

LAPORAN PRAKTIKUM
Algoritma Pemrograman

MODUL 13
REPEAT-UNTIL



Disusun oleh:

Akhmad Noval Annur

109082500100

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

//tambahkan tangkapan layar dari program (boleh lebih dari 1 jika diperlukan)
CONTOH TANGKAPAN LAYAR:

The screenshot shows a Visual Studio Code editor with a Go workspace. The Explorer panel on the left lists files like `guided1.go`, `guided2.go`, `guided3.go`, `latsol1.go`, `latsol2.go`, `latsol3.go`, `trainingdcd`, and `cihuy.py`. The main editor displays the code for `guided1.go` in `modul13`, which includes a `main` function that reads a word and a repetition count, then prints the word repeatedly. The output window on the right shows the program's execution, displaying the word "kursi" five times. A small Notepad window is also open, containing student information: NAMA : AKHMAD NOVAL ANNUR, NIM : 109082500100, and KELAS : S1IF-13-04.

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var word string
7     var repetitions int
8     fmt.Scan(&word, &repetitions)
9     counter := 0
10    for done := false; !done; {
11        fmt.Println(word)
12        counter++
13        done = (counter >= repetitions)
14    }
15 }
16
```

```
PS C:\SEMESTER 1\modul13> go run "c:\SEMESTER 1\modul13\guided1.go"
kursi
kursi
kursi
kursi
kursi
PS C:\SEMESTER 1\modul13>
```

File Edit Format View Help
NAMA : AKHMAD NOVAL ANNUR
NIM : 109082500100
KELAS : S1IF-13-04

Deskripsi program

Program ini dirancang untuk menampilkan sebuah kata secara berulang sesuai dengan jumlah pengulangan yang ditentukan oleh pengguna. Pada awal eksekusi, pengguna diminta untuk memasukkan sebuah kata dan sebuah bilangan bulat yang menunjukkan berapa kali kata tersebut ingin dicetak. Kedua masukan ini menjadi dasar utama jalannya program.

Setelah menerima input, program menginisialisasi sebuah variabel penghitung yang berfungsi untuk melacak jumlah pengulangan yang sudah dilakukan. Struktur perulangan yang digunakan adalah `repeat-until`, yang diimplementasikan dalam bahasa Go menggunakan perintah `for`. Pada setiap iterasi, program mencetak kata yang telah dimasukkan pengguna dan kemudian menambah nilai penghitung satu per satu.

Perulangan akan terus dijalankan selama jumlah pengulangan yang dilakukan masih lebih kecil dari jumlah yang diminta pengguna. Ketika kondisi tersebut terpenuhi, perulangan akan dihentikan secara otomatis. Dengan mekanisme ini, program memastikan bahwa kata hanya dicetak sebanyak yang diinginkan tanpa kelebihan atau kekurangan.

2. Guided 2

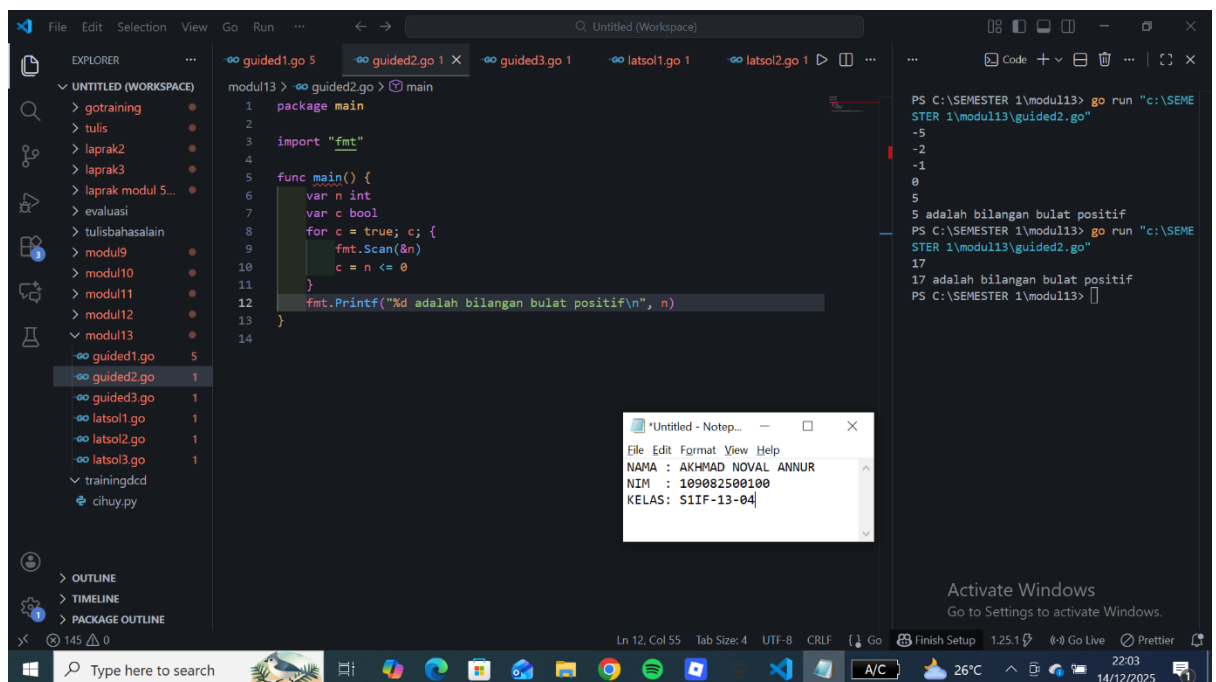
Source Code

```
package main

import "fmt"

func main() {
    var n int
    var c bool
    for c = true; c; {
        fmt.Scan(&n)
        c = n <= 0
    }
    fmt.Printf("%d adalah bilangan bulat positif\n", n)
}
```

Screenshoot program



Deskripsi program

Program ini berfungsi untuk memastikan bahwa masukan yang diberikan oleh pengguna merupakan bilangan bulat positif. Saat program pertama kali dijalankan, pengguna diminta untuk memasukkan sebuah bilangan bulat. Namun, program tidak langsung menerima masukan tersebut apabila nilainya masih kurang dari atau sama dengan nol.

Untuk menangani kondisi tersebut, program menggunakan struktur perulangan repeat-until yang akan terus berjalan selama bilangan yang dimasukkan belum memenuhi syarat sebagai bilangan bulat positif. Setiap kali pengguna memasukkan nilai yang tidak valid, program akan kembali meminta masukan tanpa menampilkan pesan kesalahan tambahan, sehingga fokus utama tetap pada proses validasi input.

Perulangan akan berhenti ketika pengguna akhirnya memasukkan bilangan bulat yang lebih besar dari nol. Setelah kondisi berhenti tercapai, program menampilkan sebuah pesan yang menyatakan bahwa bilangan tersebut adalah bilangan bulat positif. Dengan demikian, program ini dapat digunakan sebagai contoh sederhana proses validasi input menggunakan repeat-until.

3. Guided 3

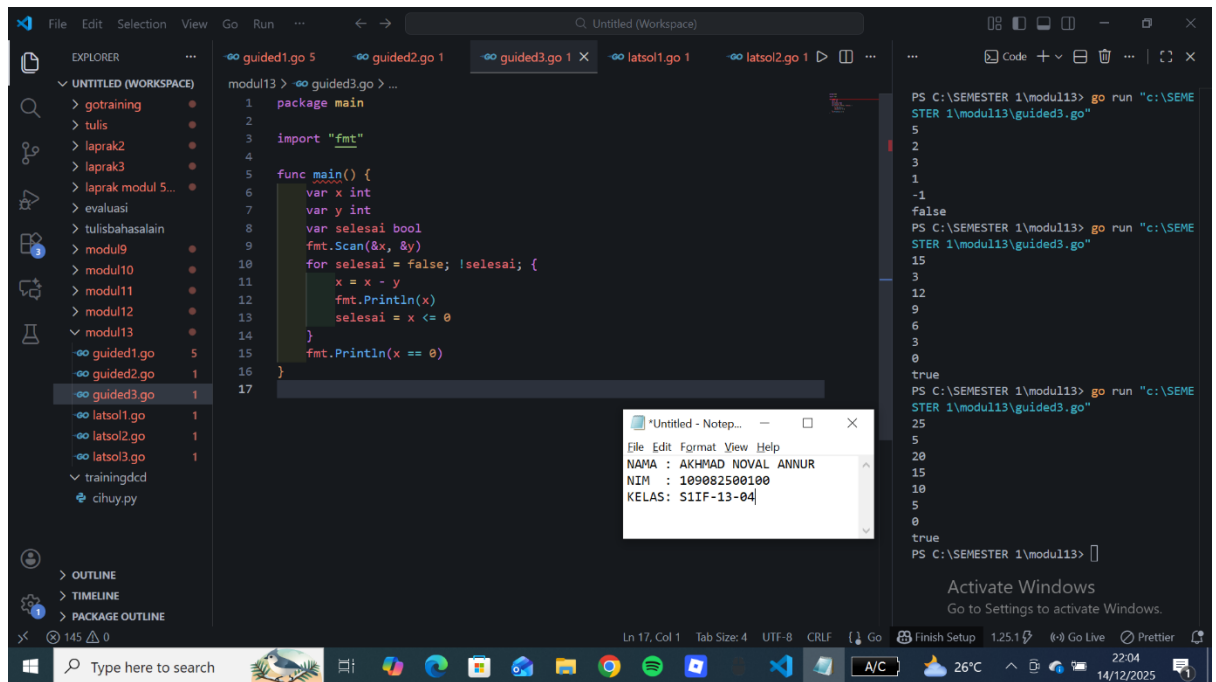
Source Code

```
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 mengecek apakah suatu bilangan merupakan kelipatan dari bilangan lainnya dengan menggunakan metode pengurangan berulang. Pada awal program, pengguna diminta memasukkan dua buah bilangan bulat positif, yaitu bilangan X dan Y. Bilangan X akan dikurangi secara terus-menerus dengan bilangan Y untuk melihat apakah hasil akhirnya dapat mencapai nol.

Proses pengurangan dilakukan di dalam perulangan repeat-until. Setiap hasil pengurangan akan langsung ditampilkan ke layar agar pengguna dapat melihat proses perhitungannya. Perulangan akan berhenti ketika nilai X sudah lebih kecil atau sama dengan nol. Setelah perulangan selesai, program akan menampilkan nilai boolean yang menunjukkan apakah X merupakan kelipatan dari Y, yaitu bernilai true jika hasil akhirnya nol dan false jika tidak.

TUGAS

1. Tugas 1

Source code

```
package main

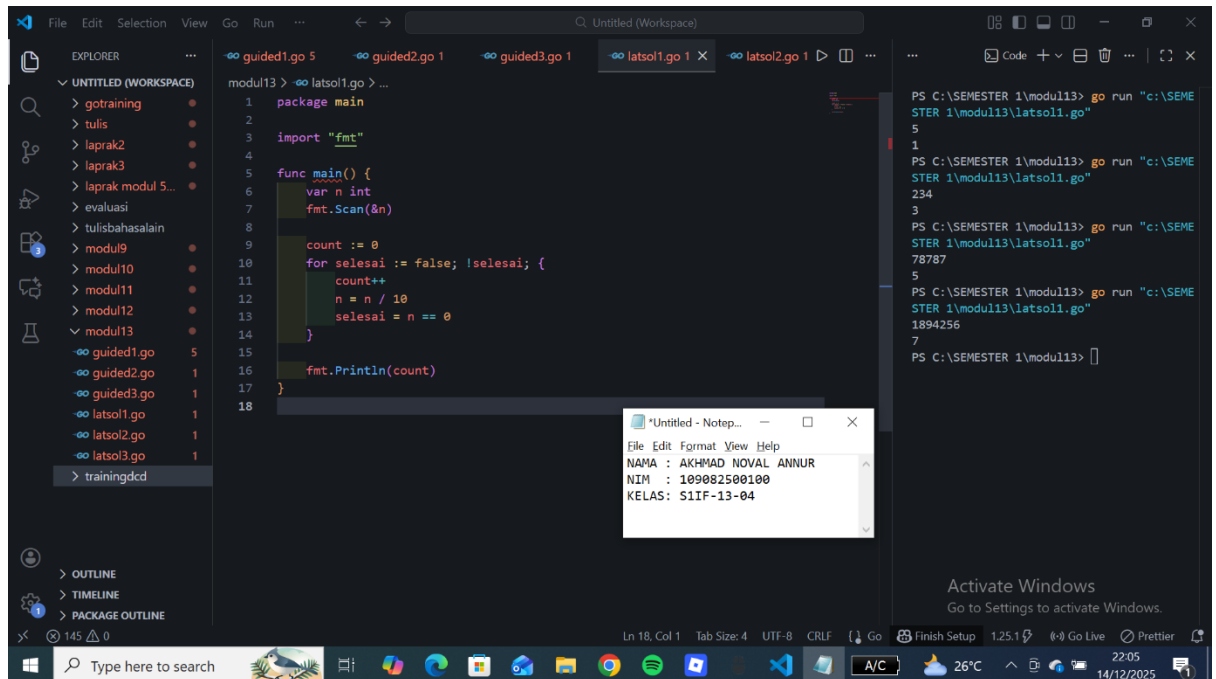
import "fmt"

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

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

    fmt.Println(count)
}
```

Screenshoot program



Deskripsi program

Program ini digunakan untuk menghitung jumlah digit dari sebuah bilangan bulat positif yang dimasukkan oleh pengguna. Pada awal program, pengguna diminta memasukkan satu buah bilangan bulat positif sebagai masukan. Program tidak memeriksa nilai masing-masing digit, melainkan hanya menghitung banyaknya digit yang membentuk bilangan tersebut.

Untuk melakukan perhitungan, program menggunakan operasi pembagian dengan sepuluh secara berulang. Setiap kali bilangan dibagi sepuluh, satu digit terakhir dari bilangan tersebut akan hilang. Bersamaan dengan itu, sebuah variabel penghitung akan ditambah satu untuk mencatat jumlah digit yang telah diproses.

Proses ini dilakukan menggunakan struktur repeat-until hingga nilai bilangan menjadi nol. Ketika kondisi tersebut tercapai, perulangan dihentikan dan program menampilkan hasil akhir berupa jumlah digit dari bilangan yang dimasukkan. Pendekatan ini memungkinkan perhitungan jumlah digit tanpa menggunakan array atau struktur data tambahan.

2. Tugas 2

Source code

```
package main

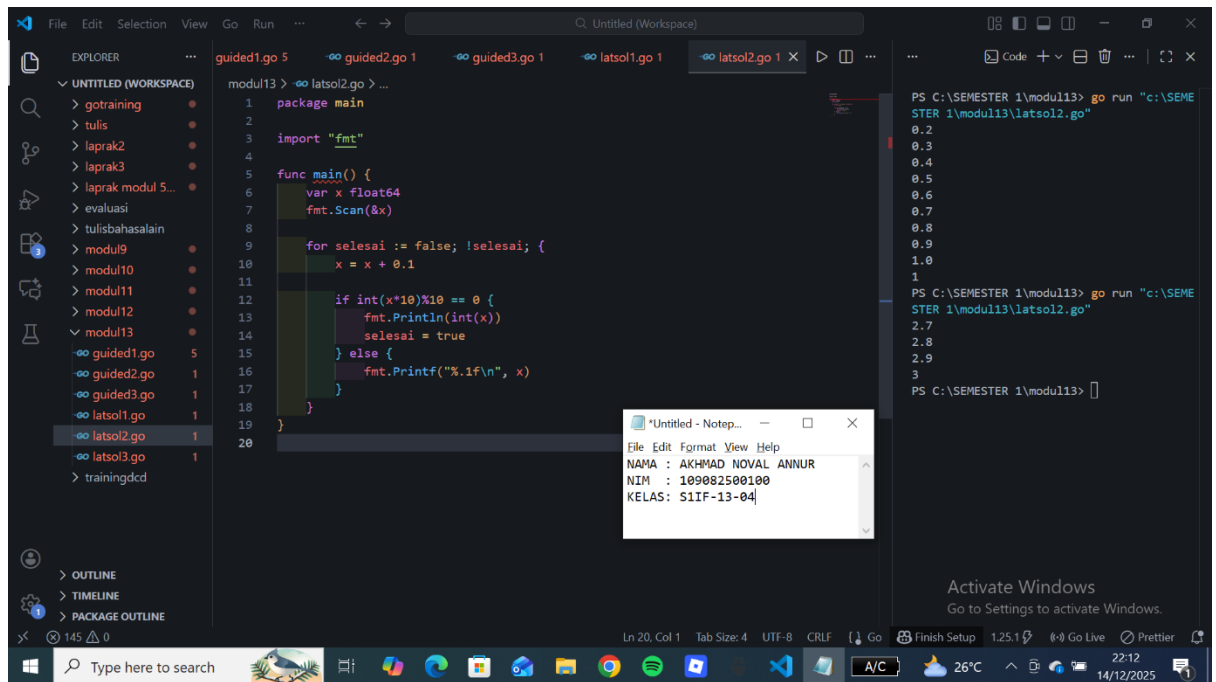
import "fmt"

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

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

    fmt.Println(count)
}
```

Screenshoot program



Deskripsi program

Program ini bertujuan untuk menentukan bilangan bulat optimal dari sebuah bilangan desimal yang dimasukkan oleh pengguna. Bilangan desimal tersebut akan dinaikkan secara bertahap hingga mencapai bilangan bulat terdekat di atasnya. Setiap kenaikan nilai akan ditampilkan agar pengguna dapat melihat proses perubahan nilai secara berurutan.

Agar terhindar dari kesalahan pembulatan bilangan desimal, program menyimpan nilai masukan dalam bentuk bilangan bulat yang merepresentasikan satuan persepuluhan. Dengan cara ini, setiap penambahan satu pada variabel tersebut setara dengan penambahan nilai 0.1 pada bilangan desimal sebenarnya. Pendekatan ini membuat proses perhitungan menjadi lebih stabil dan mudah dikendalikan.

Selama nilai tersebut belum mencapai kelipatan sepuluh, program akan menampilkan hasil dalam bentuk bilangan desimal. Ketika nilai sudah mencapai kelipatan sepuluh, program menampilkan hasil akhir dalam bentuk bilangan bulat tanpa desimal dan menghentikan perulangan. Dengan demikian, program memastikan bahwa keluaran terakhir tidak menampilkan angka seperti “1.0”, melainkan langsung “1”.

3. Tugas 3

Source code

```
package main

import "fmt"

func main() {
    var target int
    var donasi int
    total := 0
    donatur := 0

    fmt.Scan(&target)

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

        donatur++
        total = 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

```
modul13.go:1: package main
modul13.go:2: import "fmt"
modul13.go:5: func main() {
modul13.go:6:     var target int
modul13.go:7:     var donasi int
modul13.go:8:     total := 0
modul13.go:9:     donatur := 0
modul13.go:11:     fmt.Scan(&target)
modul13.go:12:     for selesai := false; !selesai; {
modul13.go:13:         fmt.Scan(&donasi)
modul13.go:14:         donatur++
modul13.go:15:         total = total + donasi
modul13.go:16:         fmt.Printf(
modul13.go:17:             "Donatur %d: Menyumbang %d. Total terkumpul: %d\n",
modul13.go:18:             donatur, donasi, total,
modul13.go:19:         )
modul13.go:20:         if total >= target {
modul13.go:21:             selesai = true
modul13.go:22:         }
modul13.go:23:     }
modul13.go:24:     fmt.Printf(
modul13.go:25:         "Target tercapai! Total donasi: %d dari %d donatur.\n",
modul13.go:26:         total, donatur,
modul13.go:27:     )
modul13.go:28: }
```

```
PS C:\SEMESTER 1\modul13> go run ".\modul13.go"
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:\SEMESTER 1\modul13> go run ".\modul13.go"
500
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:\SEMESTER 1\modul13> go run ".\modul13.go"
300
Donatur 1: Menyumbang 300. Total terkumpul: 300
Target tercapai! Total donasi: 300 dari 1 donatur.
PS C:\SEMESTER 1\modul13> 
```

Deskripsi program

Program ini dibuat untuk mensimulasikan proses pengumpulan donasi hingga mencapai target tertentu. Pada awal program, pengguna diminta memasukkan target donasi yang harus dicapai. Setelah itu, program akan terus menerima masukan berupa jumlah donasi dari setiap donatur secara bertahap.

Setiap kali donasi dimasukkan, program akan menambahkan nilai tersebut ke total donasi dan menghitung jumlah donatur yang telah berpartisipasi. Informasi setiap donatur, termasuk jumlah sumbangan dan total donasi sementara, akan ditampilkan ke layar. Perulangan akan berhenti ketika total donasi telah mencapai atau melebihi target yang ditentukan. Setelah itu, program menampilkan pesan bahwa target telah tercapai beserta total donasi dan jumlah donatur.

