

LAPORAN PRAKTIKUM
Algoritma Pemrograman

MODUL 13

Repeat-Until



Disusun oleh:

Hassan Donny Darmawan

109082500030

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)

    }

}
```

Screenshot program

The screenshot shows a code editor interface for Go (goland). The left sidebar lists files in the 'GOLANG' folder, including 'guided1.go', 'guided2.go', 'guided3.go', 'tugas1.go', 'tugas2.go', 'tugas3.go', 'modul5', 'praktikum', and 'teori'. The main editor area displays the content of 'guided1.go':

```
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)
    }
}
```

The terminal window at the bottom shows the output of running the program:

```
PS C:\Users\HYPE AND\OneDrive\Documents\golang> go run "c:\Users\HYPE AND\OneDrive\Documents\golang\modul 13\guided1.go"
pagi
pagi
pagi
pagi
PS C:\Users\HYPE AND\OneDrive\Documents\golang> go run "c:\Users\HYPE AND\OneDrive\Documents\golang\modul 13\guided1.go"
sore
sore
sore
sore
sore
sore
```

Deskripsi program

Program ini merupakan untuk mencetak kata sebanyak yang diinginkan user. Diawal program user diminta memasukan kata dan angka untuk berapa kali dicetak, yang nantinya akan disimpan pada masing masing variable. Dibaris selanjutnya terdapat counter yang bernilai 0, berfungsi nanti saat di program looping untuk kondisi. Masuk kebagian looping, yang dimana ini sama dengan konsep repeat until dijalankan terlebih dahulu yang ada di dalam looping baru di cek kondisi. Code yang ada di dalam looping ini mencetak kata yang diinputkan user tadi, nilai counter selalu bertambah satu setiap perulangannya dan pengecekan kondisi yaitu **counter>=repetitions**.

Guided 2

Source Code

```
package main

import "fmt"

func main () {

    var a int
```

```

var kondisi bool

for kondisi= false; !kondisi; {

    fmt.Scan(&a)

    kondisi= a>0

}

fmt.Println(a,"adalah bilangan positif")

}

```

Screenshot program

The screenshot shows a Go development environment with the following components:

- File Explorer:** Shows a tree view of files under the "GOLANG" folder, including "assignment", "coba coba", "modul 2", "modul 3", "modul 4", "modul 9", "modul 10", "modul 11", "modul 12", "modul 13" (which contains "guided1.go", "guided2.go", "guided3.go", "tugas1.go", "tugas2.go", "tugas3.go"), "modul5", "praktikum", and "teori".
- Code Editor:** The active file is "guided2.go" with the following content:

```

package main
import "fmt"

func main () {
    var a int
    var kondisi bool

    for kondisi= false; !kondisi; {
        fmt.Scan(&a)
        kondisi= a>0
    }
    fmt.Println(a,"adalah bilangan positif")
}

```
- Terminal:** Shows the command "go run "c:\Users\HYPE AMD\OneDrive\Documents\golang\modul 13\guided2.go"" being run, with the output:

```

Hassan Donny Darmawan
199082500030
S1IF-13-04
5
5 adalah bilangan positif

```
- Status Bar:** Shows "Ln 4, Col 1 46 character Plain t 100% Wind UTF-8".

Deskripsi program

Program ini berfungsi untuk meminta user memasukan angka bilangan positif jika tidak program akan meminta nya terus menerus. Pada awal program terdapat variable a yang dideklarasi dan variable kondisi, variable a bertipe integer sedangkan kondisi bertipe Boolean. Selanjutnya masuk kedalam program looping yang konsepnya menggunakan seperti system repeat until yaitu code yang ada di dalam looping akan dijalankan minimal 1 kali karena kondisi baru di cek di akhir setelah semua code yang

ada di dalam looping dijalankan. Code yang terdapat pada dalam looping ini yaitu meminta user memasukan angka lalu pada baris selanjutnya terdapat pengecekan kondisi apakah angka itu benar bilangan positif atau bukan, jika bukan maka program akan meminta user memasukan angka hingga angka positif.

Diakhir program setelah looping selesai akan mencetak untuk memberi tahu user bahwa angka yg dimasukan terakhir adalah angka positif.

Guided 3

Source Code

```
package main

import "fmt"

func main () {

    var a,b int

    fmt.Scan(&a, &b)

    for kondisi:= false;!kondisi;{

        a-=b

        kondisi = a<=0

        fmt.Println(a)

    }

    fmt.Print(a==0)

}
```

Screenshot program

The screenshot shows the VS Code interface with the following details:

- File Explorer (Left):** Shows a folder structure under "GOLANG". The "modul 13" folder contains files: guided1.go (5), guided2.go (1), guided3.go (1), tugas1.go (1), tugas2.go (1), tugas3.go (1), and teori (1).
- Code Editor (Center):** Displays the content of "guided3.go".

```
package main
import "fmt"

func main () {
    var a,b int
    fmt.Scan(&a,&b)

    for kondisi:= false;!kondisi;{
        a-=b
        kondisi = a<=0
        fmt.Println(a)
    }
    fmt.Println(a==0)
}
```
- Terminal (Bottom):** Shows the command "go run" followed by the path to the file, resulting in the output of the program for various inputs.

```
PS C:\Users\HYPE AMD\OneDrive\Documents\golang> go run "c:\Users\HYPE AMD\OneDrive\Documents\golang\modul 13\guided3.go"
5
2
3
1
-1
false
PS C:\Users\HYPE AMD\OneDrive\Documents\golang> go run "c:\Users\HYPE AMD\OneDrive\Documents\golang\modul 13\guided3.go"
15
3
12
9
6
3
0
true
PS C:\Users\HYPE AMD\OneDrive\Documents\golang>
```

Deskripsi program

Program ini digunakan untuk melakukan pengecekan apakah suatu bilangan merupakan kelipatan dari bilangan lainnya. Diawal program user diminta memasukkan dua angka yang disimpan ke dalam variabel a dan b. Setelah itu program masuk ke proses perulangan. Di dalam perulangan, nilai a akan terus dikurangi dengan nilai b, lalu hasilnya langsung ditampilkan. Proses ini akan terus berjalan selama nilai a masih lebih besar dari nol. Jika nilai a sudah kurang dari atau sama dengan nol, perulangan akan berhenti. Setelah perulangan selesai, program akan mengecek apakah nilai a sama dengan nol atau tidak. Jika hasilnya true, berarti angka pertama merupakan kelipatan dari angka kedua. Jika false, berarti bukan kelipatan.

Alur programnya dimulai dari input dua angka, lalu dilakukan pengurangan berulang, dan diakhiri dengan pengecekan hasil untuk menentukan apakah termasuk kelipatan atau bukan.

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.Println(t)
}
```

Screenshot program

```
modul 13 > < tugas1.go > ...
1 package main
2 import "fmt"
3
4 func main() {
5     var a, t int
6     fmt.Scan(&a)
7
8     for kondisi := false; !kondisi; {
9         t++
10        a /= 10
11        kondisi = a<=0
12    }
13    fmt.Println(t)
14 }
15
```

```
PS C:\Users\HYPE AMD\OneDrive\Documents\golang> go run "c:\Users\HYPE AMD\OneDrive\Documents\golang\modul 13\tugas1.go"
5
1
PS C:\Users\HYPE AMD\OneDrive\Documents\golang> go run "c:\Users\HYPE AMD\OneDrive\Documents\golang\modul 13\tugas1.go"
3
PS C:\Users\HYPE AMD\OneDrive\Documents\golang> go run "c:\Users\HYPE AMD\OneDrive\Documents\golang\modul 13\tugas1.go"
78787
5
PS C:\Users\HYPE AMD\OneDrive\Documents\golang> go run "c:\Users\HYPE AMD\OneDrive\Documents\golang\modul 13\tugas1.go"
1894256
7
```

Deskripsi program

Program ini dibuat untuk menghitung berapa banyak digit dari angka yang dimasukkan oleh pengguna. program mulai dengan deklarasi dua variable yaitu a untuk menampung angka input dan t untuk penghitung jumlah digit. Setelah pengguna memasukkan angka, program masuk ke sebuah looping. Di dalam loop ini, setiap putaran akan menambah nilai t satu kali sebagai tanda bahwa satu digit sudah dihitung. Lalu angka a dibagi 10 supaya digit paling belakang hilang. Looping ini akan terus berjalan sampai angka tersebut habis atau mencapai 0. Saat angka sudah nol atau kurang, perulangannya berhenti dan program mencetak nilai t yang berisi nilai berapa digit yang user masukan

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)
    }
}
```

Screenshot program

The screenshot shows a code editor interface with a terminal window below it. The code editor displays a Go file named `tugas2.go` with the following content:

```
modul 13 > tugas2.go > main
5
6
7 func main() {
8     var a float64
9     fmt.Scan(&a)
10
11     target := math.Ceil(a)
12
13     for {
14         a += 0.1
15         if a >= target {
16             a = target
17             fmt.Printf("%.1f\n", a)
18             break
19         }
20         fmt.Printf("%.1f\n", a)
21     }
22 }
```

The terminal window shows the output of running the program:

```
Hassan Donny Darmawan
199082500030
S11F-13-04
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.0
PS C:\Users\HYPE AMD\OneDrive\Documents\golang> go run "c:\Users\HYPE AMD\OneDrive\Documents\golang\modul 13\tugas2.go"
2.7
2.8
2.9
2.9
PS C:\Users\HYPE AMD\OneDrive\Documents\golang>
```

Deskripsi program

Program ini digunakan untuk membulatkan angka decimal dengan hasil bulat keatas, pada awal program, program membaca angka desimal dari input dan menyimpannya ke variabel `a`. Setelah itu program menentukan angka bulat terdekat di atas nilai tersebut menggunakan `math.Ceil`, lalu menyimpannya sebagai `target`. Nilai `target` ini menjadi tujuan akhir yang harus dicapai oleh program.

Setelah target ditentukan, program masuk ke dalam perulangan yang terus berjalan sampai nilai `a` mencapai `target`. Di setiap putaran, nilai `a` ditambah sedikit, yaitu `0.1`, lalu ditampilkan. Setiap kali nilai `a` berubah, program mengecek apakah nilai tersebut sudah mencapai atau melewati `target`. Jika sudah mencapai maka nilai `a` disamakan dengan `target` agar tepat, ditampilkan, dan perulangan dihentikan.

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")
}
```

Screenshot program

The screenshot shows a Go development environment in a code editor. The left sidebar displays a file tree under the 'GOLANG' folder, including files like 'atas1.go', 'dasar1.go', 'menengah1.go', and several 'modul' sub-folders. The main editor area shows the content of 'tugas3.go':

```
modul 13 > tugas3.go > main
1 package main
2 import "fmt"
3
4 func main (){
5     var target, total, donatur int
6     fmt.Scan(&target)
7     jumlah:=0
8     for kondisi :=true; kondisi{
9         fmt.Scan(&donatur)
10        jumlah++
11        total += donatur
12        fmt.Println("Donatur",jumlah,: menyumbang",donatur,". Total terkumpul:", total)
13        kondisi = total<target
14    }
15    fmt.Println("Target tercapai! Total donasi:", total, "dari",jumlah,"donatur")
16 }
```

The terminal below shows the execution of the program:

```
PS C:\Users\HYPE AMD\OneDrive\Documents\golang> go run "c:\Users\HYPE AMD\OneDrive\Documents\golang\modul 13\tugas3.go"
300
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:\Users\HYPE AMD\OneDrive\Documents\golang>
```

A small modal window is open in the bottom right corner, displaying personal information:

Hassan Donny Darmawan
1099082500030
S1IF-13-04

Deskripsi program

Program ini digunakan untuk menghitung total pemasukan dari donasi serta menghitung butuh berapa banyak donatur yang mendonasikan uangnya untuk mencapai target. Pada awal program, user diminta memasukan nilai target donasi yang akan dikumpulkan, selanjutnya terdapat variable jumlah yang di deklarasi dengan nilai 0.

Dibaris selanjutnya terdapat program for loop yang menggunakan prinsi repeat until, yang dimana program yang ada dalam loop akan dijalankan minimal sekali dan untuk pengecekan kondisi terdapat diakhir. Di dalam looping ini terdapat perintah untuk user memasukan jumlah uang dari donatur, dibaris bawahnya terdapat variable jumlah yang berfungsi menghitung berapa banya donatur dan pengumpulan donasi ini, selain itu juga ada perhitungan total uang yang terkumpul, dan pengecekan kondisi apakah total dari para donatur sudah mencapai target atau belum, jika belum mencapai target maka code yang ada didalam looping akan diulang terus sampai target terpenuhi baru proses tersebut selesai. Diakhir terdapat menampilkan rangkuman dari donasi.