

LAPORAN PRAKTIKUM
Algoritma Pemrograman

MODUL 9
IF-THEN



Disusun oleh:
Johanson Leeroy
109082500017
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 x int

    fmt.Scan(&x)

    if x < 0 {

        x = -x

    }

    fmt.Print(x)

}
```

Screenshoot program

The screenshot shows a Go IDE with the following source code in `Guided1.go`:

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var x int
7     fmt.Scan(&x)
8
9     if x < 0 {
10         x = -x
11     }
12     fmt.Print(x)
13 }
14
15
```

The terminal output shows the execution of the program with the following commands and results:

```
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided1.go"
10
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided1.go"
-3
3
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided1.go"
5
5
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided1.go"
0
0
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided1.go"
-9999
9999
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9>
```

Deskripsi program

1. Program diawali dengan `package main`, `import "fmt"`, dan `func main () {`
2. Mendeklarasikan variabel `x` dengan tipe data integer/int
3. User menginputkan bilangan bulat/integer ke dalam variabel `x`
4. Membuat struktur kontrol if-then dengan kondisi `x < 0`
5. Jika kondisi bernilai true maka program akan melakukan `x = -x` dan Jika kondisi bernilai false maka program akan langsung mengeluarkan output berupa bilangan pada variabel `x` tanpa mengeksekusi program dalam struktur kontrol if-then
6. Output berupa nilai akhir dari variabel `x`

2. Guided 2

Source Code

```
package main

import "fmt"

func main() {

    var x int
```

```

var hasil string

hasil = "Bukan Positif"

fmt.Scan(&x)

if x > 0 {

    hasil = "Positif"

}

fmt.Print(hasil)

}

```

Screenshoot program

The screenshot shows a Go program being edited in a dark-themed IDE. The code defines a `main` function that declares an integer `x` and a string `hasil`. It uses `fmt.Scan(&x)` to read input and an `if` statement to set `hasil` to "Positif" if `x` is greater than 0, otherwise it remains "Bukan Positif". The program then prints `hasil`.

Overlaid on the code is a small window showing the input values: `109082500017`, `S1IF-13-04`, and `Johanson Leeroy`.

Below the editor is a terminal window showing the execution of the program. It shows three runs: the first with input 10 outputs "Positif", the second with input -3 outputs "Bukan Positif", and the third with input 5 outputs "Positif".

Deskripsi program

1. Program diawali dengan `package main`, `import "fmt"`, dan `func main () {`
2. Mendeklarasikan variabel `x` dengan tipe data integer/int
3. Mendeklarasikan variabel `hasil` dengan tipe data string
4. Memberi string "Bukan Positif" pada variabel `hasil`

5. User menginputkan bilangan bulat/integer ke dalam variabel x
6. Membuat struktur kontrol if-then dengan kondisi $x > 0$
7. Jika kondisi bernilai true maka program akan mengubah string pada variabel hasil dari "Bukan Positif" menjadi "Positif" dan jika kondisi bernilai false maka program akan langsung mengeluarkan output berupa string pada variabel hasil tanpa mengeksekusi program dalam struktur kontrol if-then
8. Output berupa string akhir pada variabel hasil

3. Guided 3

Source Code

```
package main

import "fmt"

func main() {
    var x int
    var cek bool

    fmt.Scan(&x)

    if x < 0 {
        cek = x%2 == 0
    }

    fmt.Print(cek)
}
```

Screenshoot program

The screenshot shows a Go IDE with a file named `Guided3.go`. The code defines a `main` function that reads input from the command line, checks if it's a negative number, and prints a boolean result. A separate window shows the input values: `10908250017`, `S1IF-13-04`, and `Johanson Leeroy`. The terminal at the bottom shows the program being run multiple times with different inputs and the corresponding boolean outputs.

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var x int
7     var cek bool
8     fmt.Scan(&x)
9     if x < 0 {
10         cek = x%2 == 0
11     }
12     fmt.Print(cek)
13 }
14
```

Terminal Output:

```
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided3.go"
10
false
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided3.go"
-3
false
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided3.go"
-4
true
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided3.go"
0
false
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Guided3.go"
-2
true
PS D:\.new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9>
```

Deskripsi program

1. Program diawali dengan `package main`, `import "fmt"`, dan `func main () {`
2. Mendeklarasikan variabel `x` dengan tipe data integer/`int`
3. Mendeklarasikan variabel `cek` dengan tipe data boolean
4. User menginputkan bilangan bulat/integer ke dalam variabel `x`
5. Membuat struktur kontrol if-then dengan kondisi `x < 0`
6. Jika kondisi bernilai `true` maka program akan melakukan `cek = x%2 == 0` yang artinya jika $x \bmod 2 = 0$ maka variabel `cek` akan bernilai `true` dan Jika kondisi pada struktur kontrol if-then bernilai `false` maka program akan langsung mengeluarkan output berupa bilangan pada variabel `x` tanpa mengeksekusi program dalam struktur kontrol if-then
7. Output berupa nilai akhir dari variabel `cek`

TUGAS

1. Tugas 1

Source code

```
package main

import "fmt"

func main() {

    var motor, penumpang int

    fmt.Scan(&penumpang)

    if penumpang > 0 {

        motor = (penumpang + 1) / 2

    }

    fmt.Print("Jumlah motor dibutuhkan: ", motor)

}
```

Screenshoot program

The screenshot shows a Go IDE with a source code editor and a terminal window. The source code is as follows:

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var motor, penumpang int
7     fmt.Scan(&penumpang)
8
9     if penumpang > 0 {
10         motor = (penumpang + 1) / 2
11     }
12     fmt.Print("Jumlah motor dibutuhkan: ", motor)
13 }
14
```

The terminal window shows the execution of the program with the following output:

```
PS D:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas1.go"
10
Jumlah motor dibutuhkan: 5
PS D:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas1.go"
1
Jumlah motor dibutuhkan: 1
PS D:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas1.go"
25
Jumlah motor dibutuhkan: 13
PS D:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas1.go"
9
Jumlah motor dibutuhkan: 5
PS D:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9>
```

Deskripsi program

1. Program diawali dengan package main, import "fmt", dan func main () {
2. Mendeklarasikan variabel motor, penumpang dengan tipe data integer/int
3. User menginputkan bilangan bulat/integer ke dalam variabel penumpang
4. Membuat struktur kontrol if-then dengan kondisi penumpang > 0
5. Jika kondisi bernilai true maka program akan melakukan motor = (pengumpang + 1) / 2 dan Jika kondisi bernilai false maka program akan langsung mengeluarkan output berupa bilangan pada variabel motor tanpa mengeksekusi program dalam struktur kontrol if-then
6. Output berupa nilai akhir dari variabel motor yaitu jumlah motor yang dibutuhkan

2. Tugas 2

Source code

```
package main

import "fmt"

func main() {
```



```

var bil int

var cek string

fmt.Scan(&bil)

cek = "bukan"


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

    cek = "genap negatif"

}

fmt.Print(cek)

}

```

Screenshoot program

The screenshot shows a Go IDE with the following source code in `Tugas2.go`:

```

1  import "fmt"
2
3  func main() {
4      var bil int
5      var cek string
6      fmt.Scan(&bil)
7      cek = "bukan"
8
9      if bil < 0 && bil%2 == 0 {
10         cek = "genap negatif"
11     }
12     fmt.Print(cek)
13 }
14
15
16

```

The terminal output shows the program being run three times with different inputs:

```

PS D:\new_journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new_journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas2.go"
bukan
PS D:\new_journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new_journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas2.go"
-4
genap negatif
PS D:\new_journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new_journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas2.go"
0
bukan
PS D:\new_journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new_journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas2.go"
-2
genap negatif
PS D:\new_journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9>

```

Deskripsi program

1. Program diawali dengan package main, import "fmt", dan func main () {
2. Mendeklarasikan variabel bil dengan tipe data integer/int

3. Mendeklarasikan variabel cek dengan tipe data string
4. User menginputkan bilangan bulat/integer ke dalam variabel bil
5. Memberi string "bukan" pada variabel cek
6. Membuat struktur kontrol if-then dengan kondisi $bil < 0 \ \&\& \ bil \% 2 == 0$
7. Jika kondisi bernilai true maka program mengganti string pada variabel cek dari "bukan" menjadi "genap negatif" dan Jika kondisi bernilai false maka program akan langsung mengeluarkan output berupa string pada variabel cek tanpa mengeksekusi program dalam struktur kontrol if-then
8. Output berupa string akhir pada variabel cek

3. Tugas 3

Source code

```
package main

import "fmt"

func main() {
    var x, y int
    var cek bool
    fmt.Scan(&x, &y)
    cek = false
    if y%x == 0 {
        cek = true
    }
    fmt.Println(cek)
    cek = false
    if x%y == 0 {
        cek = true
    }
    fmt.Print(cek)
}
```

Screenshoot program

The screenshot shows a Go program in a text editor and its execution in a terminal. The program, named `Tugas3.go`, is a simple logic checker. It defines a `main` function that takes two integers, `x` and `y`, and a boolean `cek`. It uses `fmt.Scan` to read input from the user. The logic is as follows: if `y` is 0, `cek` becomes true; otherwise, it remains false. Then, if `x` is 0, `cek` becomes true; otherwise, it remains false. Finally, it prints the value of `cek`.

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var x, y int
7     var cek bool
8     fmt.Scan(&x, &y)
9     cek = false
10    if y%x == 0 {
11        cek = true
12    }
13    fmt.Println(cek)
14    cek = false
15    if x%y == 0 {
16        cek = true
17    }
18    fmt.Print(cek)
19 }
20 }
```

The terminal shows the execution of the program with three different inputs:

```
PS D:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas3.go"
10 5
false
true
PS D:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas3.go"
3 21
true
false
PS D:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> go run "d:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9\Tugas3.go"
4 4
true
true
PS D:\new journey\SMTR 1\ALGORITMA DAN PEMROGRAMAN\Minggy 9> 
```

Deskripsi program

1. Program diawali dengan `package main`, `import "fmt"`, dan `func main () {`
2. Mendeklarasikan variabel `x`, `y` dengan tipe data integer/int
3. Mendeklarasikan variabel `cek` dengan tipe data boolean
4. User menginputkan bilangan bulat/integer ke dalam variabel `x` dan `y`
5. Terlebih dahulu memberikan nilai false pada variabel `cek`
6. Membuat struktur kontrol if-then pertama dengan kondisi `y%x == 0`
7. Jika kondisi bernilai true maka program mengubah nilai pada variabel `cek` dari false menjadi true dan Jika kondisi bernilai false maka program akan langsung mengeluarkan output berupa nilai pada variabel `cek` tanpa mengeksekusi program dalam struktur kontrol if-then
8. Mengeluarkan output berupa nilai akhir pada variabel `cek`
9. Kembali memberikan nilai false pada variabel `cek`
10. Membuat struktur kontrol if-then kedua dengan kondisi `x%y==0`
11. Jika kondisi bernilai true maka program mengubah nilai pada variabel `cek` dari false menjadi true dan Jika kondisi bernilai false maka program akan langsung mengeluarkan output berupa nilai pada variabel `cek` tanpa mengeksekusi program dalam struktur kontrol if-then
12. Mengeluarkan output berupa nilai akhir pada variabel `cek`