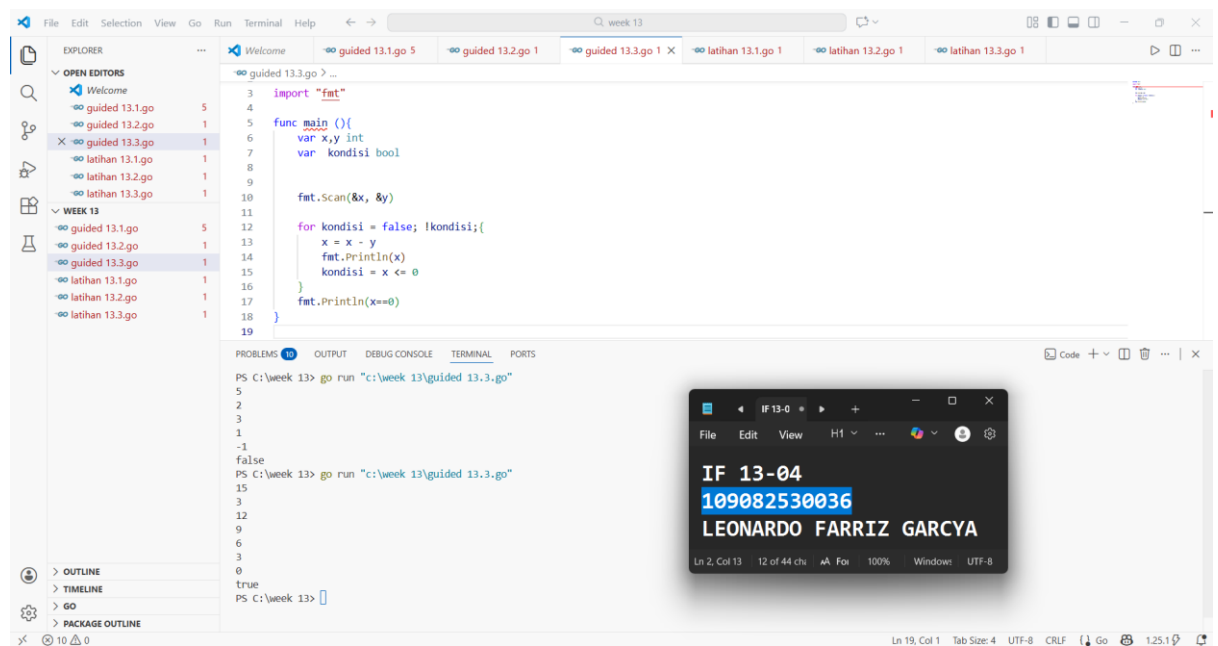
func main () {
    var x,y int
    var kondisi bool

    fmt.Scan(&x, &y)

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

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

Screenshoot program



Deskripsi program

1.package main

- Bahwa program ini adalah program utama yang bisa langsung dijalankan.
- Tanpa main, kode tidak bisa dieksekusi sebagai aplikasi.

2.import "fmt"

- Format (fmt), digunakan untuk input/output standar.
- Seperti "fmt.Scanln" : untuk membaca input dari user.
- "fmt.Println" : menampilkan output ke layar.

3.func main () { ... }

- Fungsi main adalah untuk titik awal eksekusi program Go.
- Semua intruksi akan berjalan dari fungsi ini.

4. var x,y int

- Menyatakan variabel x dan y bertipe integer.

5. var kondisi bool

- Menyatakan variabel kondisi bertipe Boolean (true/false).

6. fmt.Scan(&x, &y)

- Membaca input dari pengguna dan menyimpannya ke variabel x dan y.

7. for kondisi = false; !kondisi;{

- Membuat perulangan dengan nilai awal kondisi = false.
- Selama kondisi false, perulangan akan terus berjalan.

8. $x = x - y$

- Untuk mengurangi x dengan y setiap kali loop berjalan.

9. `fmt.Println(x)`

- Menampilkan nilai x setelah dikurangi.

10. `kondisi = x <= 0`

- Mengubah nilai kondisi berdasarkan hasil pengurangan:

- Jika $x \leq 0$, maka kondisi = true, loop berhenti.

- Jika $x > 0$, maka kondisi = false, loop lanjut.

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

Menampilkan hasil perbandingan:

- true jika $x = 0$.

- false jika x kurang dari 0.

TUGAS

1. Tugas 1

Source code

```
package main

import "fmt"

func main() {
    var a, t int

    fmt.Scan(&a)

    for kondisi := false; !kondisi; {
        t++

        a /= 10

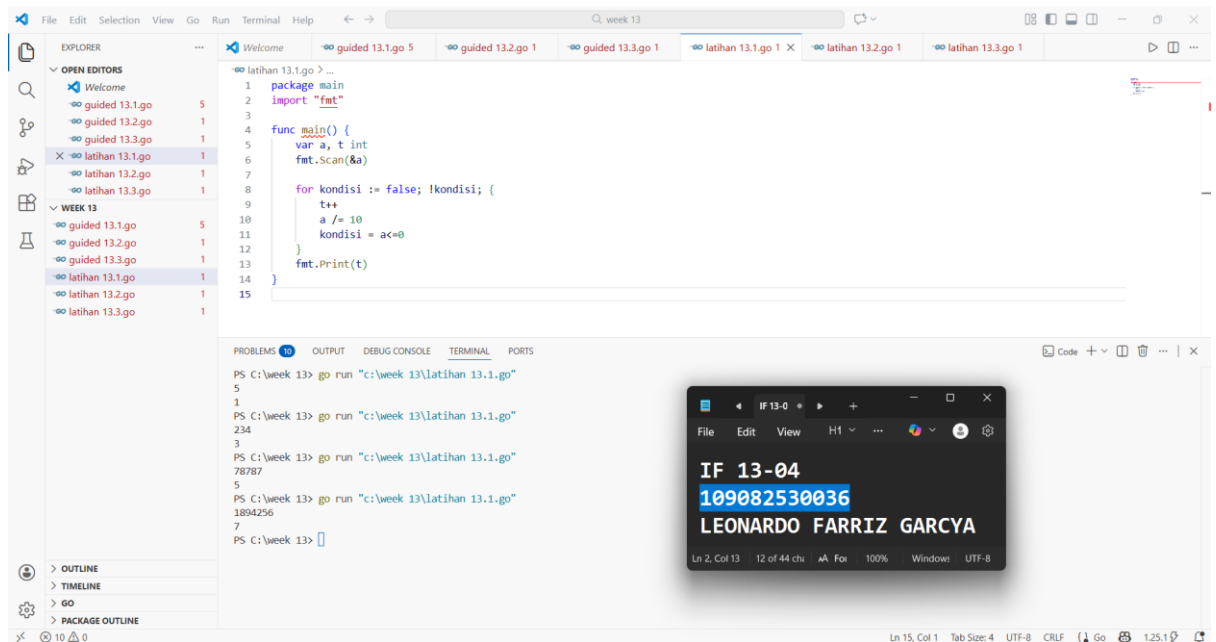
        kondisi = a <= 0
    }

    fmt.Print(t)
```



```
}
```

Screenshoot program



Deskripsi program

1. package main

- Bahwa program ini adalah program utama yang bisa langsung dijalankan.
- Tanpa main, kode tidak bisa dieksekusi sebagai aplikasi.

2. import "fmt"

- Format (fmt), digunakan untuk input/output standar.
- Seperti "fmt.Scanln" : untuk membaca input dari user.
- "fmt.Println" : menampilkan output ke layar.

3. func main () { ... }

- Fungsi main adalah untuk titik awal eksekusi program Go.
- Semua intruksi akan berjalan dari fungsi ini.

4. var a ,t int

- Menyatakan variabel a dan t bertipe integer.

5. fmt.Scan(&a)

- Membaca input dari pengguna dan menyimpannya ke variabel a dan t.

6. for kondisi := false; !kondisi;{

- Menyatakan variabel kondisi dan Membuat perulangan dengan nilai awal kondisi = false.
- Selama kondisi false, perulangan akan terus berjalan.

7. t++

- Menambah nilai t sebanyak 1 dan ini menghitung berapa kali angka a dibagi 10.

8. a /= 10

- Setiap iterasi, a dibagi 10. Ini menyusutkan angka dari belakang, sehingga bisa menghitung jumlah digit.

9. kondisi = a <= 0

- Mengubah nilai kondisi, jika a <= 0, maka kondisi = true, maka perulangan berhenti.

- Jika a > 0, kondisi = false, maka perulangan lanjut

10. fmt.Print(t)

Menampilkan nilai t ke layar.

2. Tugas 2

Source code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var a float64
    fmt.Scan(&a)

    target := math.Ceil(a)

    for {
        a += 0.1
        if a >= target {
            a = target
            fmt.Printf("%.1f\n", a)
```

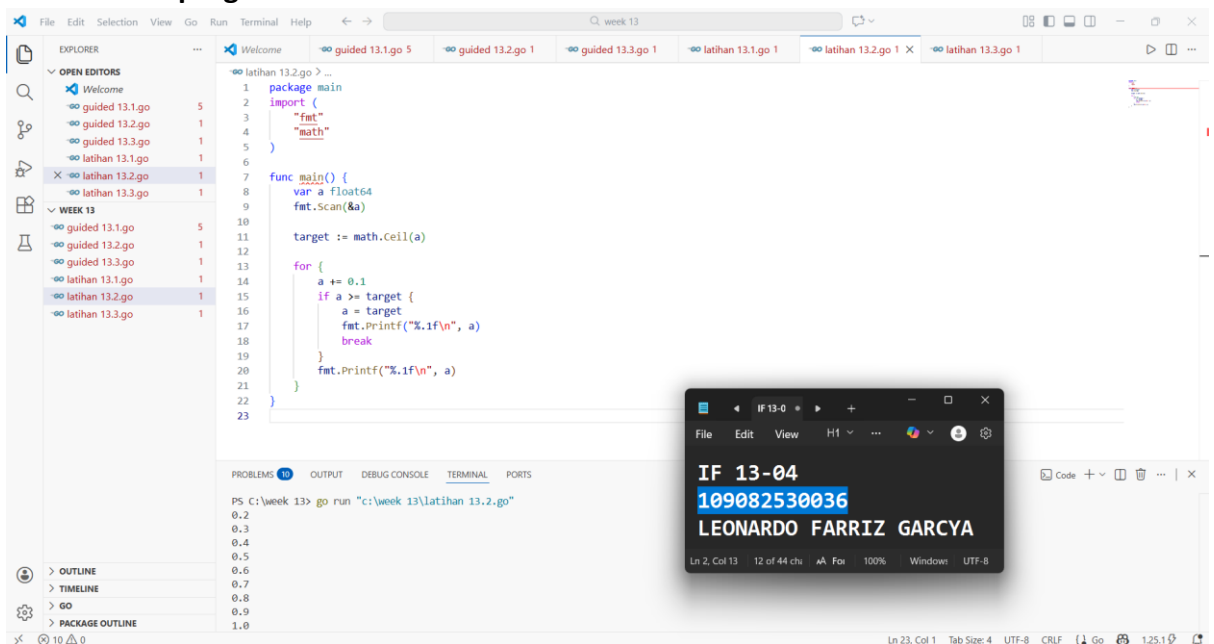
```

        break
    }

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

```

Screenshoot program



Deskripsi program

1. package main

- Bahwa program ini adalah program utama yang bisa langsung dijalankan.
- Tanpa main, kode tidak bisa dieksekusi sebagai aplikasi.

2. import "fmt" "math"

- Format (fmt), digunakan untuk input/output standar.
- math, digunakan untuk fungsi matematika

3. func main () { ... }

- Fungsi main adalah untuk titik awal eksekusi program Go.
- Semua intruksi akan berjalan dari fungsi ini.

4. var a float64

- Menyatakan variabel a dan t bertipe float64 (desimal).

5. fmt.Scan(&a)

- Membaca input dari pengguna dan menyimpannya ke variabel a.

6. target := math.Ceil(a)

- math.Ceil(a) untuk membulatkan a ke atas ke bilangan bulat terdekat.

- Nilai hasil Ceil disimpan ke variabel target.

7. for {

- Membuka perulangan *infinite loop* (tak bersyarat).

8. a += 0.1

- Menambahkan angka 0.1 ke variabel a setiap iterasi.

9. if a >= target {

- Mengecek apakah nilai a sudah mencapai atau melewati batas pembulatan (target).

10. a = target

- Menjadikan nilai a sama persis dengan target, agar tidak lewat atau terjadi error pecahan floating.

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

- Menampilkan nilai a ke layar dengan format 1 angka di belakang koma.

12. break

- Untuk Menghentikan loop karena target sudah tercapai.

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

- Jika kondisi if tidak terpenuhi (berarti a masih < target), nilai a dicetak.

3. Tugas 3

Source code

```
package main
import "fmt"

func main () {
    var target, total, donatur int
    fmt.Scan(&target)

    jumlah:=0

    for kondisi :=true; kondisi;{
        fmt.Scan(&donatur)
```

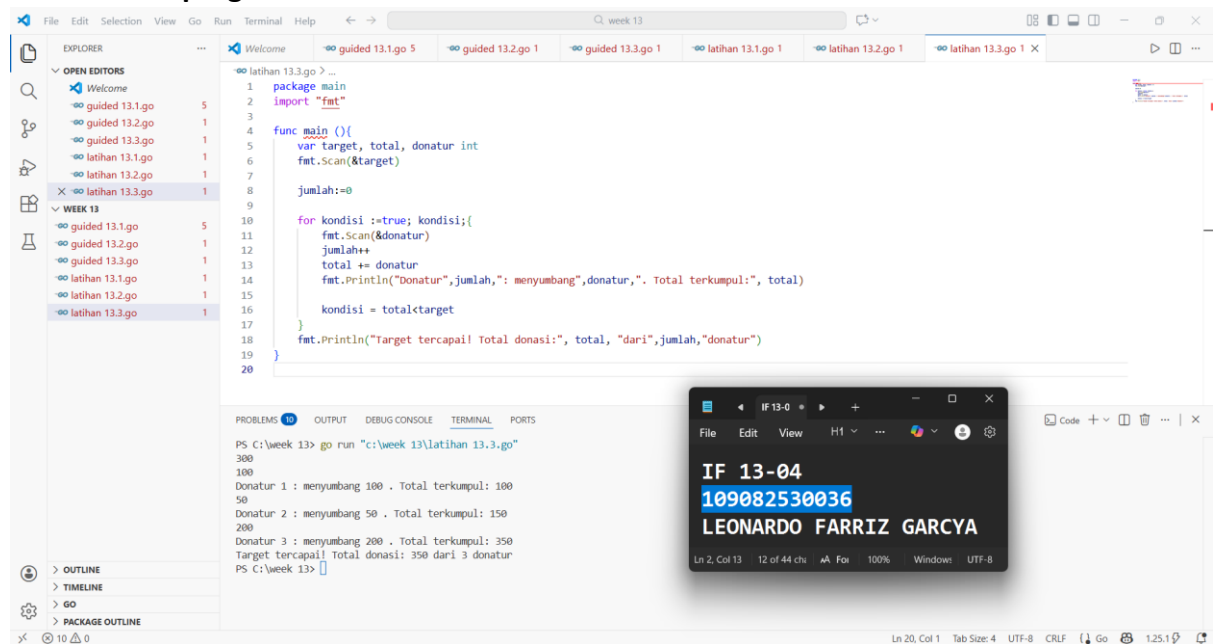
```

        jumlah++
        total += donatur
        fmt.Println("Donatur", jumlah, ":
        menyumbang", donatur, ". Total terkumpul:", total)

        kondisi = total < target
    }
    fmt.Println("Target tercapai! Total donasi:",
    total, "dari", jumlah, "donatur")
}

```

Screenshoot program



Deskripsi program

1. package main

- Bahwa program ini adalah program utama yang bisa langsung dijalankan.
- Tanpa main, kode tidak bisa dieksekusi sebagai aplikasi.

2. import "fmt"

- Format (fmt), digunakan untuk input/output standar.
- Seperti "fmt.Scanln" : untuk membaca input dari user.
- "fmt.Println" : menampilkan output ke layar.

3. func main () { ... }

- Fungsi main adalah untuk titik awal eksekusi program Go.

- Semua intruksi akan berjalan dari fungsi ini.

4. var target, total, donatur int

- Menyatakan variabel target,total,donatur bertipe integer.

5. fmt.Scan(&target)

- Membaca input dari pengguna dan menyimpannya ke variabel target.

6. jumlah:=0

- Membuat variabel baru jumlah, digunakan untuk menghitung berapa orang yang sudah berdonasi.

7. for kondisi := true; kondisi;

- Membuat perulangan dengan kondisi Boolean, nilai awal kondisi = true.

- Perulangan terus berjalan selama kondisi = true.

8. fmt.Scan(&donatur)

- Membaca input donasi dari setiap donatur.

9. jumlah++

- Untuk menambah hitungan jumlah donatur sebesar 1.

10. total += donatur

- Untuk menambahkan nilai donasi ke dalam total akumulasi.

11. fmt.Println("Donatur", jumlah, ": menyumbang", donatur, ". Total terkumpul:", total)

- Mencetak pesan untuk menampilkan nomor donatur ke berapa, nominal donasinya, dan total donasi sementara yang sudah terkumpul

12. kondisi = total < target

- Mengubah nilai kondisi menjadi:

 true, jika total masih kurang dari target.

 false, jika total sudah mencapai / melebihi target, maka loop berhenti.

13. fmt.Println("Target tercapai! Total donasi:", total, "dari", jumlah, "donatur")

- Setelah keluar dari perulangan, program menampilkan pesan akhir:

total donasi yang terkumpul.

jumlah total donatur yang berpartisipasi.