

**LAPORAN PRAKTIKUM ALGORITMA  
DAN PEMROGRAMAN 1**

**MODUL 12  
WHILE-LOOP**



**Disusun oleh:**

**Anindya Rahadita Yumnaa**

**109082500138**

**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

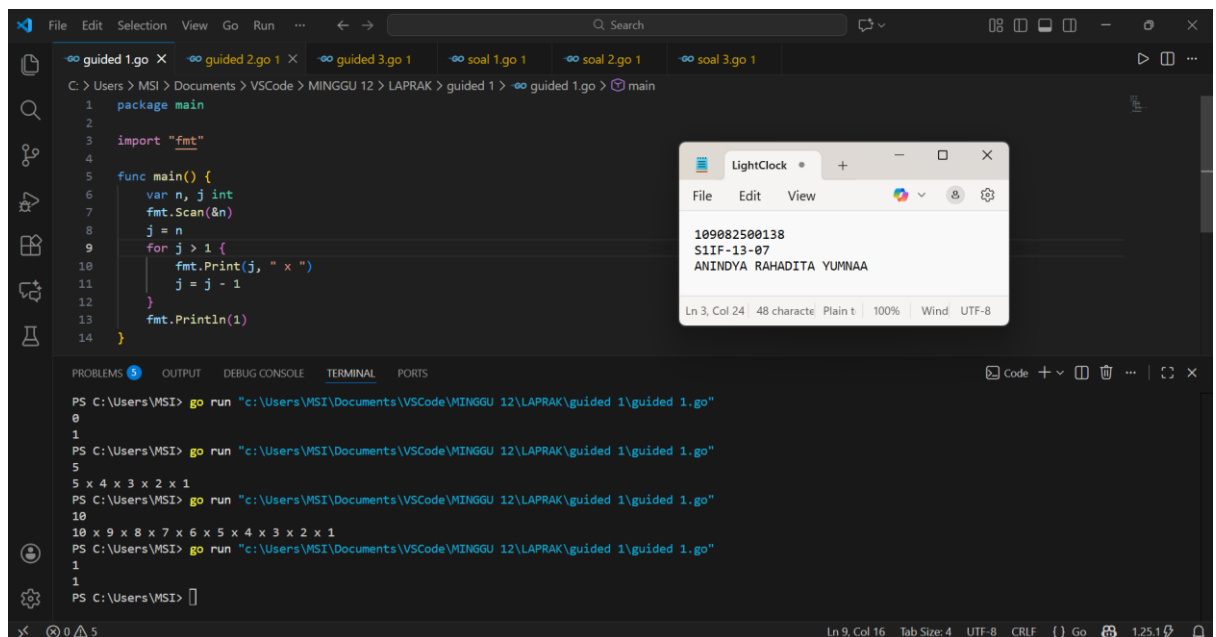
#### Source Code

```
package main

import "fmt"

func main() {
    var n, j int
    fmt.Scan(&n)
    j = n
    for j > 1 {
        fmt.Print(j, " x ")
        j = j - 1
    }
    fmt.Println(1)
}
```

#### Screenshoot program



### Deskripsi program

Program Go ini bertugas untuk menampilkan visualisasi urutan perkalian faktorial dari sebuah bilangan bulat. Program akan meminta untuk memasukkan sebuah angka **bilangan bulat (n)** yang ingin dijabarkan.

Pada penggunaan perulangan menurun (*countdown*), program tidak fokus pada hasil akhir perhitungan, melainkan pada pembentukan format teks perkaliannya.

- Selama angka tersebut (*j*) nilainya **lebih besar dari 1**, program akan mencetak angka tersebut diikuti dengan simbol " x ".
- Setelah dicetak, nilai angka tersebut dikurangi satu persatu pada setiap langkah perulangan.

Setelah kondisi perulangan selesai (saat angka mencapai 1), program akan langsung mencetak angka **1** sebagai penutup rangkaian.

## 2. Guided 2

### Source Code

```
package main

import "fmt"

func main() {

    var token string

    fmt.Scan(&token)
```

```

        for token != "12345abcde" {

            fmt.Scan(&token)

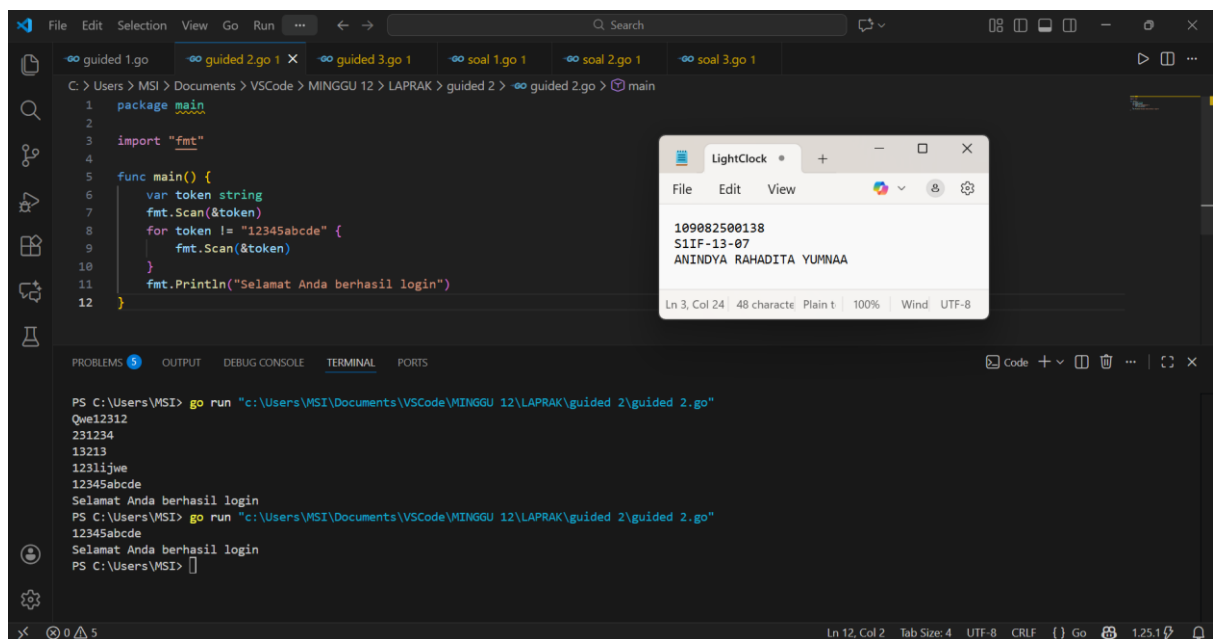
        }

        fmt.Println("Selamat Anda berhasil login")

    }

```

## Screenshoot program



## Deskripsi program

Program ini bertindak sebagai gerbang **verifikasi** sederhana untuk memasukkan **string token yang spesifik dan valid**. Struktur kode menggunakan perulangan yang memeriksa kesesuaian antara input dengan nilai token yang telah didefinisikan dalam program. Selama input yang diberikan berbeda dengan kunci verifikasi, sistem akan terus menahan dalam perulangan dan meminta input ulang secara terus-menerus. Eksekusi program hanya akan berlanjut ke tahap pesan keberhasilan apabila data yang dimasukkan benar-benar identik dengan token yang diharapkan.

### 3. Guided 3

#### Source Code

```

package main

import "fmt"

```

```

func main() {

    var N, s1, s2, j, temp int

    fmt.Scan(&N)

    s1 = 0

    s2 = 1

    j = 0

    for j < N {

        fmt.Print(s1, " ")

        temp = s1 + s2

        s1 = s2

        s2 = temp

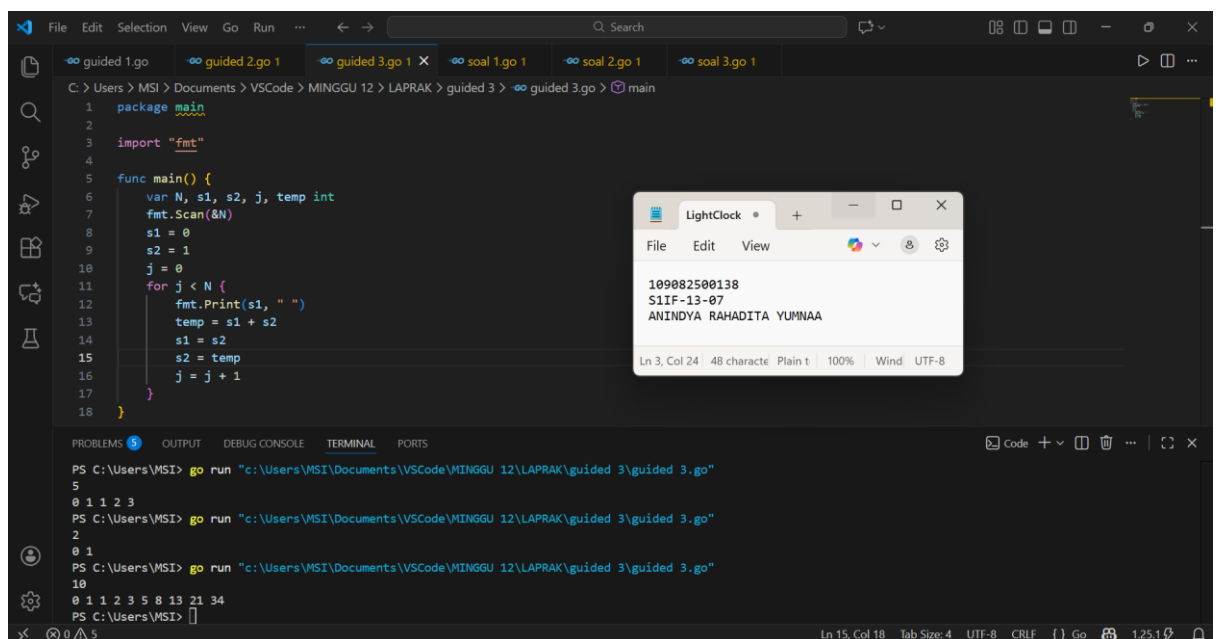
        j = j + 1

    }

}

```

## Screenshoot program



### Deskripsi program

Program Go ini bertugas membangun barisan angka matematika yang dikenal sebagai **deret Fibonacci**. Program akan meminta input berupa **bilangan bulat (N)** yang menentukan seberapa panjang deret angka yang ingin ditampilkan.

Program ini ada pada logika penjumlahan akumulatif. Program memulai start dengan dua angka awal, yaitu **0** dan **1**. Jadi, di balik layar, program menghitung angka berikutnya dengan cara menjumlahkan dua angka yang muncul paling akhir sebelumnya.

➤ Misalnya, setelah mencetak 0 dan 1, program menjumlahkannya menjadi 1. Kemudian 1 ditambah 1 menjadi 2, lalu 1 ditambah 2 menjadi 3, dan seterusnya.

Mekanisme ini berjalan otomatis di dalam sistem dengan cara menggeser nilai variabel penyimpan pada setiap langkah perulangan. Proses generasi angka akan terus berlanjut hingga jumlah bilangan yang tercetak di layar memenuhi kuota (N) yang diminta.

## TUGAS

### 1. Tugas 1

#### Source code

```
package main

import "fmt"

func main() {

    const correctUser = "Admin"

    const correctPass = "Admin"

    var user, pass string

    gagal := 0

    for {

        fmt.Scan(&user, &pass)
```

```

        if user == correctUser && pass == correctPass {

            break

        }

        gagal++

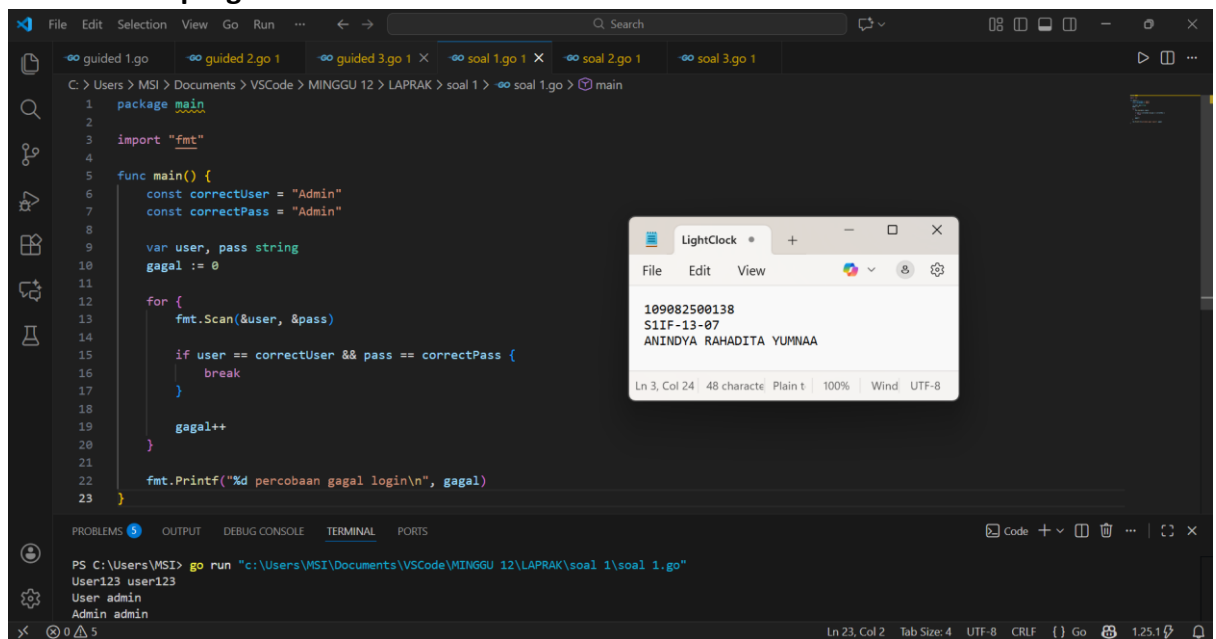
    }

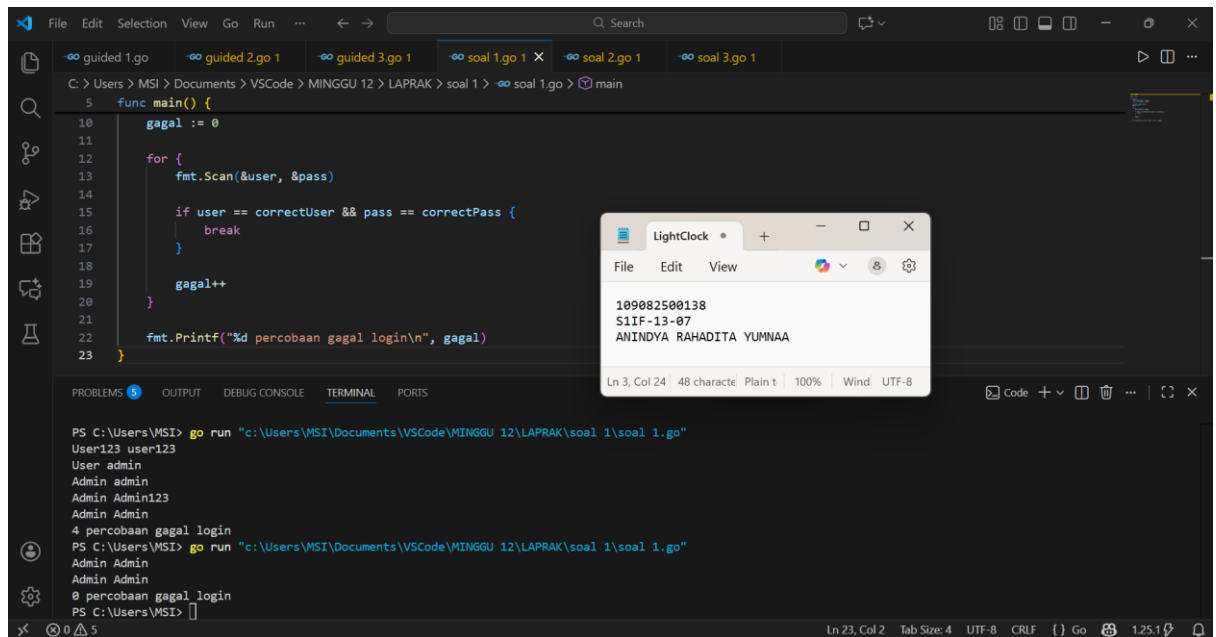
    fmt.Printf("%d percobaan gagal login\n", gagal)

}

```

## Screenshoot program





### Deskripsi program

Program Go ini bertugas mensimulasikan sistem login sederhana yang sekaligus mencatat jumlah kesalahan pengguna. Program akan terus meminta input **username** dan **password** hingga pengguna memasukkan kombinasi yang benar. Program ini ada pada penggunaan perulangan tak terbatas (*infinite loop*) yang dilengkapi penghitung kegagalan. Program membandingkan input dengan konstanta akun yang valid ("Admin").

- Misalnya, jika memasukkan data yang salah, program tidak berhenti, melainkan menambah nilai variabel **gagal** sebanyak satu poin dan meminta input ulang.
- Jika data yang dimasukkan **cocok**, program mengeksekusi perintah break untuk memaksa keluar dari perulangan.

Setelah berhasil melewati validasi (keluar dari loop), program tidak hanya menampilkan pesan sukses, tetapi juga mencetak laporan akhir berupa jumlah **percobaan gagal** yang telah dilakukan sebelum akhirnya berhasil masuk.

## 2. Tugas 2

### Source code

```
package main

import "fmt"

func main() {
```



```

var n int

fmt.Print("Masukkan angka: ")

fmt.Scan(&n)

for n > 0 {

    fmt.Println(n % 10)

    n = n / 10

}

}

```

## Screenshoot program

The screenshot shows a Go program in VS Code. The code is as follows:

```

1 package main
2
3 import "fmt"
4
5 func main() {
6     var n int
7     fmt.Print("Masukkan angka: ")
8     fmt.Scan(&n)
9
10    for n > 0 {
11        fmt.Println(n % 10)
12        n = n / 10
13    }
14 }

```

The terminal output shows three runs of the program with different inputs:

```

PS C:\Users\MSI> go run "c:\Users\MSI\Documents\VSCode\MINGGU 12\LAPRAK\soal 2\soal 2.go"
Masukkan angka: 2
2
4
5
2

PS C:\Users\MSI> go run "c:\Users\MSI\Documents\VSCode\MINGGU 12\LAPRAK\soal 2\soal 2.go"
Masukkan angka: 2544
4
5
2

PS C:\Users\MSI> go run "c:\Users\MSI\Documents\VSCode\MINGGU 12\LAPRAK\soal 2\soal 2.go"
Masukkan angka: 3423554654
4
5
6

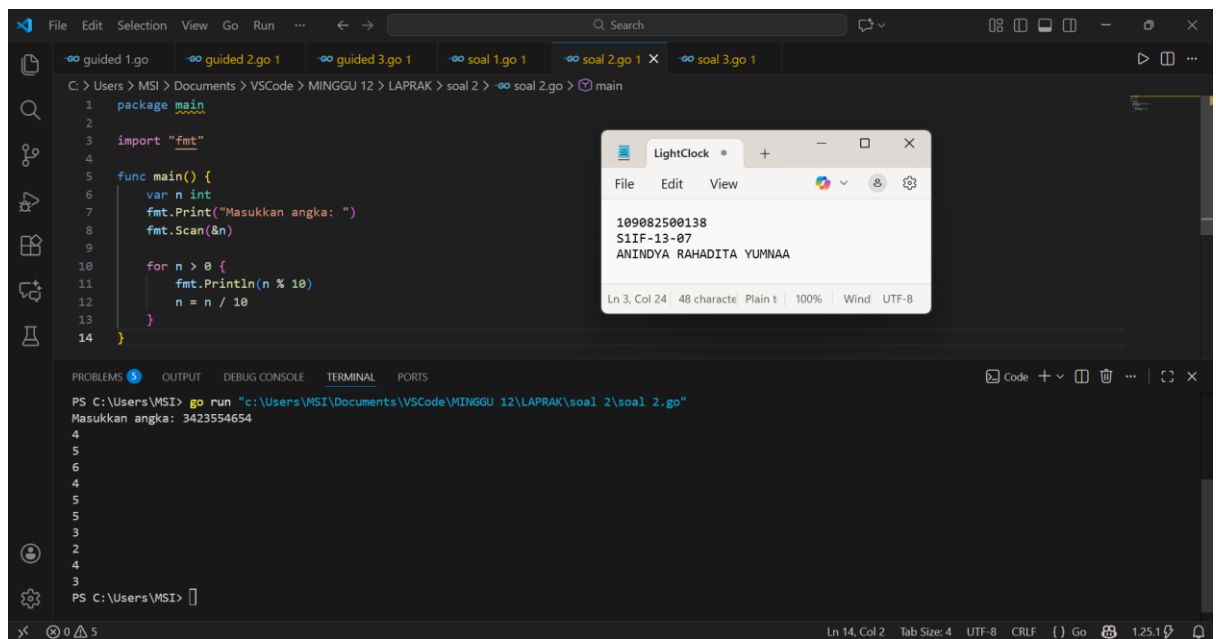
```

A small window titled "LightClock" is also visible, displaying the following information:

```

109082500138
SIIF-13-07
ANINDYA RAHADITA YUMNAA
Ln 3, Col 24 | 48 caracte Plain t | 100% Wind UTF-8

```



### Deskripsi program

Program ini adalah bertujuan untuk memecah sebuah bilangan bulat positif (integer) yang dimasukkan menjadi digit-digit penyusunnya, kemudian menampilkan setiap digit tersebut satu per satu secara berurutan mulai dari posisi **paling belakang (satuan) ke depan**.

1. **Deklarasi & Input:** Program menyiapkan variabel *n* bertipe integer dan meminta pengguna memasukkan sebuah angka.
2. **Pengecekan Kondisi (Looping):** Program memeriksa apakah nilai *n* lebih besar dari 0.
3. **Ekstraksi Digit Terakhir:** Di dalam loop, program mengambil digit paling kanan dari angka tersebut menggunakan operasi modulus (sisanya bagi 10). Hasilnya langsung dicetak ke layar.
4. **Pemotongan Angka:** Setelah digit terakhir dicetak, program membuang digit tersebut dari variabel *n* dengan cara membaginya dengan 10 (pembagian integer).
5. **Pengulangan:** Proses kembali ke langkah 2 dengan nilai *n* yang sudah baru (sudah berkurang digitnya). Proses ini berulang terus menerus hingga nilai *n* menjadi 0.

### 3. Tugas 3

#### Source code

```
package main

import "fmt"

func main() {
    var x, y int
    fmt.Print("Masukkan dua angka: ")
}
```

```

    fmt.Scan(&x, &y)

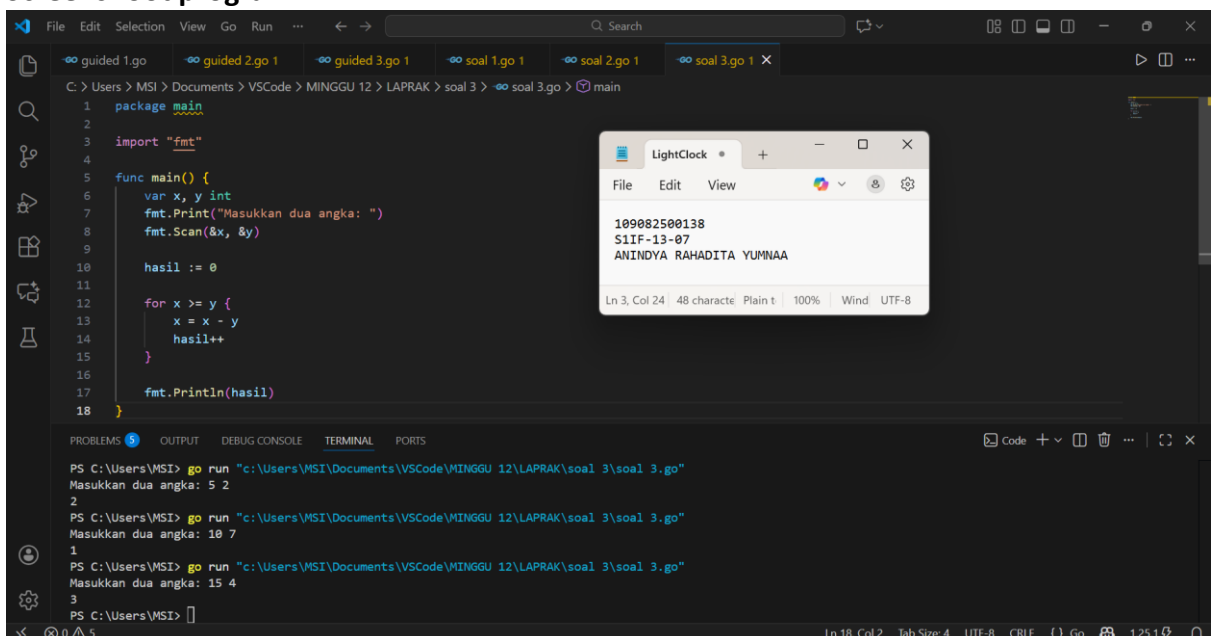
    hasil := 0

    for x >= y {
        x = x - y
        hasil++
    }

    fmt.Println(hasil)
}

```

### Screenshoot program



### Deskripsi program

Program ini adalah implementasi algoritma matematika untuk melakukan **pembagian bilangan bulat (integer division)**. Program ini menggunakan konsep **pengurangan berulang**. Program akan menghitung berapa kali bilangan pembagi (y) dapat dikurangkan dari bilangan yang dibagi (x) sampai sisanya kurang dari y.

1. **Input Data:** Program meminta pengguna memasukkan dua angka:
  - o x: Bilangan yang akan dibagi (dividen).
  - o y: Bilangan pembagi (divisor).
2. **Inisialisasi Penghitung:** Program menyiapkan variabel hasil yang dimulai dari 0. Variabel ini berfungsi sebagai penghitung (counter) berapa kali pengurangan terjadi.

3. **Proses Pengurangan (Looping):**

- Program memeriksa apakah x masih lebih besar atau sama dengan y.
- Jika **ya**, maka x dikurangi dengan y, dan nilai hasil ditambah 1.
- Langkah ini diulang terus menerus.

4. **Penghentian:** Loop berhenti ketika x sudah lebih kecil dari y (nilai x yang tersisa ini sebenarnya adalah sisa bagi atau modulus).

5. **Output:** Program mencetak nilai hasil yang merepresentasikan hasil bagi.