LAPORAN PRAKTIKUM ALGORITMA

DAN PEMROGRAMAN 1

MODUL 9

IF-THEN

Disusun oleh:

RIZKY TABRIZ DEANOVA

109082500177

S1IF-13-07

Asisten Praktikum

Adithana Dharma Putra

Apri Pandu Wicaksono

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

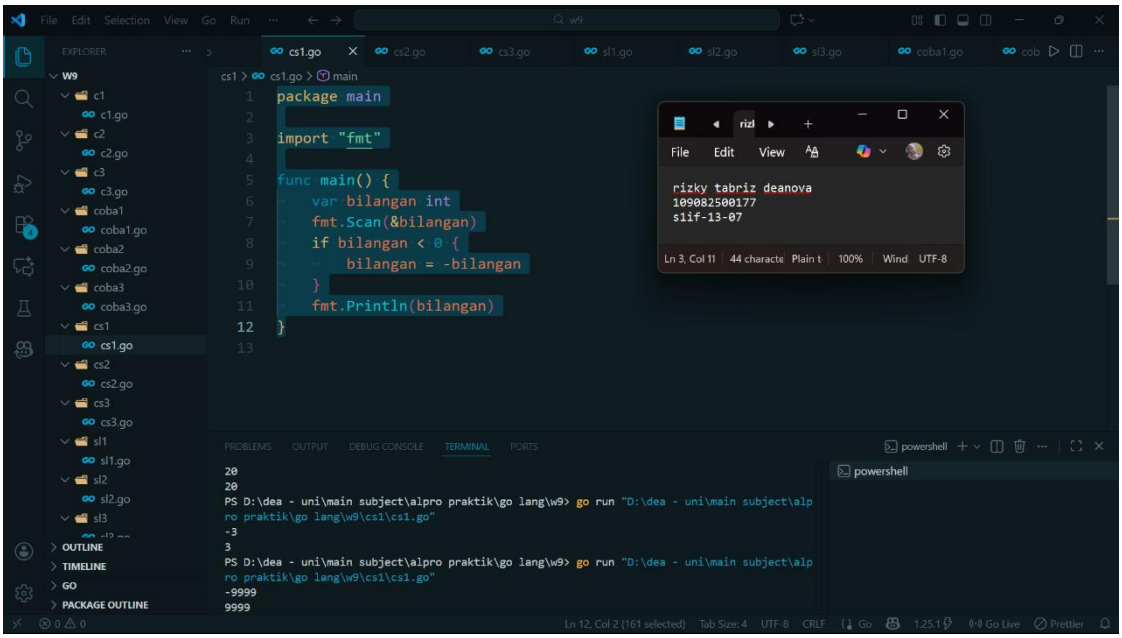
LATIHAN KELAS – GUIDED

1. Guided 1

Source Code

```
package main

import "fmt"

func main() {

    var bilangan int

    fmt.Scan(&bilangan)

    if bilangan < 0 {

        bilangan = -bilangan

    }

    fmt.Println(bilangan)

}
```

Screenshoot program



Deskripsi program

Permulaan program ini dimulai dengan `package main` sebagai penanda bahwa program ini merupakan program *executable* atau program yang dapat dieksekusi (dijalankan).

Dilanjutkan dengan `import "fmt"` yang digunakan untuk memasukkan library `"fmt"` untuk dipakai menjalankan program nantinya.

Lalu, terdapat `func main () {}` yang berperan sebagai penanda tempat program mulai berjalan, seluruh kode ataupun instruksi yang terdapat di dalam kurung `{}` setelah `func main ()` nantinya akan dieksekusi atau dijalankan secara sistematis (berurutan).

Fungsi `var` dalam program adalah sebagai wadah utama untuk penyimpanan data sebelum nanti dibagi lagi sesuai tipe data.
Terdapat `fmt.Println` yang dalam bahasa pemrograman Go digunakan sebagai instruksi untuk mencetak atau menampilkan output di dalam baris baru, dapat dibilang sebagai 'Enter'.
Kemudian `fmt.Scan` yang dalam bahasa pemrograman Go digunakan untuk menjeda hasil output sejenak guna membaca input user, yang kemudian disimpan di dalam variabel.
Lalu `if` berfungsi sebagai pengecekan kondisi, bila mana input memenuhi kondisi yang tertera di dalam `if`, maka akan dilanjutkan ke *command* dalam { }. Jika tidak sesuai, maka perintah lanjutan dari `if` akan dihiraukan atau di*skip*

Yang dilakukan program di dalam gambar di atas adalah **Penghitungan Nilai Mutlak Suatu Bilangan** atau **Membalikkan Tanda** yang mana terlihat melalui kode dari **baris 8 hingga baris 9**.

`if bilangan < 0 {` → Memberikan instruksi untuk melakukan pengecekan kondisi dari variabel bilangan, bila input yang diberikan user dalam variabel bilangan kurang dari 0, maka akan dilanjutkan ke *command* selanjutnya untuk dilakukan operasi hitung yang berguna memberikan nilai mutlak dari suatu bilangan jika input awal bukan merupakan nilai mutlak. Sebaliknya, jika input yang diberikan user dari awal merupakan nilai mutlak dan saat pengecekan kondisi melebihi angka 0, maka *command* dalam `if` akan dihiraukan.

`bilangan = -bilangan` → Instruksi untuk melakukan operasi hitung bila mana input memenuhi kondisi dalam `if`

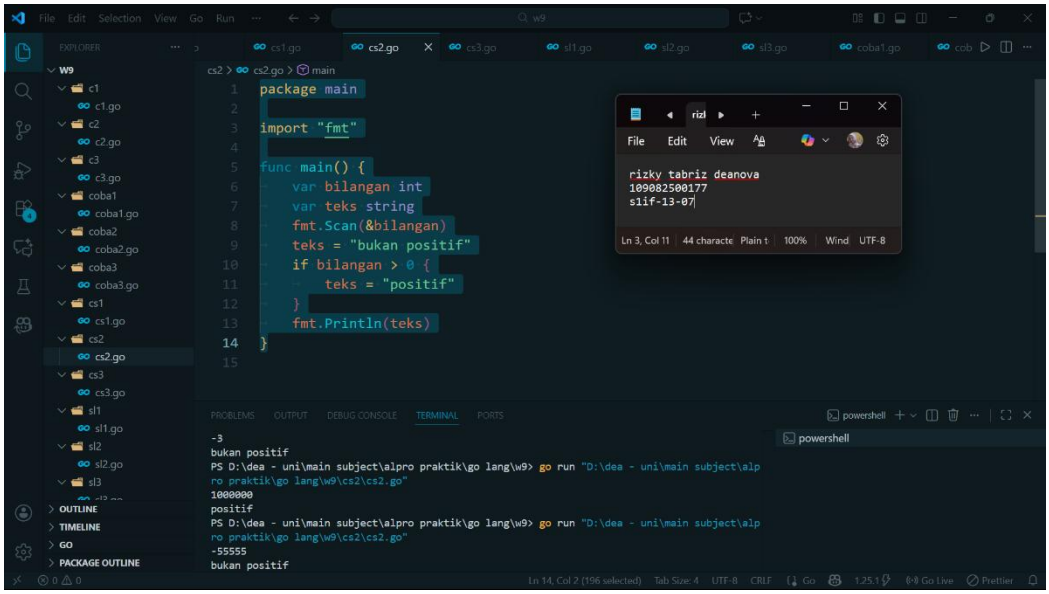
Runtutan Eksekusi:
1. Menunggu user melakukan input data (`fmt.Scan(&bilangan)`) yang nantinya data tersebut akan dimasukkan ke dalam variabel bilangan. Hasil dari kode di baris ini adalah -9999
2. Melakukan pengecekan kondisi (`if bilangan < 0`) untuk variabel bilangan. Jika memenuhi, maka operasi akan dilanjutkan ke *command* dalam `if` dan jika tidak, maka akan dihiraukan. Dalam contoh, karena -9999 masuk dalam kondisi `if`, maka dilanjutkan untuk dilakukan operasi hitung (`-(-9999)`), karena negatif bertemu negatif, maka hasilnya adalah 9999 atau yang disebut nilai mutlak
3. Melakukan print output dari variabel bilangan (`fmt.Println(bilangan)`), yang mana output dari variabel bilangan didapatkan dari pengecekan kondisi di baris sebelumnya. Hasil dari kode di baris ini adalah 9999

2. Guided 2

Source Code

```
package main
import "fmt"
func main() {
    var bilangan int
    var teks string
    fmt.Scan(&bilangan)
    teks = "bukan positif"
    if bilangan > 0 {
        teks = "positif"
    }
    fmt.Println(teks)
}
```

Screenshoot program



Deskripsi program

Permulaan program ini dimulai dengan `package main` sebagai penanda bahwa program ini merupakan program *executable* atau program yang dapat dieksekusi (dijalankan).

Dilanjutkan dengan `import "fmt"` yang digunakan untuk memasukkan library `"fmt"` untuk dipakai menjalankan program nantinya.

Lalu, terdapat `func main () {}` yang berperan sebagai penanda tempat program mulai berjalan, seluruh kode ataupun instruksi yang terdapat di dalam kurung `{}` setelah `func main ()` nantinya akan dieksekusi atau dijalankan secara sistematis (berurutan).
Fungsi `var` dalam program adalah sebagai wadah utama untuk penyimpanan data sebelum nanti dibagi lagi sesuai tipe data.
Terdapat `fmt.Println` yang dalam bahasa pemrograman Go digunakan sebagai instruksi untuk mencetak atau menampilkan output di dalam baris baru, dapat dibilang sebagai 'Enter'.
Kemudian `fmt.Scan` yang dalam bahasa pemrograman Go digunakan untuk menjeda hasil output sejenak guna membaca input user, yang kemudian disimpan di dalam variabel.
Lalu `if` berfungsi sebagai pengecekan kondisi, bila mana input memenuhi kondisi yang tertera di dalam `if`, maka akan dilanjutkan ke *command* dalam { }. Jika tidak sesuai, maka perintah lanjutan dari `if` akan dihiraukan atau di*skip*

Yang dilakukan program di dalam gambar di atas adalah **Pengecekan Positif Negatifnya Suatu Bilangan** yang mana terlihat melalui kode dari **baris 9 hingga baris 10**.

`teks = "bukan positif"` → Dalam contoh ini, dilakukan nilai output dari variabel `teks` terlebih dahulu sebagai 'bukan positif' untuk output teksnya.

`if bilangan > 0 {` → Instruksi untuk melakukan pengecekan dari variabel bilangan yang mana jika memenuhi kondisi (`bilangan > 0`), maka akan lanjut ke *command* dalam `if`

`teks = "positif"` → Instruksi lanjutan dari pengecekan `if` untuk melakukan perubahan output variabel `teks` bila kondisi variabel `bilangan` memenuhi kondisi dalam `if`

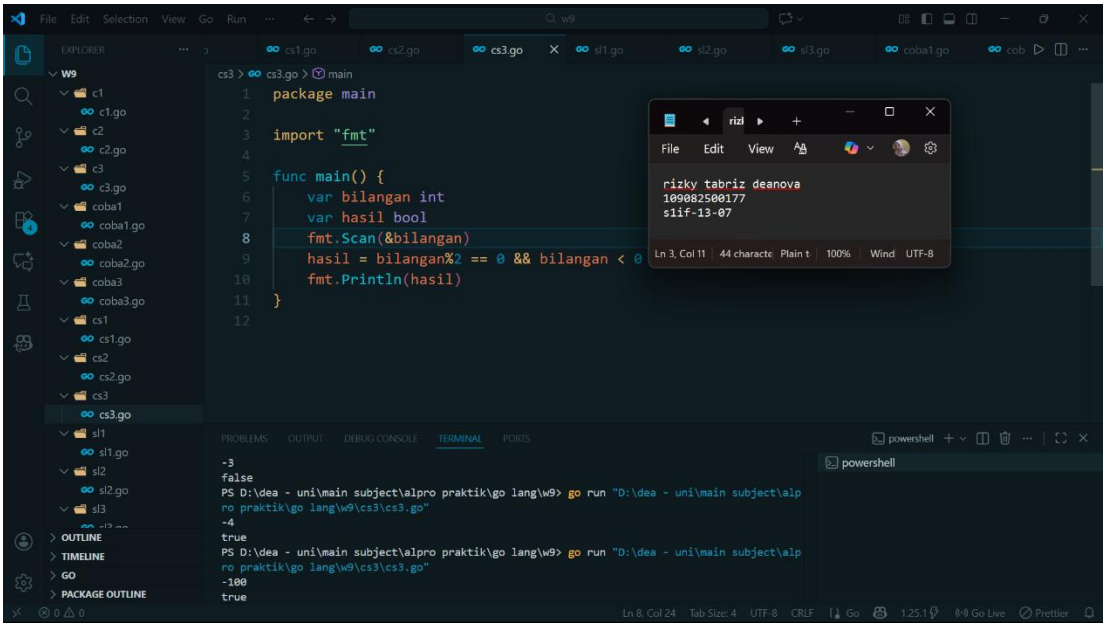
Runtutan Eksekusi:
1. Menunggu user melakukan input data (`fmt.Scan(&bilangan)`) yang nantinya data tersebut akan dimasukkan ke dalam variabel bilangan. Hasil dari kode di baris ini adalah -3
2. Melakukan pemberian output 'bukan positif' terlebih dahulu untuk variabel `teks` (`teks = "bukan positif"`)
3. Melakukan pengecekan kondisi (`if bilangan > 0`) untuk variabel bilangan. Jika memenuhi, maka operasi akan dilanjutkan ke *command* dalam `if` dan jika tidak, maka akan dihiraukan. Dalam contoh, karena -3 tidak masuk dalam kondisi `if`, maka dihiraukan dan langsung diberikan output atau keterangan jenis bilangan
4. Melakukan print output dari variabel teks (`fmt.Println(teks)`), yang mewakili jenis input dalam variabel bilangan termasuk positif atau negatif (bukan positif). Hasil teks diperoleh melalui deklarasi value atau output dari teks dalam baris 9 dan pengecekan dalam baris 10 yang mana dihiraukan karena tidak termasuk kondisi dalam `if`

3. Guided 3

Source Code

```
package main

import "fmt"

func main() {

    var bilangan int

    var hasil bool

    fmt.Scan(&bilangan)

    hasil = bilangan%2 == 0 && bilangan < 0

    fmt.Println(hasil)

}
```

Screenshoot program



Deskripsi program

Permulaan program ini dimulai dengan `package main` sebagai penanda bahwa program ini merupakan program *executable* atau program yang dapat dieksekusi (dijalankan).

Dilanjutkan dengan `import "fmt"` yang digunakan untuk memasukkan library `"fmt"` untuk dipakai menjalankan program nantinya.

Lalu, terdapat `func main () {}` yang berperan sebagai penanda tempat program mulai berjalan, seluruh kode ataupun instruksi yang terdapat di dalam kurung `{}` setelah `func main ()` nantinya akan dieksekusi atau dijalankan secara sistematis (berurutan).

Fungsi `var` dalam program adalah sebagai wadah utama untuk penyimpanan data sebelum nanti dibagi lagi sesuai tipe data.

Terdapat `fmt.Println` yang dalam bahasa pemrograman Go digunakan sebagai instruksi untuk mencetak atau menampilkan output di dalam baris baru, dapat dibilang sebagai 'Enter'.
Kemudian `fmt.Scan` yang dalam bahasa pemrograman Go digunakan untuk menjeda hasil output sejenak guna membaca input user, yang kemudian disimpan di dalam variabel.

Yang dilakukan program di dalam gambar di atas adalah **Menentukan Suatu Bilangan Termasuk Genap Negatif Atau Bukan** yang mana terlihat melalui kode dari **baris 9**.

`hasil = bilangan%2 == 0 && bilangan < 0` → Dalam contoh ini, dilakukan operasi hitung secara langsung untuk mengecek kondisi bilangan. Dilakukan operasi modulus untuk memeriksa apakah input yang diberikan habis dibagi 2 atau terdapat sisa. Kemudian, hasil dari operasi hitung tersebut diperiksa apakah sama dengan kondisi (`0 && bilangan < 0`), di mana jika hasil operasi hitung sesuai dengan kondisi, yaitu menghasilkan 0 dan bilangan tersebut kurang dari 0 (negatif), maka akan diberikan nilai boolean true, jika tidak maka false

Runtutan Eksekusi:

1. Menunggu user melakukan input data (`fmt.Scan(&bilangan)`) yang nantinya data tersebut akan dimasukkan ke dalam variabel bilangan. Hasil dari kode di baris ini adalah -4
2. Melakukan operasi hitung (`hasil = bilangan%2 == 0 && bilangan < 0`). Input bilangan -4 akan dimodulokan dengan angka 2 yang menghasilkan 0 atau tidak ada sisa dari operasi hitung tersebut, lalu diperiksa kembali apakah hasil modulo 0 dan input bilangan adalah kurang dari 0. Karena terpenuhi seluruh kondisi, maka bilangan tersebut diberikan nilai true
3. Melakukan print output dari variabel hasil (`fmt.Println(hasil)`), karena untuk input -4 memenuhi kondisi operasi hasil, maka output yang ditampilkan adalah true, jika dalam kondisi lain dengan input bilangan yang tidak memenuhi operasi hasil, maka output akan menampilkan false. Hasil dari kode di baris ini adalah true

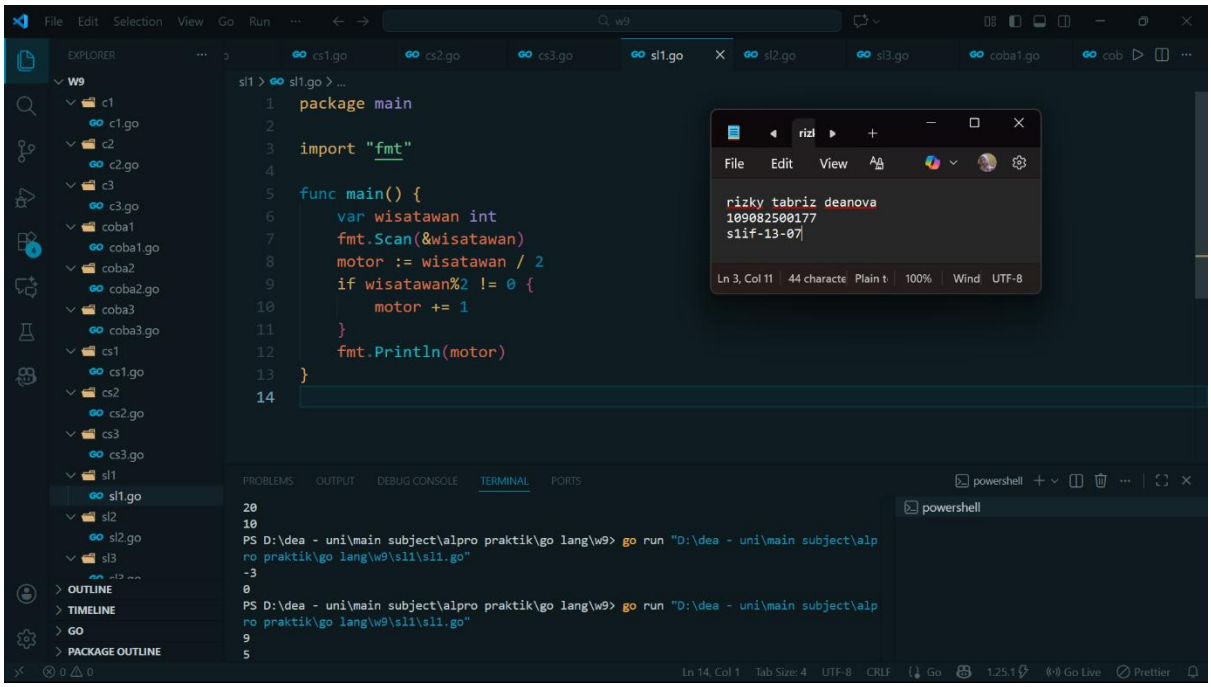
TUGAS

1. Tugas 1

Source code

```
package main

import "fmt"

func main() {

    var wisatawan int

    fmt.Scan(&wisatawan)

    motor := wisatawan / 2

    if wisatawan%2 != 0 {

        motor += 1

    }

    fmt.Println(motor)

}
```

Screenshoot program



Deskripsi program

Permulaan program ini dimulai dengan `package main` sebagai penanda bahwa program ini merupakan program *executable* atau program yang dapat dieksekusi (dijalankan).

Dilanjutkan dengan `import "fmt"` yang digunakan untuk memasukkan library `"fmt"` untuk dipakai menjalankan program nantinya.

Lalu, terdapat `func main () {}` yang berperan sebagai penanda tempat program mulai berjalan, seluruh kode ataupun instruksi yang terdapat di dalam kurung `{}` setelah `func main ()` nantinya akan dieksekusi atau dijalankan secara sistematis (berurutan).
Fungsi `var` dalam program adalah sebagai wadah utama untuk penyimpanan data sebelum nanti dibagi lagi sesuai tipe data.
Terdapat `fmt.Println` yang dalam bahasa pemrograman Go digunakan sebagai instruksi untuk mencetak atau menampilkan output di dalam baris baru, dapat dibilang sebagai 'Enter'.
Kemudian `fmt.Scan` yang dalam bahasa pemrograman Go digunakan untuk menjeda hasil output sejenak guna membaca input user, yang kemudian disimpan di dalam variabel.
Lalu `if` berfungsi sebagai pengecekan kondisi, bila mana input memenuhi kondisi yang tertera di dalam `if`, maka akan dilanjutkan ke *command* dalam { }. Jika tidak sesuai, maka perintah lanjutan dari `if` akan dihiraukan atau di*skip*

Yang dilakukan program di dalam gambar di atas adalah **Pengecekan Jumlah Motor Yang Dibutuhkan** yang mana terlihat melalui kode dari **baris 8 hingga baris 9**.

`motor := wisatawan / 2` → Dilakukan deklarasi variabel baru tanpa perlu menentukan tipe data karena Go Lang akan menentukannya setelah memeriksa operasi apa atau kondisi apa yang dicantumkan oleh user. Dalam hal ini, dilakukan deklarasi operasi hitung untuk variabel wisatawan dibagi dengan angka 2. Kode dalam baris ini adalah sebagai titik awal jumlah motor yang dibutuhkan

`if wisatawan%2 != 0 {` → Instruksi untuk melakukan pengecekan dari variabel wisatawan yang mana jika memenuhi kondisi (`wisatawan%2 != 0`), maka akan lanjut operasi ke *command* dalam `if`. Dilakukan untuk menambahkan jumlah motor sebanyak 1 jika kondisi input wisatawan adalah ganjil, karena dalam bahasa Go Lang, secara umum jika melakukan pembagian tanpa kode spesifik, maka hasil akan dibulatkan ke bawah, jika menghasilkan koma. Berguna untuk antisipasi jumlah wisatawan sebanyak angka ganjil

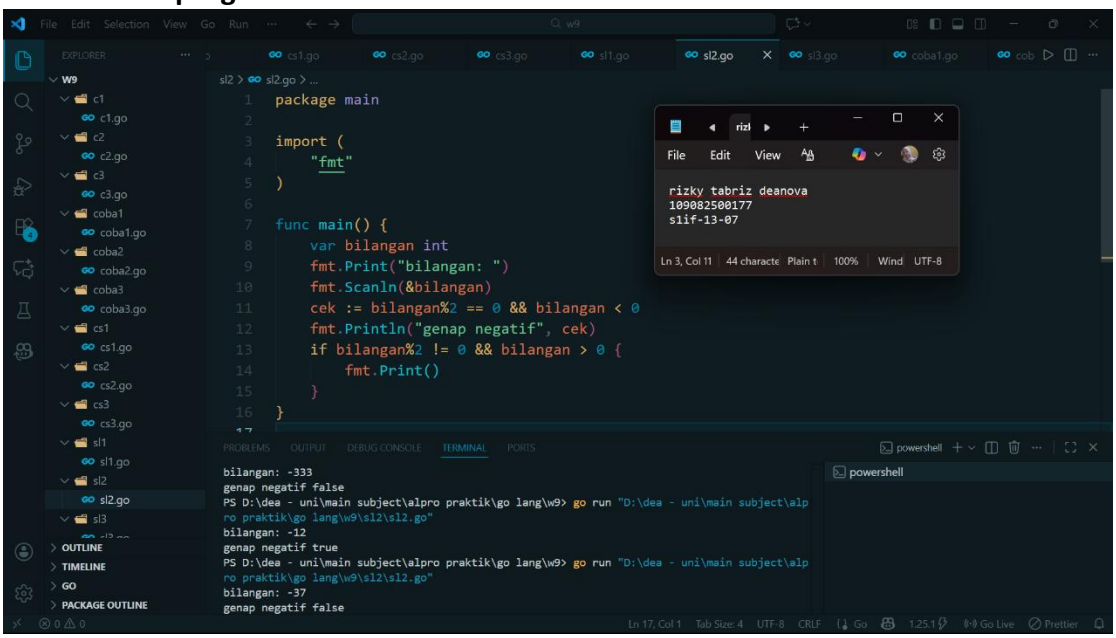
Runtutan Eksekusi:
1. Menunggu user melakukan input data (`fmt.Scan(&wiasatawan)`) yang nantinya data tersebut akan dimasukkan ke dalam variabel wiasatawan. Hasil dari kode di baris ini adalah 9
2. Melakukan deklarasi variabel motor untuk melalukan operasi hitung untuk variabel wisatawan dibagi dengan angka 2. Dalam baris ini, contoh input 9 wisatawan akan dibagi dengan 2
3. Melakukan pengecekan kondisi (`if wisatawan%2 != 0`) untuk variabel wisatawan. Jika memenuhi, maka operasi akan dilanjutkan ke *command* dalam `if` dan jika tidak, maka akan dihiraukan. Dalam contoh, karena 9 masuk dalam kondisi `if`, maka dilanjutkan ke operasi penambahan variabel motor sebanyak 1, nilail 1 ditambahkan ke nilai motor dari operasi jumlah motor di kode sebelumnya, yaitu pada baris 8. Haisl dari kode di baris ini adalah 5

2. Tugas 2

Source code

```
package main
import (
    "fmt"
)
func main() {
    var bilangan int
    fmt.Print("bilangan: ")
    fmt.Scanln(&bilangan)
    cek := bilangan%2 == 0 && bilangan < 0
    fmt.Println("genap negatif", cek)
    if bilangan%2 != 0 && bilangan > 0 {
        fmt.Print()
    }
}
```

Screenshoot program



Deskripsi program

Permulaan program ini dimulai dengan `package main` sebagai penanda bahwa program ini merupakan program *executable* atau program yang dapat dieksekusi (dijalankan).

Dilanjutkan dengan `import "fmt"` yang digunakan untuk memasukkan library `"fmt"` untuk dipakai menjalankan program nantinya.
Lalu, terdapat `func main () {}` yang berperan sebagai penanda tempat program mulai berjalan, seluruh kode ataupun instruksi yang terdapat di dalam kurung `{}` setelah `func main ()` nantinya akan dieksekusi atau dijalankan secara sistematis (berurutan).
Fungsi `var` dalam program adalah sebagai wadah utama untuk penyimpanan data sebelum nanti dibagi lagi sesuai tipe data.
Terdapat `fmt.Println` yang dalam bahasa pemrograman Go digunakan sebagai instruksi untuk mencetak atau menampilkan output di dalam baris baru, dapat dibilang sebagai 'Enter'.
Kemudian `fmt.Scanln` yang dalam bahasa pemrograman Go digunakan untuk menjeda hasil output sejenak guna membaca input user, yang kemudian disimpan di dalam variabel. Dalam perintah ini menggunakan newline.
Lalu `if` berfungsi sebagai pengecekan kondisi, bila mana input memenuhi kondisi yang tertera di dalam `if`, maka akan dilanjutkan ke *command* dalam { }. Jika tidak sesuai, maka perintah lanjutan dari `if` akan dihiraukan atau di*skip*

Yang dilakukan program di dalam gambar di atas adalah **Pengecekan Status Bilangan Benar Genap Negatif Tidak** yang mana terlihat melalui kode dari **baris 11 dan 13**.

`cek := bilangan%2 == 0 && bilangan < 0`→ Dilakukan deklarasi variabel baru tanpa perlu menentukan tipe data karena Go Lang akan menentukannya setelah memeriksa operasi apa atau kondisi apa yang dicantumkan oleh user. Dalam hal ini, dilakukan deklarasi operasi hitung modulo untuk variabel bilangan dibagi dengan angka 2. Kode dalam baris ini adalah sebagai titik awal nilai true atau false untuk status genap negatif suatu bilangan

`if bilangan%2 != 0 && bilangan > 0 {` → Instruksi untuk melakukan pengecekan dari variabel bilangan yang mana jika memenuhi kondisi (`bilangan%2 != 0 && bilangan > 0`), maka akan lanjut operasi ke *command* dalam `if`. Dilakukan untuk **memastikan** status genap negatif tidaknya suatu bilangan. Jika tidak, maka akan dihiraukan. Dalam operasi ini, akan diperiksa apakah nilai bilangan dimodulo 2 tidak sama dengan 0 dan nilai bilangan lebih dari 0

Runtutan Eksekusi:
1. Menunggu user melakukan input data (`fmt.Scan(&bilangan)`) yang nantinya data tersebut akan dimasukkan ke dalam variabel bilangan. Hasil dari kode di baris ini adalah -12
2. Melakukan deklarasi variabel cek untuk melalukan operasi hitung variabel bilangan dimodulo dengan angka 2 yang mana diperintahkan untuk sama dengan 0 (hasil modulo) dan bilangan input merupakan bilangan negatif (< 0). Dalam deklarasi variabel cek ini diberikan kondisi lanjutan yang mana jika termasuk dalam kondisi yang tercantum, maka termasuk ke dalan nilai true dan jika salah diberikan atau tidak tidak sesuai dengan kondisi variabel cek, maka diberikan nilai false. Melalui hal tersebut, Go Lang akan menangkap variabel cek dengan tipe data boolean

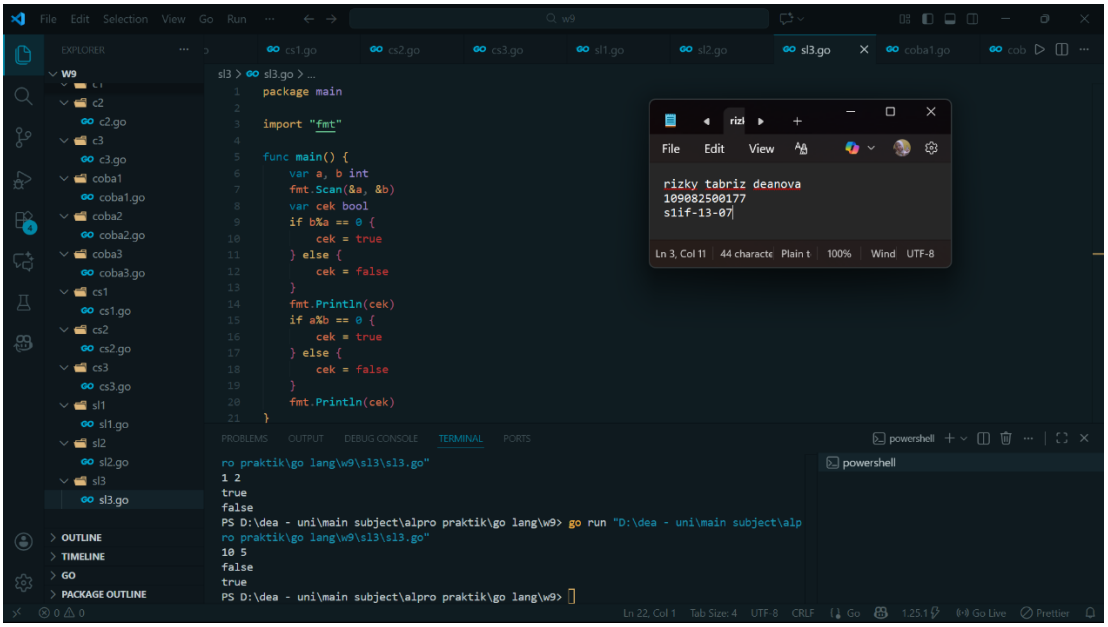
3. Melakukan pengecekan kembali melalui *command* if (`if bilangan%2 != 0 && bilangan > 0`) untuk memastikan nilai genap negatif. Jika nilai input dalam variabel bilangan tidak sama dengan 0 saat dimodulo dengan angka 2 dan nilai bilangan lebih dari 0, maka akan diberikan nilai false
4. Melakukan print output untuk hasil pengecekan (`fmt.Println("genap negatif, cek")`). Hasil dalam kode baris ini adalah genap negatif true

3. Tugas 3

Source code

```
package main
import "fmt"
func main() {
    var a, b int
    fmt.Scan(&a, &b)
    var cek bool
    if b%a == 0 {
        cek = true
    } else {
        cek = false
    }
    fmt.Println(cek)
    if a%b == 0 {
        cek = true
    } else {
        cek = false
    }
    fmt.Println(cek)
}
```

Screenshoot program



Deskripsi program

Permulaan program ini dimulai dengan `package main` sebagai penanda bahwa program ini merupakan program *executable* atau program yang dapat dieksekusi (dijalankan).

Dilanjutkan dengan `import "fmt"` yang digunakan untuk memasukkan library `"fmt"` untuk dipakai menjalankan program nantinya.

Lalu, terdapat `func main () {}` yang berperan sebagai penanda tempat program mulai berjalan, seluruh kode ataupun instruksi yang terdapat di dalam kurung `{}` setelah `func main ()` nantinya akan dieksekusi atau dijalankan secara sistematis (berurutan).

Fungsi `var` dalam program adalah sebagai wadah utama untuk penyimpanan data sebelum nanti dibagi lagi sesuai tipe data.

Terdapat `fmt.Println` yang dalam bahasa pemrograman Go digunakan sebagai instruksi untuk mencetak atau menampilkan output di dalam baris baru, dapat dibilang sebagai 'Enter'.

Kemudian `fmt.Scan` yang dalam bahasa pemrograman Go digunakan untuk menjeda hasil output sejenak guna membaca input user, yang kemudian disimpan di dalam variabel.

Lalu `if` berfungsi sebagai pengecekan kondisi, bila mana input memenuhi kondisi yang tertera di dalam `if`, maka akan dilanjutkan ke *command* dalam { }. Jika tidak sesuai, maka perintah lanjutan dari `if` akan dihiraukan atau di*skip*

Yang dilakukan program di dalam gambar di atas adalah **Pengecekan Keterkaitan Faktor Antara Dua Bilangan** yang mana terlihat melalui kode dari **baris 9 dan baris 15**.

`if b%a == 0 {` → Instruksi untuk melakukan pengecekan pada variabel b agar dimodulokan dengan nilai input variabel a, jika hasil dari modulo adalah 0 atau memenuhi kondisi if, maka akan dilanjutkan ke *command* selanjutnya untuk diberikan nilai true atau false tergantung hasil dari kondisi. Pengecekan if ini dilakukan untuk memeriksa apakah variabel a merupakan faktor dari b

`if a%b == 0 {` → Instruksi untuk melakukan pengecekan pada variabel a agar dimodulokan dengan nilai input variabel b, jika hasil dari modulo adalah 0 atau memenuhi kondisi if, maka akan dilanjutkan ke *command* selanjutnya untuk diberikan nilai true atau false tergantung hasil dari kondisi. Pengecekan if ini dilakukan untuk memeriksa apakah variabel a merupakan faktor dari a

Runtutan Eksekusi:

1. Menunggu user melakukan input data (`fmt.Scan(&a, b)`) yang nantinya data tersebut akan dimasukkan ke dalam variabel a kemudian b. Hasil dari kode di baris ini adalah 10 5
2. Melakukan pengecekan (`if b%a == 0`) pada variabel a agar dimodulokan dengan nilai input variabel b, jika hasil dari modulo adalah 0 atau memenuhi kondisi if, maka akan dilanjutkan ke *command* selanjutnya untuk diberikan nilai true atau false

tergantung hasil dari kondisi. Dalam baris ini, 5 akan dimodulo dengan 10 yang mana tidak menghasilkan nilai 0 karena angka 5 sendiri lebih kecil dari 10 dan tidak mencukupi nilai 10. Karena hasil dari modulo tidak sama dengan 0, maka dilanjutkan ke *command* else yang menyatakan nilai cek dari 5 modulo 10 sebagai 10 merupakan faktor dari 5 adalah false atau salah atau 10 tidak merupakan faktor dari 5

3. Melakukan print output cek dari hasil if pada baris sebelumnya. Hasil dari kode dalam baris ini adalah false
4. Melakukan pengecekan (`if a%b == 0`) pada variabel a agar dimodulokan dengan nilai input variabel b, jika hasil dari modulo adalah 0 atau memenuhi kondisi if, maka akan dilanjutkan ke *command* selanjutnya untuk diberikan nilai true atau false tergantung hasil dari kondisi. Dalam baris ini, 10 akan dimodulo dengan 5 yang mana menghasilkan nilai 0 karena angka 10 sendiri dapat dibagi dengan angka 5 secara merata. Karena hasil dari modulo sama dengan 0, maka dilanjutkan ke *command* lanjutan (then) yang menyatakan nilai cek dari 10 modulo 5 sebagai 5 merupakan faktor dari 10 adalah true atau benar atau 5 merupakan faktor dari 10
5. Melakukan print output cek dari hasil if pada baris sebelumnya. Hasil dari kode dalam baris ini adalah true