

LAPORAN PRAKTIKUM

Algoritma Pemrograman

**MODUL 13
REPEAT-UNTIL**



Disusun oleh:

RAYSA RAHMA IRAHIM

109082500167

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)

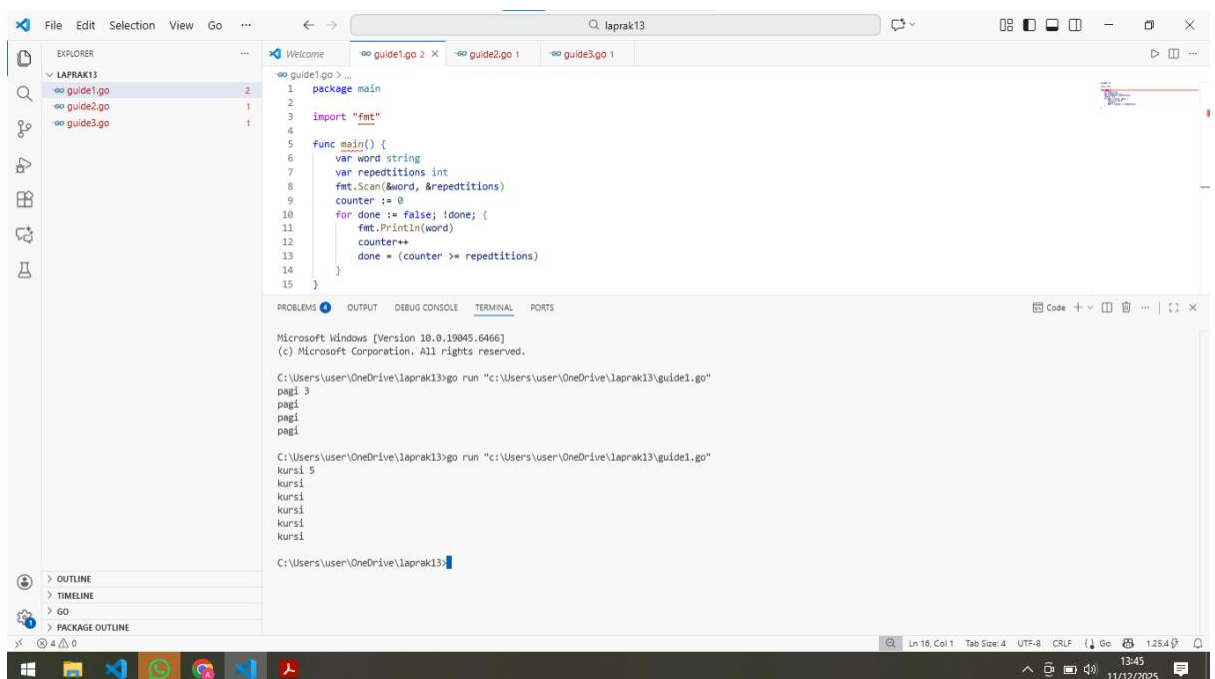
        counter++

        done = (counter >= repetitions)

    }

}
```

Screenshoot Program



Deskripsi program

1. package main

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi `main()` sebagai titik awal berjalannya program. Tanpa `package main`, program tidak bisa dijalankan sebagai aplikasi.

2. import "fmt"

Berfungsi supaya program bisa membaca input dari keyboard menggunakan `Scan` dan menampilkan output ke layar menggunakan `Print` atau `Println`. Tanpa mengimpor `fmt`, perintah input-output seperti `fmt.Scan` atau `fmt.Println` tidak dapat digunakan. Paket `fmt` merupakan singkatan dari `format`, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

3. func main()

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layar. Tanpa `main()`, program tidak akan bisa dijalankan.

4. Tanda buka dan tutup kurung kurawal { }

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi `main`.

5. var word string

Baris `var word string` mendeklarasikan variabel bernama `word` dengan tipe data `string`. Variabel ini digunakan sebagai tempat penyimpanan kata yang dimasukkan. Karena kata merupakan data berbentuk teks, tipe `string` menjadi pilihan yang sesuai dan dapat menampung berbagai bentuk karakter.

6. var repedititions int

Ini mendeklarasikan variabel `repeditions` dengan tipe `integer`. Variabel ini digunakan untuk menyimpan jumlah pengulangan yang diminta. Tipe `integer` dipilih karena jumlah pengulangan merupakan bilangan bulat yang tidak menggunakan nilai desimal.

7. fmt.Scan(&word, &repeditions)

`fmt.Scan` digunakan untuk membaca input yang dimasukkan. Tanda `&` (ampersand) menunjukkan bahwa nilai hasil input akan disimpan langsung ke dalam alamat memori variabel `word` dan `repeditions`.

8. counter := 0

Baris ini mendeklarasikan variabel counter sekaligus memberikan nilai awal 0. Variabel counter sebagai penghitung jumlah pengulangan yang sudah dilakukan oleh program. Nilai counter bertambah setiap kali kata dicetak ke layar sehingga program dapat membandingkan apakah proses pengulangan sudah mencapai batas yang telah dimasukkan.

9. for done := false; !done; { ... }

Struktur perulangan inibentuk khusus dari for pada Go yang menggunakan variabel done sebagai penentu kapan pengulangan harus berhenti. Pada bagian inisialisasi, done diatur bernilai false, yang berarti perulangan akan terus berjalan. Kondisi !done menunjukkan bahwa selama done masih bernilai false, blok perulangan akan terus dieksekusi.

10. fmt.Println(word)

Setiap kali loop dijalankan, program mengeksekusi fmt.Println(word) untuk mencetak kata yang dimasukkan. Perintah ini memastikan bahwa kata tersebut tampil pada layar sebanyak jumlah pengulangan yang ditentukan. Setiap pencetakan dilakukan pada baris baru karena fungsi Println secara otomatis menambahkan baris baru.

11. counter++

Pada baris ini, nilai counter ditambah satu setiap kali perulangan dilakukan. Hal ini penting untuk melacak sudah berapa kali kata dicetak. Dengan terus meningkatkan nilai counter, program dapat membandingkannya dengan jumlah pengulangan yang dimasukkan sehingga logika penghentian perulangan berjalan dengan benar.

12. done = (counter >= repetitions)

Baris ini merupakan inti dari logika penghentian loop. Ketika nilai counter sudah mencapai atau melebihi jumlah pengulangan yang dimasukkan, ekspresi (counter >= repetitions) akan bernilai true. Nilai true ini kemudian diberikan pada variabel done, yang membuat kondisi !done pada perulangan menjadi false. Akibatnya, perulangan berhenti dan program selesai dijalankan.

2. Guided 2

Source Code

```
package main

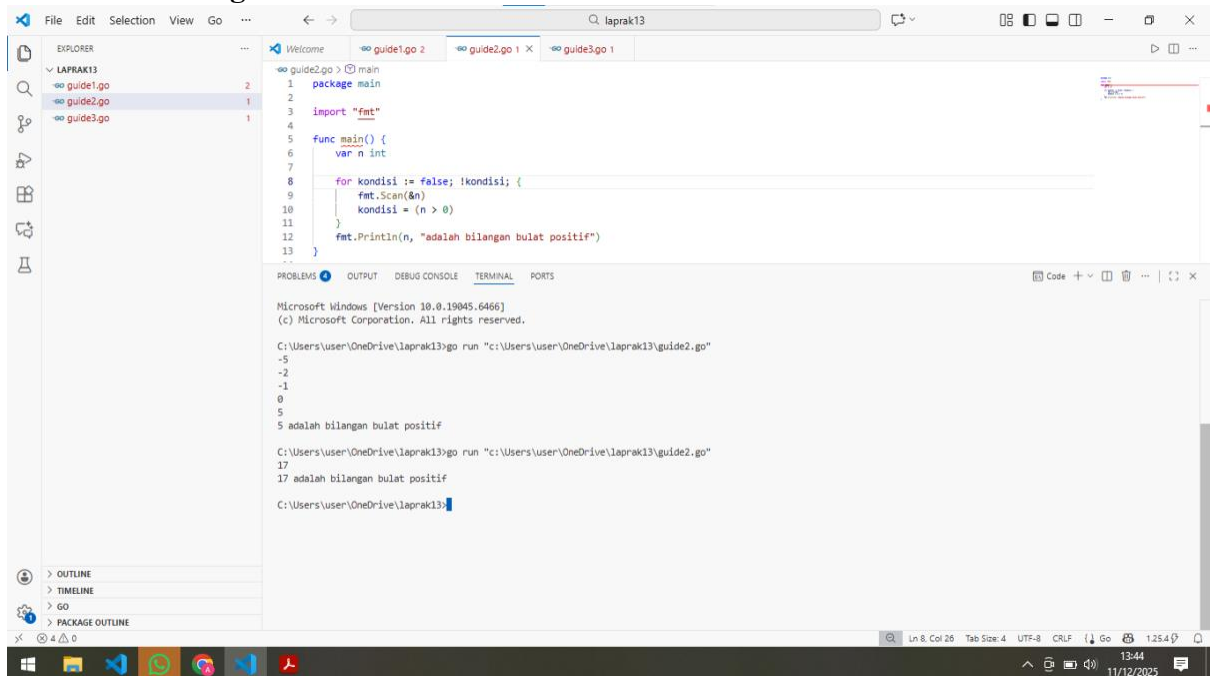
import "fmt"

func main() {
    var n int

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

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

Screenshoot Program



Deskripsi program

1. package main

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi `main()` sebagai titik awal berjalannya program. Tanpa `package main`, program tidak bisa dijalankan sebagai aplikasi.

2. import "fmt"

Berfungsi supaya program bisa membaca input dari keyboard menggunakan `Scan` dan menampilkan output ke layar menggunakan `Print` atau `Println`. Tanpa mengimpor `fmt`, perintah input-output seperti `fmt.Scan` atau `fmt.Println` tidak dapat digunakan. Paket `fmt` merupakan singkatan dari `format`, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

3. func main()

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layar. Tanpa `main()`, program tidak akan bisa dijalankan.

4. Tanda buka dan tutup kurung kurawal { }

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi `main`.

5. var n int

Baris `var n int` mendeklarasikan sebuah variabel bernama `n` dengan tipe data integer. Variabel ini disiapkan sebagai tempat penyimpanan angka yang dimasukkan. Karena yang dibutuhkan adalah bilangan bulat positif, tipe integer adalah tipe data yang tepat untuk menampung nilai tersebut.

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

Bagian ini merupakan struktur perulangan yang dikendalikan oleh variabel `kondisi`. Pada awal perulangan, `kondisi` diberi nilai `false` sehingga ekspresi `!kondisi` bernilai `true` dan perulangan dimulai. Loop ini terus berjalan selama `kondisi` masih bernilai `false`.

7. fmt.Scan(&n)

`fmt.Scan(&n)` digunakan untuk membaca input yang dimasukkan dan menyimpannya ke dalam variabel `n`. Tanda `&` menunjukkan bahwa nilai yang dimasukkan pengguna akan ditempatkan langsung pada alamat memori variabel tersebut.

8. kondisi = (n > 0)

Setelah memasukkan angka, baris ini mengecek apakah angka tersebut merupakan bilangan bulat positif. Ekspresi `(n > 0)` akan menghasilkan nilai `true` apabila angka yang dimasukkan lebih besar dari nol. Nilai `true` ini kemudian disimpan dalam variabel `kondisi`. Jika memasukkan angka kurang dari atau sama dengan nol, ekspresi akan bernilai `false` sehingga perulangan tetap berjalan dan program meminta input kembali.

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

Setelah loop selesai, baris ini dijalankan untuk menampilkan pesan bahwa nilai yang dimasukkan pengguna merupakan bilangan bulat positif. Program akan cetak nilai `n` yang sudah dicek, diikuti dengan teks “adalah bilangan bulat positif”.

3. Guided 3

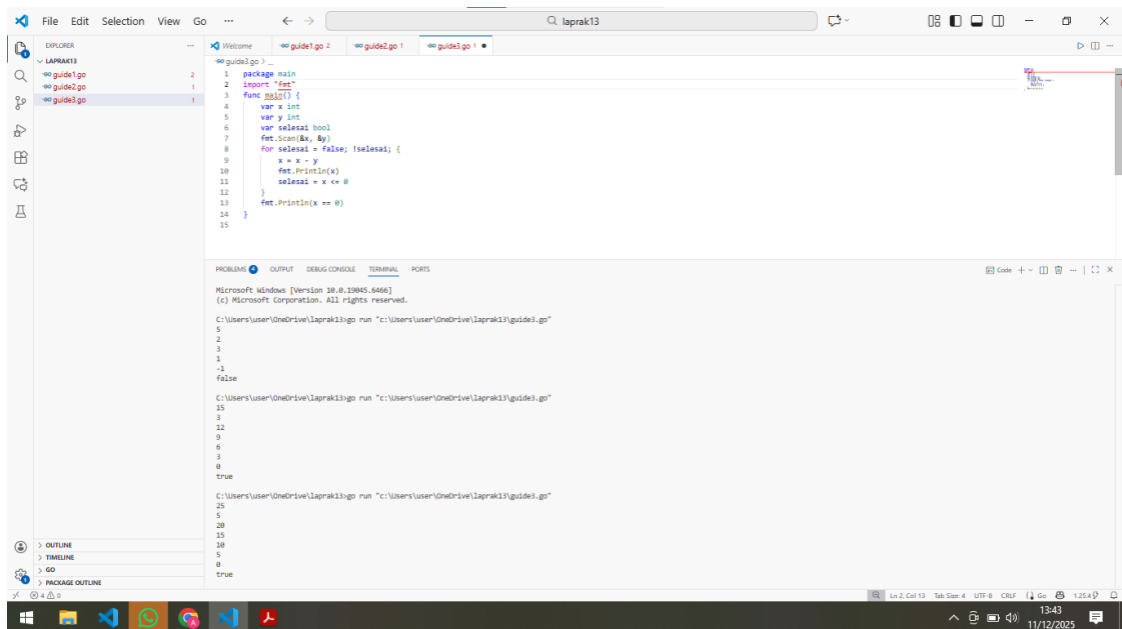
Source Code

```
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



Deskripsi program

1. package main

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi `main()` sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

2. import "fmt"

Berfungsi supaya program bisa membaca input dari keyboard menggunakan `Scan` dan menampilkan output ke layar menggunakan `Print` atau `Println`. Tanpa mengimpor `fmt`, perintah input-output seperti `fmt.Scan` atau `fmt.Println` tidak dapat digunakan. Paket `fmt` merupakan singkatan dari `format`, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

3. func main()

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layer. Tanpa `main()`, program tidak akan bisa dijalankan.

4. Tanda buka dan tutup kurung kurawal { }

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi `main`.

5. var x int

Baris ini mendeklarasikan variabel `x` dengan tipe data integer. Variabel ini digunakan sebagai nilai awal yang akan dikurangi secara berulang. Karena `x` menyimpan bilangan bulat, tipe integer adalah tipe yang tepat untuk variabel ini.

6. var y int

Variabel `y` juga dideklarasikan sebagai integer dan berfungsi sebagai nilai pengurang. Setiap iterasi, nilai ini akan dikurangkan dari `x`. Penggunaan tipe integer memastikan proses pengurangan berlangsung dalam bilangan bulat tanpa nilai desimal.

7. var selesai bool

Baris ini mendeklarasikan variabel `selesai` dengan tipe boolean. Variabel ini berfungsi sebagai penanda kondisi untuk menghentikan perulangan. Ketika `selesai` bernilai `false`, perulangan akan terus berjalan. Sebaliknya, jika `selesai` berubah menjadi `true`, loop berhenti.

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

Ini digunakan untuk membaca dua input dari pengguna, yaitu nilai awal `x` dan nilai pengurang `y`. Tanda `&` menunjukkan bahwa input yang dimasukkan akan disimpan langsung pada alamat variabel tersebut.

9. `for selesai = false; !selesai; { ... }`

Struktur perulangan ini menginisialisasi variabel selesai dengan nilai false. Selama kondisi !selesai masih bernilai true, perulangan akan terus berjalan.

10. `x = x - y`

Pada setiap iterasi, program mengurangi nilai y dari x dan menyimpan hasilnya kembali ke variabel x. Perintah ini merupakan inti proses pengurangan berulang. Nilai x akan semakin kecil pada setiap siklus perulangan.

11. `fmt.Println(x)`

Setelah terjadi pengurangan, program menampilkan nilai x yang terbaru ke layar. Ini memungkinkan kita melihat bagaimana nilai x berubah pada setiap iterasi sampai program berhenti.

12. `selesai = x <= 0`

Instruksi ini mengecek apakah nilai x telah mencapai 0 atau kurang dari 0. Jika kondisi tersebut terpenuhi, ekspresi `x <= 0` bernilai true dan diberikan ke variabel selesai. Perubahan ini menyebabkan !selesai menjadi false pada iterasi berikutnya sehingga loop berhenti.

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

Setelah perulangan selesai, program menampilkan hasil evaluasi apakah nilai akhir x sama dengan 0. Jika x benar-benar berakhir pada 0, program mencetak true. Jika x berhenti pada nilai negatif, program mencetak false.

TUGAS

1. Tugas 1

Source code

```
package main

import "fmt"

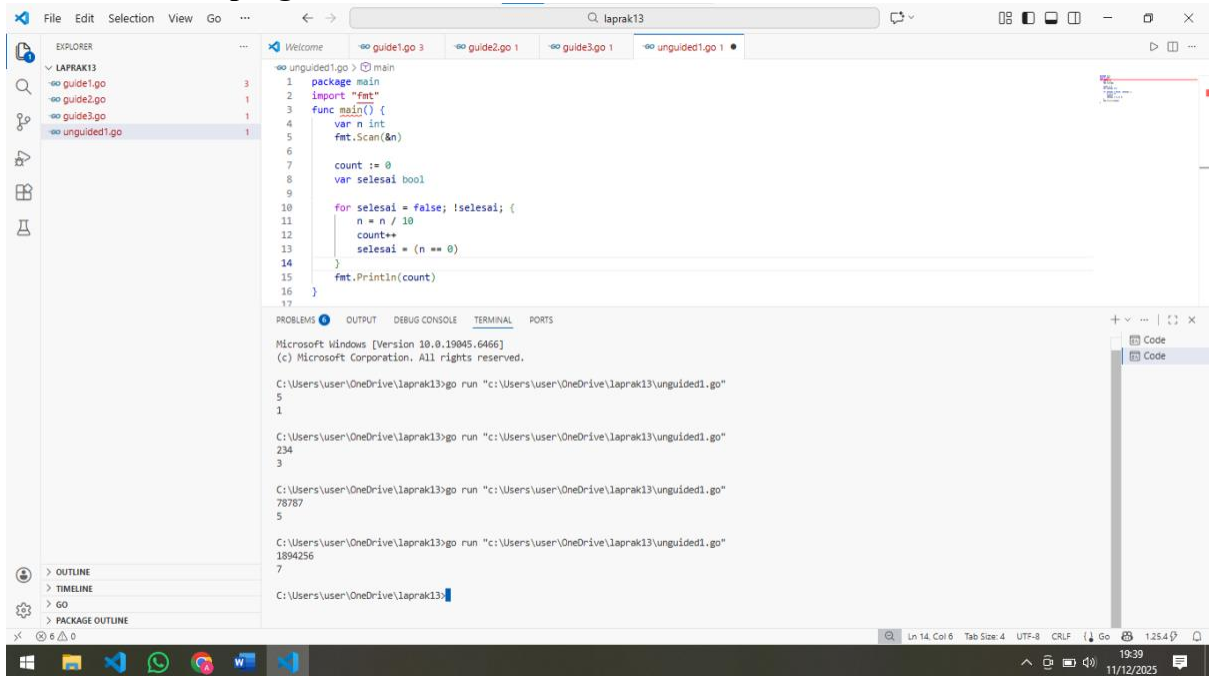
func main() {
    var n int
    fmt.Scan(&n)

    count := 0
    var selesai bool

    for selesai = false; !selesai; {
        n = n / 10
        count++
        selesai = (n == 0)
    }

    fmt.Println(count)
}
```

Screenshoot program



Deskripsi program

1. package main

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

2. import "fmt"

Berfungsi supaya program bisa membaca input dari keyboard menggunakan Scan dan menampilkan output ke layar menggunakan Print atau Println. Tanpa mengimpor fmt, perintah input-output seperti fmt.Scan atau fmt.Println tidak dapat digunakan. Paket fmt merupakan singkatan dari format, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

3. func main()

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layer. Tanpa main(), program tidak akan bisa dijalankan.

4. Tanda buka dan tutup kurung kurawal { }

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main.

5. var n int

Baris ini mendeklarasikan variabel n bertipe integer. Variabel sebagai tempat penyimpanan angka yang dimasukkan. Nilai ini akan diproses untuk dihitung berapa banyak digit yang dimilikinya.

6. fmt.Scan(&n)

fmt.Scan(&n) digunakan untuk membaca input yang dimasukkan dan menyimpannya ke dalam variabel n. Program akan berhenti sementara dan menunggu angka dimasukkan sebuah bilangan bulat.

7. count := 0

Variabel count diinisialisasi dengan nilai 0. Variabel ini sebagai penghitung jumlah digit. Setiap kali angka n dibagi 10, satu digit hilang, sehingga count akan bertambah sebagai banyaknya digit yang telah diproses.

8. var selesai bool

Baris ini mendeklarasikan variabel selesai bertipe boolean. Variabel ini digunakan untuk menentukan kapan perulangan harus dihentikan. Selama selesai bernilai false, program akan terus melakukan proses pembagian dan penghitungan digit.

9. for selesai = false; !selesai; { ... }

Struktur perulangan ini digunakan untuk melakukan proses pembagian angka secara berulang. Perulangan dimulai dengan selesai = false, dan akan terus berjalan selama !selesai bernilai true.

10. n = n / 10

Inti dari proses penghitungan digit terdapat pada baris ini. Pembagian bilangan n dengan 10 akan menghilangkan digit paling belakang dari bilangan tersebut. Operasi ini dilakukan secara berulang hingga nilai n menjadi 0, yang berarti seluruh digit telah diproses.

11. count++

Setiap kali pembagian dilakukan, variabel count ditambahkan satu. Penambahan ini mengambarka bahwa satu digit telah dihitung. Ketika proses selesai, nilai count menunjukkan jumlah digit asli dari bilangan yang dimasukkan.

12. selesai = (n == 0)

Baris ini menentukan apakah perulangan harus dihentikan. Ketika hasil pembagian membuat nilai n menjadi 0, ekspresi (n == 0) bernilai true dan diberikan ke variabel selesai.

13. fmt.Println(count)

Setelah perulangan selesai, program menampilkan nilai count ke layar. Nilai inilah yang merupakan hasil akhir perhitungan dan menunjukkan jumlah digit dari bilangan yang dimasukkan.

2. Tugas 2

Source code

```
package main

import (
    "fmt"
    "math"
)

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

    batas := int(math.Ceil(n))
    awal := int(n*10 + 0.000001)
    target := batas * 10

    hasil := awal
    var selesai bool

    for selesai = false; !selesai; {
        hasil = hasil + 1

        if hasil%10 == 0 {
            fmt.Println(hasil / 10)
        } else {
            fmt.Printf("%.1f\n", float64(hasil)/10.0)
        }
        selesai = hasil >= target
    }
}
```

Screenshoot program

```
package main
import (
    "fmt"
    "math"
)

func main() {
    var n float64
    fmt.Scan(&n)
    batas := int(math.Ceil(n))
    awal := int(n*10 + 0.000001)
    target := batas * 10
    hasil := awal
```

```
Microsoft Windows [Version 10.0.19045.6466]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user\OneDrive\laprak13> go run "C:\Users\user\OneDrive\laprak13\unguide2.go"
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
C:\Users\user\OneDrive\laprak13> go run "C:\Users\user\OneDrive\laprak13\unguide2.go"
2.7
2.8
2.9
3
C:\Users\user\OneDrive\laprak13>
```

Deskripsi program

1. package main

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

2. import "fmt" "math"

Baris import digunakan untuk memasukkan dua paket pendukung, yaitu fmt dan math. Paket fmt menyediakan fungsi input–output yang diperlukan untuk membaca angka desimal dari pengguna dan menampilkan hasil perhitungan ke layar. Sementara itu, paket math digunakan untuk memanfaatkan fungsi matematika seperti math.Ceil, yang berperan penting dalam proses pembulatan angka ke atas.

3. func main()

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layar. Tanpa main(), program tidak akan bisa dijalankan.

4. Tanda buka dan tutup kurung kurawal { }

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main.

5. `var n float64`

Baris ini mendeklarasikan variabel `n` dengan tipe `float64` untuk menyimpan angka desimal yang dimasukkan. Penggunaan tipe `float64` memungkinkan program menerima nilai pecahan dengan tingkat presisi yang cukup baik.

6. `fmt.Scan(&n)`

Ini digunakan untuk membaca input yang dimasukkan dan menyimpannya ke variabel `n`. Tanda `&` pada variabel `n` menunjukkan bahwa nilai yang dimasukkan langsung dituliskan ke dalam lokasi memori variabel tersebut.

7. `batas := int(math.Ceil(n))`

Baris ini menetapkan variabel `batas`, yaitu pembulatan ke atas dari nilai `n`. Dengan `math.Ceil`, program memastikan bahwa `batas` akhir perhitungan selalu merupakan bilangan bulat yang berada di atas atau sama dengan nilai yang dimasukkan. Pembulatan ini menjadi batas atas deret angka yang akan diproses dan ditampilkan.

8. `awal := int(n*10 + 0.000001)`

Variabel `awal` menyimpan nilai `n` yang telah dikalikan 10 dan dikonversi ke bilangan bulat. Penambahan nilai kecil 0.000001 diperlukan untuk menghindari kesalahan pembulatan akibat representasi desimal pada komputer. Representasi ini membuat perhitungan menggunakan satu angka di belakang koma menjadi lebih akurat dan stabil.

9. `target := batas * 10`

Variabel `target` berfungsi sebagai batas perhitungan dalam bentuk satuan persepuluh. Karena program bekerja dengan kelipatan 0.1, batas akhir harus dikonversi ke skala yang sama, yaitu `batas × 10`. Dengan demikian, perulangan akan berhenti saat hasil mencapai nilai `target`.

10. `hasil := awal`

Variabel `hasil` digunakan untuk menyimpan nilai yang akan meningkat setiap iterasi. Pada awalnya, `hasil` diisi dengan nilai `awal` sehingga program mulai menampilkan deret angka tepat pada atau setelah nilai input `n`.

11. var selesai bool

Variabel selesai dideklarasikan sebagai penanda kapan perulangan harus berhenti. Selama selesai bernilai false, program akan terus menambahkan nilai dan menampilkan hasilnya.

12. for selesai = false; !selesai; { ... }

Struktur perulangan ini mengatur agar proses berjalan terus-menerus hingga variabel selesai bernilai true.

13. hasil = hasil + 1

Setiap iterasi perulangan meningkatkan nilai hasil sebanyak satu. Karena nilai hasil dikalikan 10 sejak awal, penambahan 1 berarti menambah angka sebesar 0.1 dalam bentuk desimal. Proses ini membuat deret naik secara bertahap dengan interval 0.1.

14. if hasil % 10 == 0 { ... } else { ... }

Bagian ini digunakan untuk menentukan cara penampilan angka. Jika hasil habis dibagi 10, berarti angka merupakan bilangan bulat, sehingga ditampilkan tanpa angka di belakang koma. Sebaliknya, jika tidak habis dibagi 10, angka ditampilkan dalam format satu angka di belakang koma.

15. fmt.Println(hasil / 10)

Perintah ini menampilkan bilangan bulat ketika hasil merupakan kelipatan 10. Pembagian hasil dengan 10 mengembalikan angka ke nilai desimal sebenarnya, karena sebelumnya angka disimpan dalam skala dikali sepuluh.

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

Jika angka tidak bulat, program menggunakan fmt.Printf dengan format %.1f untuk menampilkan angka satu digit di belakang koma.

17. selesai = hasil >= target

Baris ini mengevaluasi apakah nilai hasil telah mencapai atau melampaui nilai target. Jika kondisi terpenuhi, variabel selesai bernilai true sehingga perulangan berhenti. Program hanya menampilkan angka hingga pembulatan ke atas dari nilai input n.

3. Tugas 3

Source Code

```
package main

import "fmt"

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

    total := 0
    donatur := 0
    var selesai bool

    for selesai = false; !selesai; {
        var donasi int
        fmt.Scan(&donasi)

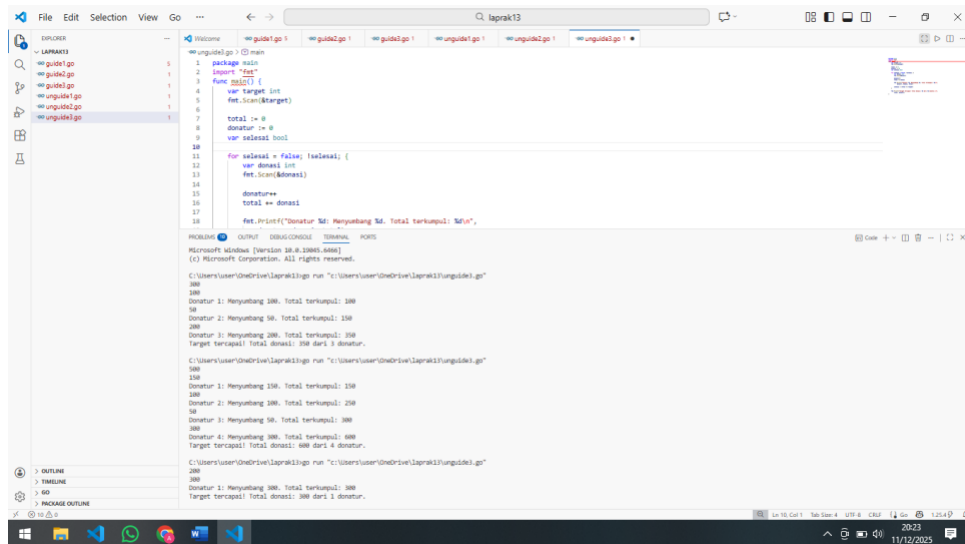
        donatur++
        total += donasi

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

        selesai = (total >= target)
    }

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

Screenshoot program



```
1 package main
2 import "fmt"
3 func main() {
4     var target int
5     fmt.Scan(&target)
6     total := 0
7     donatur := 0
8     var selesai bool
9
10    for selesai = false; !selesai; {
11        var donasi int
12        fmt.Scan(&donasi)
13        donatur++
14        total += donasi
15        fmt.Printf("Donatur %d: Menyumbang %d. Total terkumpul: %d\n", donatur, donasi, total)
16    }
17 }
18
```

OUTPUT

```
C:\Users\User\OneDrive\laprak13> go run *.go
Donatur 1: Menyumbang 100. Total terkumpul: 100
Donatur 2: Menyumbang 50. Total terkumpul: 150
Donatur 3: Menyumbang 200. Total terkumpul: 350
Target tercapai! Total donasi: 350 dari 3 donatur.
C:\Users\User\OneDrive\laprak13> go run *.go
Donatur 1: Menyumbang 150. Total terkumpul: 150
Donatur 2: Menyumbang 200. Total terkumpul: 350
Donatur 3: Menyumbang 50. Total terkumpul: 500
Target tercapai! Total donasi: 400 dari 4 donatur.
C:\Users\User\OneDrive\laprak13> go run *.go
Donatur 1: Menyumbang 100. Total terkumpul: 100
Donatur 2: Menyumbang 200. Total terkumpul: 300
Target tercapai! Total donasi: 300 dari 2 donatur.
```

Deskripsi program

1. package main

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

2. import "fmt"

Berfungsi supaya program bisa membaca input dari keyboard menggunakan Scan dan menampilkan output ke layar menggunakan Print atau Println. Tanpa mengimpor fmt, perintah input-output seperti fmt.Scan atau fmt.Println tidak dapat digunakan. Paket fmt merupakan singkatan dari format, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

3. func main()

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layar. Tanpa main(), program tidak akan bisa dijalankan.

4. Tanda buka dan tutup kurung kurawal { }

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main.

4. var target int dan fmt.Scan(&target)

Baris ini mendeklarasikan variabel target bertipe integer yang berfungsi sebagai batas jumlah donasi yang harus dicapai. Setelah deklarasi, program membaca input yang dimasukkan menggunakan `fmt.Scan(&target)`. Input ini menentukan sampai kapan proses pengumpulan donasi akan berlangsung. Jika total donasi belum mencapai target ini, perulangan akan terus berjalan.

5. total := 0 dan donatur := 0

Program menginisialisasi dua variabel penting, yaitu total dan donatur. Variabel total menyimpan jumlah donasi yang sudah terkumpul. Nilai awalnya 0 karena belum ada donasi masuk. Variabel donatur digunakan untuk menghitung jumlah orang yang telah menyumbang. Nilai awalnya juga 0 dan akan bertambah setiap kali ada input donasi baru.

6. var selesai bool

Variabel selesai dideklarasikan sebagai tipe boolean dan digunakan sebagai penanda apakah proses pengumpulan donasi sudah selesai atau belum. Selama selesai bernilai false, program akan terus meminta input donasi. Variabel ini akan berubah menjadi true ketika total donasi sudah mencapai atau melebihi target.

7. Perulangan for selesai = false; !selesai; { ... }

Perulangan ini menggunakan bentuk pengaturan kondisi secara manual. Pada awalnya selesai diset menjadi false sehingga kondisi !selesai (tidak selesai) bernilai true dan perulangan berjalan. Perulangan ini tidak menggunakan batas angka tertentu, tetapi akan berhenti hanya ketika selesai bernilai true.

8. fmt.Scan(&donasi), total += donasi, dan donatur++

Di dalam perulangan, program membaca nilai donasi dari yang dimasukkan menggunakan `fmt.Scan(&donasi)`.

Setiap kali ada donasi baru:

- donatur bertambah satu untuk menandakan bahwa ada satu orang lagi yang menyumbang.
- total ditambah dengan nilai donasi yang baru masuk, sehingga total donasi tersimpan dengan benar.
- Logika ini membuat program terus memperbarui total donasi sampai mencapai target.

9. Menampilkan informasi setiap donatur

Setelah memperbarui data, program menampilkan informasi dalam format:

Donatur X: Menyumbang Y. Total terkumpul: Z

Bagian ini memberi tahu urutan donatur, jumlah donasi yang diberikan, serta total donasi terbaru. Informasi ini ditampilkan setiap kali input donasi masuk sehingga kita dapat memantau perkembangan donasi secara langsung.

10. selesai = (total >= target)

Baris ini memeriksa apakah total donasi sudah mencapai atau melebihi target. Jika $\text{total} \geq \text{target}$, maka ekspresi tersebut bernilai true dan diberikan ke variabel selesai. Ketika selesai bernilai true, pada iterasi berikutnya kondisi !selesai akan bernilai false dan perulangan otomatis berhenti.

11. Menampilkan pesan akhir setelah target tercapai

Setelah keluar dari perulangan, program menampilkan pesan:

Target tercapai! Total donasi: X dari Y donatur.

Pesan ini menunjukkan bahwa proses pengumpulan donasi berakhir karena target sudah terpenuhi. Total akhir donasi serta jumlah donatur yang berpartisipasi ditampilkan sebagai ringkasan akhir program.