

LAPORAN PRAKTIKUM ALGORITMA

DAN PEMROGRAMAN 1

MODUL 9

IF-THEN



Disusun oleh:

M MAHDAN ARGYA SYARIF

109082500059

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 x int
    fmt.Scan(&x)

    if x < 0 {
        x *= -1
    }
    fmt.Println(x)
}
```

Screenshot program

The screenshot shows a GoLang IDE interface. On the left, the code editor displays the provided Go program. On the right, the terminal window shows the execution of the program and its output.

Code Editor:

```
1 package main
2 import "fmt"
3
4 func main(){ main redeclared in this block (see details)
5     var x int
6     fmt.Scan(&x)
7
8     if x < 0{
9         x *= -1
10    }
11    fmt.Println(x)
12 }
```

Terminal Output:

```
PS C:\GoLang> go run guide1.go
10
10
PS C:\GoLang> go run guide1.go
-3
3
PS C:\GoLang> go run guide1.go
5
5
PS C:\GoLang> go run guide1.go
0
0
PS C:\GoLang> go run guide1.go
-999
999
```

IDE Status Bar:

PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

File Edit View Nama: M Mahdan Argya Syarif
NIM: 109082500059
Kelas: IF-13-07

Ln 3, Col 16 | 61 characters | Plain text | 120% | Window

Deskripsi program

Program ini dirancang untuk memastikan outputnya selalu bilangan positif, tidak peduli apakah inputnya positif atau negatif. Setelah pengguna memasukkan sebuah bilangan integer (x) menggunakan fmt.Scan, program tersebut menjalankan satu algoritma kondisional inti: if n < 0 { x *= -1 }. Logika ini secara spesifik memeriksa apakah angka yang dimasukkan (x) bernilai negatif; jika ya, angka tersebut akan dikalikan dengan -1 untuk mengubahnya menjadi positif. Jika angka yang dimasukkan sudah positif atau nol, kondisi if tidak terpenuhi, dan angka tersebut dicetak apa adanya.

2. Guided 2

Source Code

```
package main

import "fmt"

func main() {
    var x int

    fmt.Scan(&x)

    if x > 0 {
        fmt.Println("positif")
    } else {
        fmt.Println("bukan positif")
    }
}
```

Screenshot program

The screenshot shows a code editor interface with a terminal window below it. The terminal window displays command-line interactions with a Go program. The code editor shows a Go file named 'guide2.go' with syntax highlighting. The terminal output shows the program's behavior for different input values.

```
1 package main
2 import "fmt"
3
4 func main() {    main redeclared in this block
5     var x int
6
7     fmt.Scan(&x)
8
9     if x > 0{
10         fmt.Println("positif")
11     } else {
12         fmt.Println("bukan positif")
13     }
14 }
```

```
PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

● PS C:\GoLang> go run guide2.go
10
positif
● PS C:\GoLang> go run guide2.go
-3
bukan positif
● PS C:\GoLang> go run guide2.go
5
positif
● PS C:\GoLang> go run guide2.go
0
bukan positif
```

Terminal Output:

```
Nama: M Mahdan Argya Syarif
NIM: 109082500059
Kelas: IF-13-07
```

Code Editor Status Bar:

```
Ln 3, Col 16 | 61 characters | Plain text | 120% | Windows
```

Deskripsi program

Program ini dirancang untuk mengklasifikasikan sebuah angka masukan (n) sebagai "positif" atau "bukan positif" menggunakan struktur if-else. Inti algoritmanya terletak pada kondisi $if n > 0$, yang berarti hanya jika nilai yang diinput pengguna tepat lebih besar dari 0, program akan mencetak "positif". Untuk kasus lainnya, termasuk jika pengguna memasukkan 0 atau angka negatif, program akan menjalankan bagian else dan mencetak "bukan positif". Seperti yang Anda catat, 0 secara spesifik tidak dianggap positif dalam logika ini karena kondisi yang digunakan adalah $>$ (lebih besar dari), bukan \geq (lebih besar atau sama dengan).

3. Guided 3

Source Code

```
package main

import "fmt"

func main() {

    var x int

    var t bool
```

```

fmt.Scan(&x)

if x < 0 && x % 2 == 0 {

    t = true

}

fmt.Println(t)
}

```

Screenshot program

The screenshot shows a GoLang IDE interface. On the left, the code editor displays a Go program:

```

1 package main
2 import "fmt"
3
4 func main() {    main redeclared in this block
5     var x int
6     var t bool
7
8     fmt.Scan(&x)
9
10    if x < 0 && x % 2 == 0 {
11        t = true
12    }
13    fmt.Println(t)
14 }

```

A red error highlight is visible on line 4, column 17, indicating a redeclaration of the `main` function. On the right, the terminal window shows the execution of the program:

```

File Edit View Aa Nama: M Mahdan Argya Syarif
Nama: M Mahdan Argya Syarif
NIM: 109082500059
Kelas: IF-13-07
Ln 3, Col 16 61 character Plain t 120% Win

```

The terminal also lists several previous runs of the program with different inputs and outputs.

Deskripsi program

Program ini dirancang khusus untuk bernilai true hanya jika angka yang diinput pengguna adalah bilangan genap negatif. Logikanya menggunakan operator `&&` (DAN), yang mengharuskan kedua kondisi terpenuhi secara bersamaan: pertama, angka tersebut harus negatif ($n < 0$), dan kedua, angka tersebut harus genap ($n \% 2 == 0$). Inilah sebabnya mengapa jika pengguna memasukkan bilangan bulat positif, meskipun angkanya genap, hasilnya akan tetap false; karena syarat pertama (negatif) tidak terpenuhi, sehingga operator `&&` secara otomatis gagal.

TUGAS

1. Tugas 1

Source code

```
package main

import (
    "fmt"
    "math"
)

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

    if orang == orang {
        orang /= 2
        motor := math.Round(orang)
        fmt.Println(motor)
    } else {
        fmt.Println("bukan")
    }
}
```

Screenshot program

The screenshot shows a GoLang development environment. The code editor displays a Go program with a syntax error: "main" is redeclared in the same block. The terminal window shows four runs of the program with inputs 10, 5, 1, and 25, resulting in outputs 10, 5, 1, and 13 respectively. The status bar at the bottom indicates the file is named "soal1_w9.go".

```
1 package main
2 import (
3     "fmt"
4     "math"
5 )
6
7 func main(){    main redeclared in this block
8     var orang float64
9     fmt.Scan(&orang)
10
11    if orang == orang {
12        orang /= 2
13        motor := math.Round(orang)
14        fmt.Println(motor)
15    } else {
16        fmt.Println("bukan")
17    }
18 }
```

PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

- PS C:\GoLang> go run soal1_w9.go
10
5
- PS C:\GoLang> go run soal1_w9.go
1
1
- PS C:\GoLang> go run soal1_w9.go
25
13
- PS C:\GoLang> go run soal1_w9.go
9
5

PS C:\GoLang>

File Edit View AA Nama: M Mahdan Argya Syarif NIM: 109082500059 Kelas: IF-13-07 Ln 3, Col 16 | 61 character Plain t 120% Win

Deskripsi program

Program ini dirancang untuk menghitung jumlah minimum motor yang diperlukan untuk oang yang diinput, di mana 1 motor menampung maksimal 2 orang. Inti algoritmanya adalah mengatasi sisa pembagian: karena 9 orang membutuhkan 5 motor (bukan 4 atau 4.5), program ini menggunakan tipe data float untuk perhitungan ($\text{jumlahOrang} / 2.0$). Setelah mendapatkan hasil desimal (seperti 4.5), program ini kemudian menggunakan fungsi dari *library* math untuk membulatkan hasilnya ke atas (seperti *math.Ceil* atau *math.Round* untuk kasus x.5), memastikan bahwa setiap sisa orang tetap dihitung mendapatkan satu motor penuh.

2. Tugas 2

Source code

```
package main

import "fmt"

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

    if x < 0 && x % 2 == 0 {
        fmt.Println("genap negatif")
    } else {
        fmt.Println("bukan")
    }
}
```

Screenshot program

The screenshot shows a GoLang IDE interface. On the left, the code editor displays the provided Go program. On the right, the terminal window shows the execution of the program and its output.

Code Editor:

```
1 package main
2 import "fmt"
3
4 func main() { main redeclared in this block
5     var x int
6     fmt.Scan(&x)
7
8     if x < 0 && x % 2 == 0 {
9         fmt.Println("genap negatif")
10    } else {
11        fmt.Println("bukan")
12    }
13 }
```

Terminal Output:

```
PS C:\GoLang> go run soal2_w9.go
10
bukan
PS C:\GoLang> go run soal2_w9.go
-4
genap negatif
PS C:\GoLang> go run soal2_w9.go
0
bukan
PS C:\GoLang> go run soal2_w9.go
-2
genap negatif
PS C:\GoLang> []
```

Output Window:

```
Nama: M Mahdan Argya Syarif
NIM: 109082500059
Kelas: IF-13-07
```

Deskripsi program

Program ini pada intinya dirancang untuk memvalidasi apakah sebuah angka masukan (n) adalah bilangan genap negatif. Algoritma utamanya menggunakan satu pernyataan if dengan operator logika $\&\&$ (DAN), yaitu if $n < 0 \&\& n \% 2 == 0$. Ini berarti sebuah angka hanya akan dianggap "benar" atau lolos jika memenuhi kedua syarat sekaligus: pertama, $n < 0$ (angka itu harus negatif), dan kedua, $n \% 2 == 0$ (angka itu harus genap). Seperti contoh Anda, nilai -3 akan gagal karena meskipun memenuhi syarat pertama (negatif), ia tidak memenuhi syarat kedua (ganjil), sehingga operator $\&\&$ akan menghasilkan false.

3. Tugas 3

Source code

```
package main
import "fmt"

func main() {
    var x, y int
    var t bool
    fmt.Scan(&x, &y)

    if y == y {
        t = y % x == 0
        t = bool(t)
    }
    fmt.Println(t)

    if x == x {
        t = x % y == 0
        t = bool(t)
    }
    fmt.Println(t)
}
```

Screenshot program

The screenshot shows a GoLang IDE interface. On the left, the code editor displays a Go program named `soal3_w9.go`. The code defines a package `main`, imports `fmt`, and contains a function `main()` that reads two integers `x` and `y` from standard input, checks if `x` is a factor of `y` and vice versa, and prints the results. A red error bar highlights the line `func main() {` with the message "main redeclared in this block". On the right, the terminal window shows three runs of the program with different inputs:

```
PS C:\GoLang> go run soal3_w9.go
10 5
false
true
PS C:\GoLang> go run soal3_w9.go
3 21
true
false
PS C:\GoLang> go run soal3_w9.go
4 4
true
true
```

Deskripsi program

Program ini adalah alat pengecek faktor sederhana untuk dua angka masukan (`x` dan `y`) yang mengandalkan operator modulo (%) untuk menguji sisanya bagi. Inti logikanya adalah melakukan dua evaluasi boolean yang terpisah: pertama, ia memeriksa (`y % x == 0`) untuk melihat apakah `x` adalah faktor dari `y`, dan kedua, ia memeriksa (`x % y == 0`) untuk melihat apakah `y` adalah faktor dari `x`. Hasil akhir program, seperti "false & true", hanyalah output gabungan dari kedua tes independen ini, yang sepenuhnya ditentukan oleh nilai yang dimasukkan pengguna.