

**LAPORAN PRAKTIKUM ALGORITMA  
DAN PEMROGRAMAN 1**

**MODUL 12**

**DO WHILE**



**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 n, j int

    fmt.Scan(&n)

    j = n

    for j > 1 {

        fmt.Print(j, " x ")

        j = j - 1

    }

    fmt.Println(1)

}
```

#### Screenshoot program

The screenshot shows a Go program being run in a terminal and its output. The program is named `guided1.go` and is located in the `C:\GoLang` directory. The program's logic is as follows:

```
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)
}
```

The terminal output shows the program being run multiple times, with the input `10` and the resulting output `10 x 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1`.

The text editor shows the program's output in a window titled `Nan`. The output is:

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

### Deskripsi program

Program ini bertujuan untuk menghitung serta menjabarkan nilai faktorial dari suatu bilangan bulat, seperti contohnya input 5 yang diterjemahkan menjadi 5 dengan perhitungan  $5 \times 4 \times 3 \times 2 \times 1$ , di mana implementasinya menggunakan bahasa pemrograman Go. Karena bahasa Go meniadakan struktur do-while demi alasan efisiensi desain, algoritma ini sepenuhnya mengandalkan perulangan for yang melibatkan dua variabel bertipe integer, yaitu n sebagai input pengguna dan j sebagai variabel iterasi yang nilainya diinisialisasi sama dengan n. Mekanisme utama di dalam perulangan for tersebut adalah melakukan operasi penurunan nilai (*decrement*) secara berulang dengan kondisi syarat  $j > 1$ , yang mana logika ini sekaligus menjadi solusi praktis untuk kasus faktorial  $0!$  dan  $1!$ , ketika pengguna memasukkan angka 0 atau 1, syarat  $j > 1$  tidak terpenuhi sehingga program tidak akan pernah masuk ke dalam blok perulangan dan langsung mencetak hasil akhir 1, sesuai dengan aturan matematika bahwa  $0!$  dan  $1!$  adalah sama dengan 1.

## 2. Guided 2

### Source Code

```
package main

import "fmt"

func main() {

    var token string

    fmt.Scan(&token)

    for token != "12345abcde" {

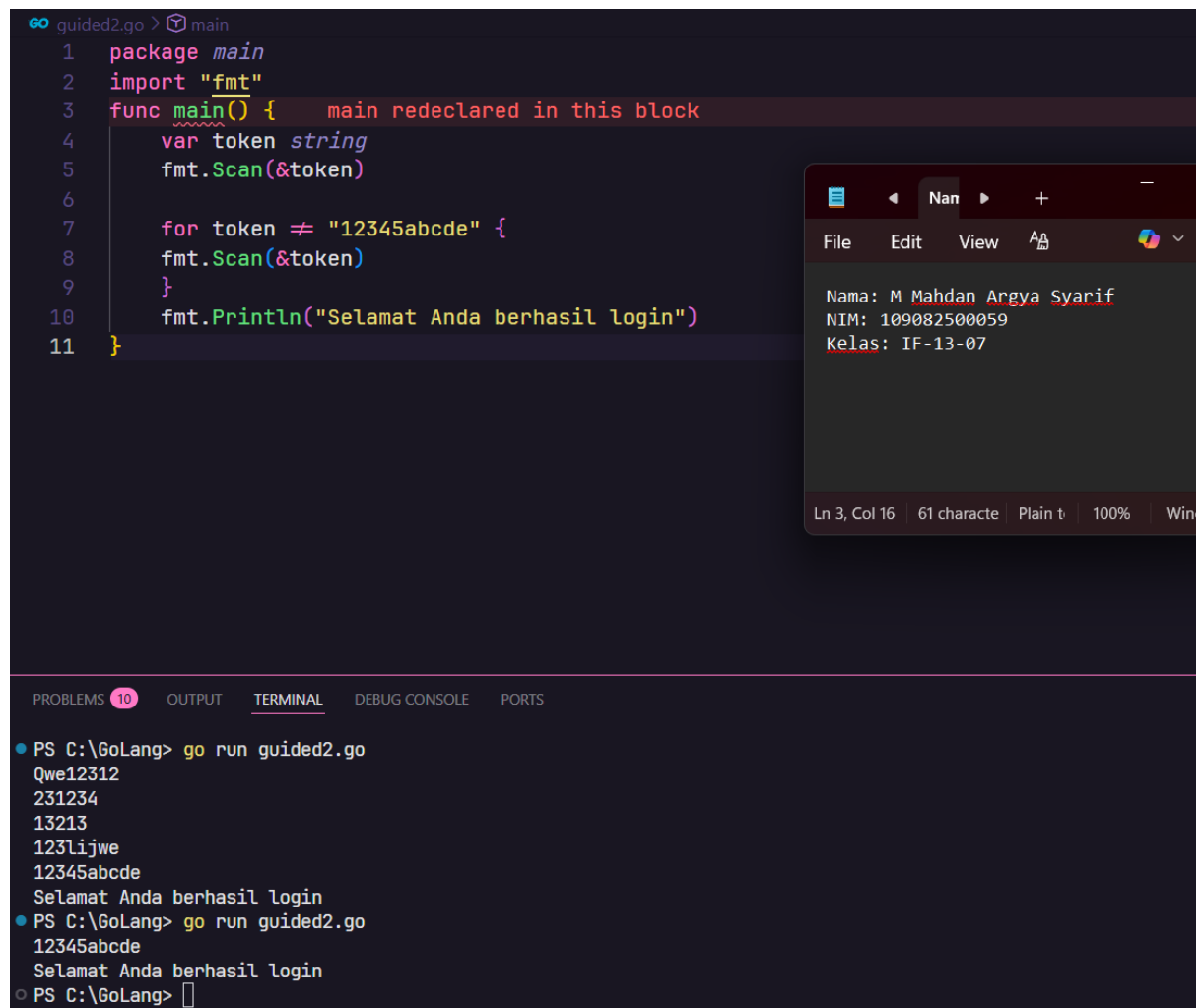
        fmt.Scan(&token)

    }

    fmt.Println("Selamat Anda berhasil login")

}
```

## Screenshoot program



```
guided2.go > main
1 package main
2 import "fmt"
3 func main() {    main redeclared in this block
4     var token string
5     fmt.Scan(&token)
6
7     for token != "12345abcde" {
8         fmt.Scan(&token)
9     }
10    fmt.Println("Selamat Anda berhasil login")
11 }
```

File Edit View A 100% Win

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

Ln 3, Col 16 61 character Plain t 100% Win

PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

```
PS C:\GoLang> go run guided2.go
Qwe12312
231234
13213
1231ijwe
12345abcde
Selamat Anda berhasil login
PS C:\GoLang> go run guided2.go
12345abcde
Selamat Anda berhasil login
PS C:\GoLang>
```

## Deskripsi program

Program ini dirancang untuk mengidentifikasi validitas pengisian token dengan membandingkan input pengguna terhadap kunci yang telah ditentukan sebelumnya, yaitu "12345abcde", yang disimpan dalam sebuah variabel bertipe data *string*. Mekanisme validasi ini menerapkan algoritma perulangan (yang mensimulasikan logika *do-while* menggunakan *for*) dengan kondisi syarat token `!= "12345abcde"`, yang artinya sistem akan terus melakukan perulangan tanpa henti selama input yang diberikan pengguna tidak sama dengan token yang benar. Sebagai konsekuensinya, jika pengguna memasukkan nilai yang salah seperti "231234", "Qwe12312", atau "13213", program akan terus meminta input ulang; proses *looping* ini baru akan berhenti total ketika pengguna memasukkan persis string "12345abcde", yang kemudian akan memicu program untuk menampilkan pesan keluaran akhir "Selamat anda berhasil login".

### 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

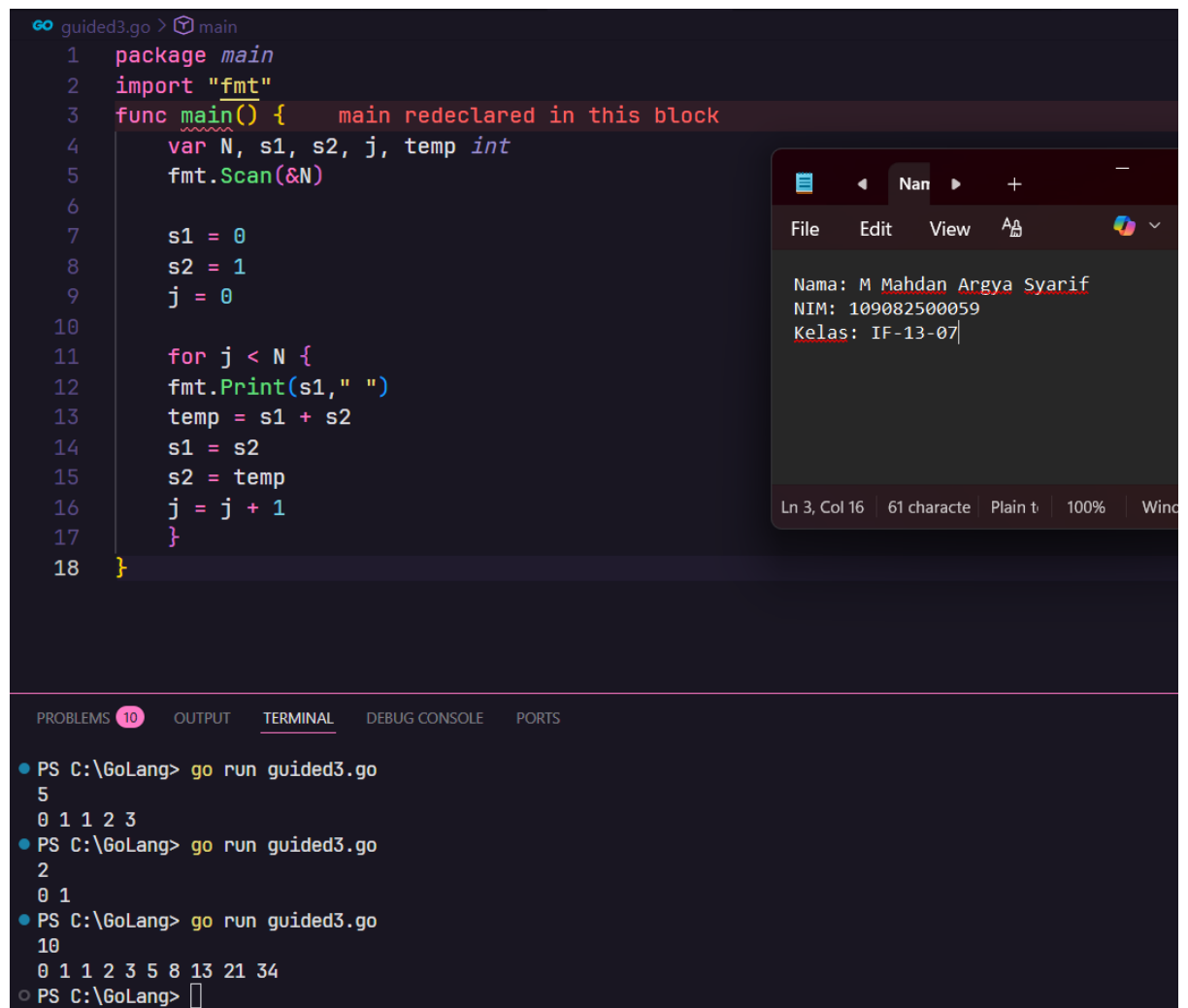
        s2 = temp

        j = j + 1

    }

}
```

## Screenshoot program



```
guided3.go > main
1 package main
2 import "fmt"
3 func main() { main redeclared in this block
4     var N, s1, s2, j, temp int
5     fmt.Scan(&N)
6
7     s1 = 0
8     s2 = 1
9     j = 0
10
11     for j < N {
12         fmt.Print(s1, " ")
13         temp = s1 + s2
14         s1 = s2
15         s2 = temp
16         j = j + 1
17     }
18 }
```

File Edit View

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

Ln 3, Col 16 | 61 character | Plain text | 100% | Window

PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

```
PS C:\GoLang> go run guided3.go
5
0 1 1 2 3
PS C:\GoLang> go run guided3.go
2
0 1
PS C:\GoLang> go run guided3.go
10
0 1 1 2 3 5 8 13 21 34
PS C:\GoLang>
```

## Deskripsi program

Dalam program ini, seluruh variabel (N, s1, s2, j, temp) dideklarasikan menggunakan tipe data *int* (Integer) karena karakteristik deret Fibonacci yang hanya melibatkan operasi bilangan bulat tanpa pecahan. Variabel N berfungsi sebagai parameter input pengguna untuk menentukan panjang deret, sementara j bertindak sebagai *counter* untuk mengontrol kapan perulangan harus berhenti. Mekanisme inti algoritma ini bergantung pada interaksi antara s1 dan s2, di mana variabel temp memegang peran krusial sebagai penyimpanan sementara hasil penjumlahan (s1 + s2); tanpa kehadiran temp, nilai penjumlahan akan tertimpa saat pembaruan variabel dilakukan, sehingga temp menjamin proses pergeseran nilai antar-variabel berjalan akurat untuk menghasilkan urutan deret yang valid.

## TUGAS

### 1. Tugas 1

#### Source code

```
package main

import "fmt"

func main() {

    var gagal int

    var username string

    var password string

    gagal = 0

    for {

        fmt.Print("Masukkan username dan password: ")

        fmt.Scan(&username, &password)

        if username == "Admin" && password == "Admin"{

            break

        } else {

            gagal += 1

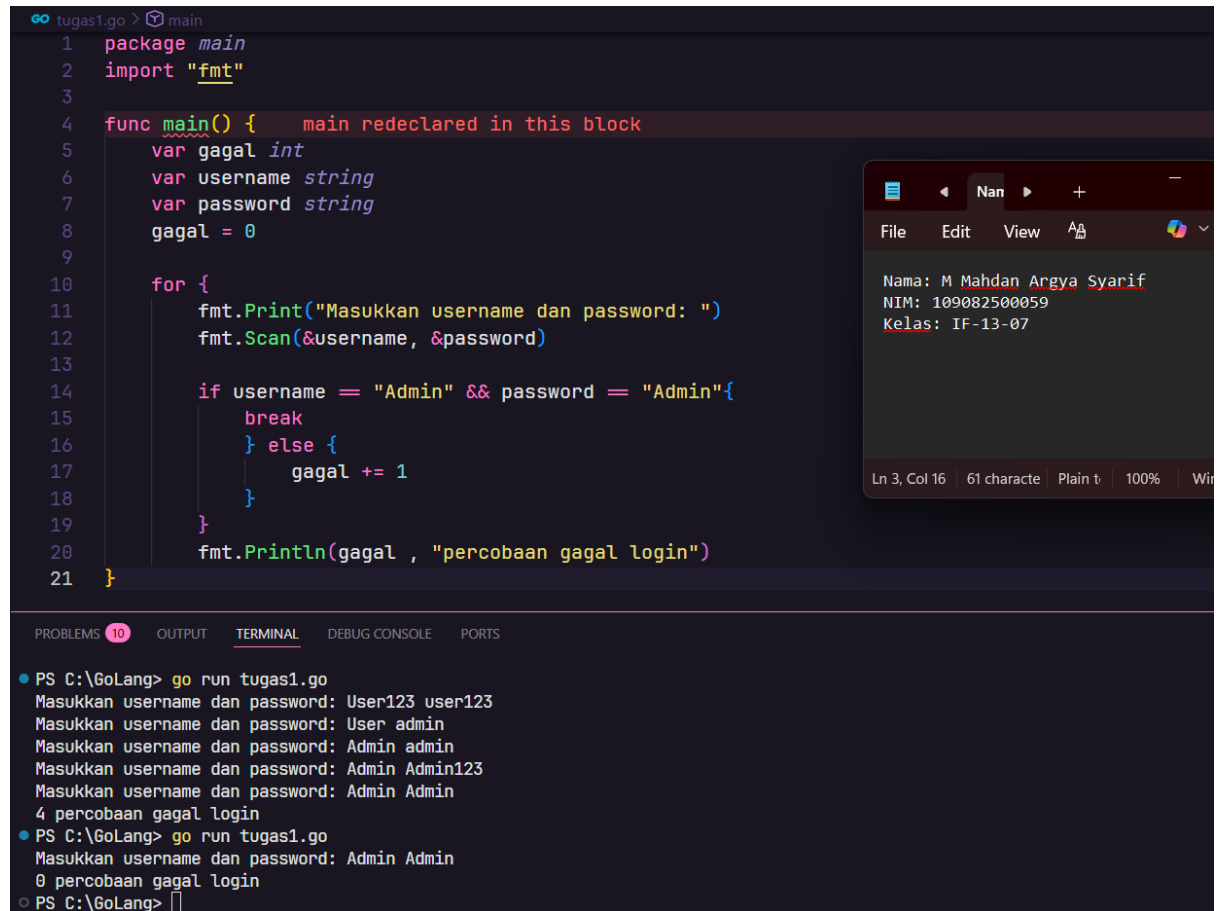
        }

    }

    fmt.Println(gagal , "percobaan gagal login")

}
```

## Screenshoot program



```
1 package main
2 import "fmt"
3
4 func main() { main redeclared in this block
5     var gagal int
6     var username string
7     var password string
8     gagal = 0
9
10    for {
11        fmt.Print("Masukkan username dan password: ")
12        fmt.Scan(&username, &password)
13
14        if username == "Admin" && password == "Admin"{
15            break
16        } else {
17            gagal += 1
18        }
19    }
20    fmt.Println(gagal , "percobaan gagal login")
21 }
```

File Edit View

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

Ln 3, Col 16 61 character Plain t 100% Win

PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

- PS C:\GoLang> go run tugas1.go  
Masukkan username dan password: User123 user123  
Masukkan username dan password: User admin  
Masukkan username dan password: Admin admin  
Masukkan username dan password: Admin Admin123  
Masukkan username dan password: Admin Admin  
4 percobaan gagal login
- PS C:\GoLang> go run tugas1.go  
Masukkan username dan password: Admin Admin  
0 percobaan gagal login
- PS C:\GoLang>

## Deskripsi program

Program ini berfungsi untuk menghitung frekuensi kegagalan login pengguna dengan memvalidasi input terhadap kredensial yang benar, yaitu username "Admin" dan password "Admin". Implementasinya melibatkan tiga variabel utama: gagal bertipe integer yang diinisialisasi dari 0, serta variabel username dan password bertipe string. Algoritma ini berjalan menggunakan struktur perulangan for (sebagai pengganti *do-while* dalam Go) yang dikombinasikan dengan logika kondisional if-else; apabila input pengguna cocok (username == "Admin" && password == "Admin"), program akan mengeksekusi perintah break untuk menghentikan perulangan, namun jika tidak cocok, blok else akan aktif untuk menambahkan nilai variabel gagal sebesar 1. Pada akhirnya, program akan menampilkan output dengan format "[jumlah gagal] Percobaan gagal login", yang berarti jika pengguna langsung berhasil login maka akan muncul "0 Percobaan gagal login", sedangkan jika terjadi 4 kali kesalahan sebelumnya, outputnya akan menjadi "4 percobaan gagal login".



## 2. Tugas 2

### Source code

```
package main

import "fmt"

func main() {

    var n, urut int

    fmt.Print("Masukkan angka: ")

    fmt.Scan(&n)

    for n > 0 {

        urut = n % 10

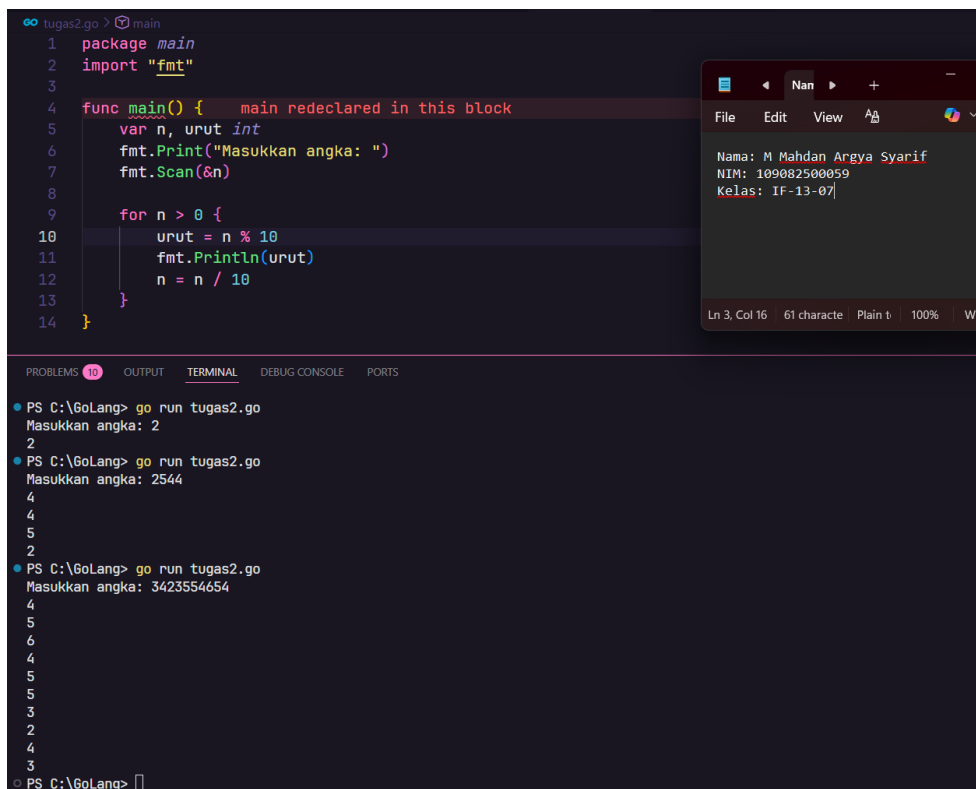
        fmt.Println(urut)

        n = n / 10

    }

}
```

### Screenshoot program



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

PS C:\GoLang> go run tugas2.go  
Masukkan angka: 2  
2

PS C:\GoLang> go run tugas2.go  
Masukkan angka: 2544  
4  
4  
5  
2

PS C:\GoLang> go run tugas2.go  
Masukkan angka: 3423554654  
4  
5  
6  
4  
5  
5  
3  
2  
4  
3

PS C:\GoLang>

### Deskripsi program

Program ini dirancang untuk menguraikan dan menampilkan kembali digit-digit dari sebuah bilangan bulat dimulai dari posisi paling belakang (kanan) ke depan, di mana outputnya ditampilkan secara berurutan dari atas ke bawah. Proses ini melibatkan dua variabel bertipe integer, yaitu *n* sebagai input pengguna dan *urut* sebagai variabel penampung digit saat ini. Algoritma utamanya bekerja secara iteratif dengan mengambil digit terakhir menggunakan operasi modulus ( $n \% 10$ ) yang hasilnya disimpan dalam variabel *urut* untuk kemudian dicetak, lalu dilanjutkan dengan memangkas digit tersebut dari bilangan utama melalui operasi pembagian ( $n / 10$ ). Langkah-langkah ini diulang terus-menerus hingga nilai *n* habis atau mencapai 0; sebagai ilustrasi, jika input adalah 2544, proses akan menghasilkan output 4, 4, 5, dan 2 secara berurutan, dengan ketentuan bahwa logika ini berlaku selama input yang diberikan tidak bernilai negatif ( $n \geq 0$ ).

### 3. Tugas 3

#### Source code

```
package main
import "fmt"

func main() {
    var x, y, hasil int
    fmt.Print("Masukkan bilangan: ")
    fmt.Scan(&x, &y)

    hasil = 0

    for x >= y {
        x = x - y
        hasil ++
    }
    fmt.Print(hasil)
}
```

## Screenshoot program

```
1 package main
2 import "fmt"
3
4 func main() {
5     var x, y, hasil int
6     fmt.Print("Masukkan bilangan: ")
7     fmt.Scan(&x, &y)
8
9     hasil = 0
10
11     for x >= y {
12         x = x - y
13         hasil ++
14     }
15     fmt.Print(hasil)
16 }
```

PROBLEMS 10 OUTPUT TERMINAL DEBUG CONSOLE PORTS

PS C:\GoLang> go run tugas3.go  
Masukkan bilangan: 5 2  
2

PS C:\GoLang> go run tugas3.go  
Masukkan bilangan: 10 7  
1

PS C:\GoLang> go run tugas3.go  
Masukkan bilangan: 120 4  
30

PS C:\GoLang>

## Deskripsi program

Program ini bertujuan untuk menghitung hasil pembagian bilangan bulat ( $x \div y$ ) menggunakan metode pengurangan berulang alih-alih menggunakan operator pembagian langsung. Algoritma intinya bekerja dengan melakukan perulangan selama nilai  $x \geq y$ , di mana pada setiap iterasi, variabel penampung hasil akan diinkremen (+1) dan nilai  $x$  akan diperbarui melalui pengurangan dengan  $y$  ( $x - y$ ). Sebagai ilustrasi, untuk input  $x=5$  dan  $y=2$ , program akan memvalidasi  $5 \geq 2$  (benar) lalu mengubah hasil menjadi 1 dan  $x$  menjadi 3, kemudian berlanjut memvalidasi  $3 \geq 2$  (benar) yang mengubah hasil menjadi 2 dan  $x$  menjadi 1, perulangan pun berhenti ketika kondisi  $1 \geq 2$  bernilai *false*, sehingga menetapkan 2 sebagai hasil akhir pembagian tersebut.