

**LAPORAN PRAKTIKUM**  
**ALGORITMA DAN PEMROGRAMAN 1**  
**MODUL V & VI**  
**“FOR - LOOP”**



**Disusun oleh :**  
**NAMA : Felix Pedrosa Valentino**  
**NIM : 103112400056**  
**S1 IF-12-01**

**Dosen Pengampu :**  
**Yohani Setiya Rafika Nur, M. Kom.**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2024/2025**

## DASAR TEORI

### Paradigma Perulangan

Perulangan merupakan salah satu struktur kontrol yang memungkinkan suatu instruksi yang sama dilakukan berulang kali dalam waktu atau jumlah yang lama. Tanpa instruksi perulangan, maka suatu instruksi akan ditulis dalam jumlah yang sangat banyak.

Tugas utama komputer adalah melakukan suatu instruksi/proses secara berulang dan terus-menerus tanpa adanya perbedaan. Hal ini berbeda dengan manusia yang bila melakukan hal sama secara berulang bisa melakukan kesalahan. Setiap baris instruksi dieksekusi satu persatu. Satu atau lebih instruksi bisa dieksekusi berulang kali

### Syarat Perulangan :

- Perulangan harus berhenti
- Apabila perulangan tidak pernah berhenti, maka algoritmanya salah (proses lama  $\neq$  tidak pernah berhenti)
- Pemrogram harus mengetahui perulangan akan berhenti/tidak sebelum program dijalankan

### Jenis Instruksi Perulangan :

- Berdasarkan jumlah iterasi
- Berdasarkan kondisi (kapan harus diulangi/berhenti)

### Perulangan berdasarkan Iterasi

- Untuk menyelesaikan kasus dengan jumlah iterasi diketahui di awal
- Iterasi adalah variabel bertipe integer untuk menampung iterasi perulangan
- Variabel i, init dan end adalah variabel atau nilai bertipe integer
- Aksi dieksekusi berulang sebanyak end init +1 kali (iterasi)
- Setiap aksi dieksekusi, nilai dari variabel iterasi selalu bertambah 1, dari init hingga end.

## Perulangan For-Loop

For loop adalah statement yang digunakan untuk mengeksekusi suatu blok kode secara berulang. Biasanya digunakan untuk melakukan iterasi terhadap nilai yang ada pada data slice, array, atau pada data yang bersifat iterable (dapat dilakukan perulangan).

### Karakteristik For-Loop (Perulangan berdasarkan iterasi)

Salah satu instruksi perulangan yang paling mudah adalah **for-loop**, yang mana dengan instruksi ini dapat digunakan untuk mengulangi instruksi sebanyak ***n*** kali (iterasi). Batasan besar nilai dari ***n*** menyesuaikan dengan batasan dari tipe data integer yang digunakan.

Instruksi for-loop memiliki beberapa komponen, yaitu :

1) **inisialisasi** merupakan assignment **variabel iterasi** yang bertipe integer. Pada contoh di

atas biasanya **variabel iterasi = 0** atau **1**, artinya iterasi dimulai dari 0 atau 1.

2) **kondisi** merupakan suatu operasi bernilai boolean yang menyatakan kapan perulangan

harus dilakukan. Pada contoh di atas **kondisi** adalah **variabel iterasi <= n** (kurang dari

atau sama dengan)

3) **update** merupakan ekspresi yang menyatakan perubahan nilai dari **variabel iterasi**.

## CONTOH SOAL

### 1.) Contoh Soal 1

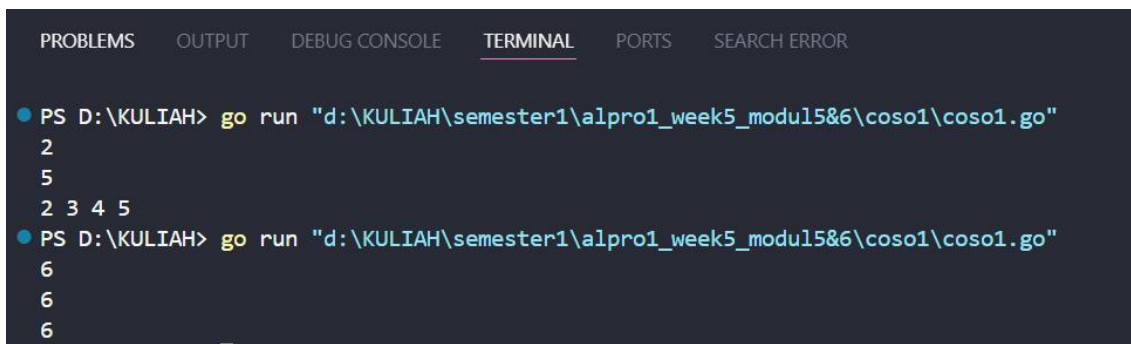
Source Code :

```
package main

import "fmt"

func main() {
    var a, b int
    var j int
    fmt.Scan(&a, &b)
    for j = a; j <= b; j = j + 1 {
        fmt.Print(j, " ")
    }
}
```

Output :

A screenshot of a terminal window showing the execution of a Go program. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (selected), PORTS, and SEARCH ERROR. The first command is 'PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1\_week5\_modul5&6\coso1\coso1.go"', which produces the output '2\n5\n2 3 4 5'. The second command is 'PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1\_week5\_modul5&6\coso1\coso1.go"', which produces the output '6\n6\n6'.

Deskripsi Program :

Program di atas adalah program yang melakukan input dan output berbasis teks untuk mencetak rentang angka dari sebuah bilangan awal hingga bilangan akhir.

Penjelasan dari setiap bagian program :

1. Deklarasi Variabel :

- Tiga variabel dideklarasikan: a, b, dan j. Semua variabel ini bertipe int (bilangan bulat).

- a adalah bilangan awal dari rentang.
- b adalah bilangan akhir dari rentang.
- j digunakan sebagai variabel iterasi dalam perulangan.

## 2. Input :

- Program meminta pengguna untuk memasukkan dua nilai integer (bilangan bulat), yaitu a dan b, menggunakan fungsi `fmt.Scan(&a, &b)`.

## 3. Proses Perulangan :

- Program menggunakan perulangan `for` untuk mencetak semua angka dari a hingga b.
- Perulangan dimulai dengan `j = a` dan akan terus berulang hingga `j` lebih besar dari b.
- Pada setiap iterasi, nilai `j` akan bertambah 1 (`j = j + 1`).

## 4. Output :

- Setiap angka dalam rentang `[a, b]` akan dicetak dalam satu baris, dipisahkan dengan spasi menggunakan `fmt.Print(j, " ")`.

## 2.) Contoh Soal 2

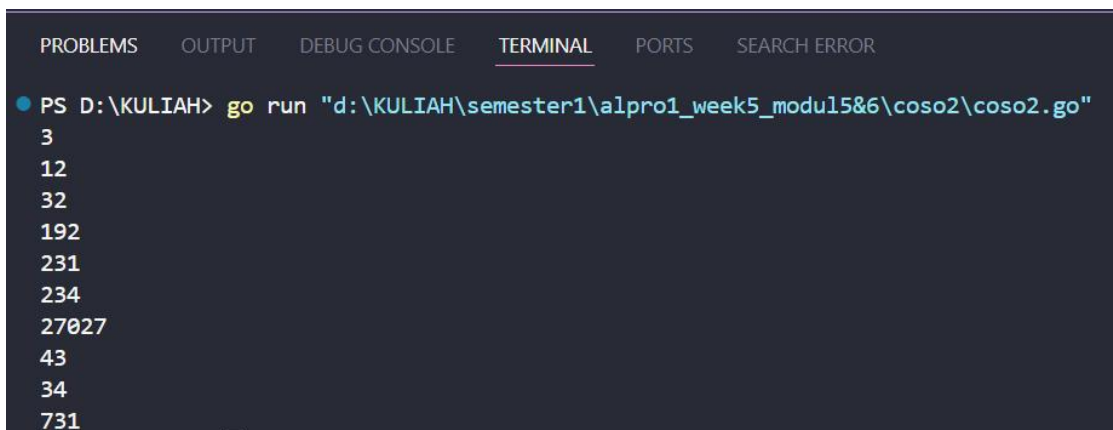
Source Code :

```
package main

import "fmt"

func main() {
    var j, alas, tinggi, n int
    var luas float64
    fmt.Scan(&n)
    for j = 1; j <= n; j += 1 {
        fmt.Scan(&alas, &tinggi)
        luas = 0.5 * float64(alas*tinggi)
        fmt.Println(luas)
    }
}
```

Output :



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\coso2\coso2.go"
3
12
32
192
231
234
27027
43
34
731
```

Deskripsi Program :

Program di atas adalah program yang menghitung luas beberapa segitiga berdasarkan input pengguna.

Penjelasan dari setiap bagian program :

1. Deklarasi Variabel :

- Empat variabel dideklarasikan: j, alas, tinggi, dan n bertipe int, serta luas bertipe float64.
  - n adalah jumlah segitiga yang akan dihitung luasnya.
  - alas dan tinggi menyimpan nilai alas dan tinggi dari masing-masing segitiga.
  - luas digunakan untuk menyimpan hasil perhitungan luas setiap segitiga.

2. Input :

- Program meminta pengguna untuk memasukkan sebuah nilai integer n, yaitu jumlah segitiga yang akan dihitung luasnya, menggunakan `fmt.Scan(&n)`.

3. Proses Perulangan :

- Program menggunakan perulangan for dengan variabel j yang berjalan dari 1 hingga n untuk menghitung luas sebanyak n kali.
- Pada setiap iterasi, program meminta pengguna untuk memasukkan dua nilai integer, yaitu alas dan tinggi segitiga, menggunakan `fmt.Scan(&alas, &tinggi)`.

#### 4. Perhitungan Luas :

- Luas segitiga dihitung menggunakan rumus :

$$luas = \frac{1}{2} \times alas \times tinggi$$

#### 5. Output :

- Setelah luas dihitung, program mencetak hasil perhitungan luas untuk setiap segitiga menggunakan `fmt.Println(luas)`.

### 3.) Contoh Soal

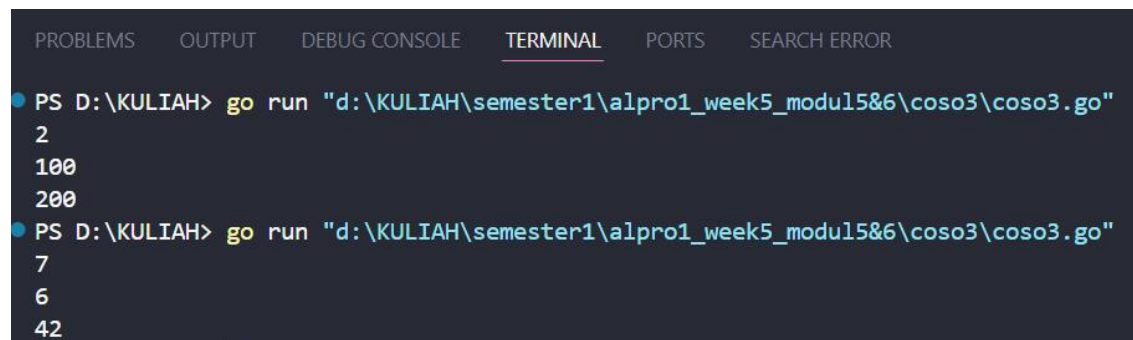
Source Code :

```
package main

import "fmt"

func main() {
    var j, hasil, v1, v2 int
    fmt.Scan(&v1, &v2)
    for j = 1; j <= v2; j++ {
        hasil = hasil + v1
    }
    fmt.Print(hasil)
}
```

Output :



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\coso3\coso3.go"
2
100
200
PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\coso3\coso3.go"
7
6
42
```

Deskripsi Program :

Program di atas adalah program yang melakukan operasi perkalian dua bilangan bulat dengan menggunakan penjumlahan berulang.

Penjelasan dari setiap bagian program :

1. Deklarasi Variabel :

- Tiga variabel dideklarasikan: j, hasil, v1, dan v2 bertipe int.
  - v1 adalah bilangan yang akan dijumlahkan secara berulang.
  - v2 adalah jumlah pengulangan penjumlahan bilangan v1.
  - hasil digunakan untuk menyimpan hasil akhir dari penjumlahan berulang. Pada awalnya, hasil memiliki nilai default 0.

2. Input :

- Program meminta pengguna untuk memasukkan dua bilangan bulat (v1 dan v2) menggunakan `fmt.Scan(&v1, &v2)`.
- v1 adalah bilangan yang akan dikalikan, dan v2 adalah berapa kali v1 akan dijumlahkan untuk mencapai hasil perkalian.

3. Proses Perulangan :

- Program menggunakan perulangan for untuk menjumlahkan v1 sebanyak v2 kali.
- Perulangan dimulai dari  $j = 1$  dan berjalan hingga  $j \leq v2$ , yang berarti program akan melakukan penjumlahan v1 sebanyak v2 kali.
- Pada setiap iterasi, nilai v1 akan ditambahkan ke dalam variabel hasil.

4. Output :

- Setelah perulangan selesai, hasil akhir dari penjumlahan berulang tersebut, yaitu perkalian antara v1 dan v2, dicetak menggunakan `fmt.Print(hasil)`.



## LATIHAN SOAL

### 1.) Latihan Soal 1

Source code :

```
package main

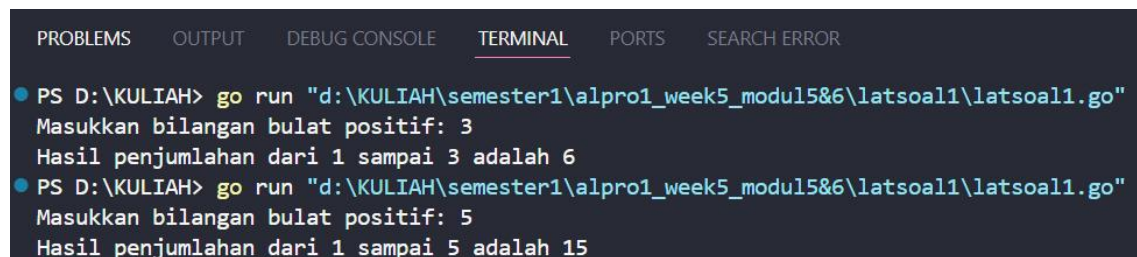
import "fmt"

func main() {
    var n, total int
    fmt.Print("Masukkan bilangan bulat positif: ")
    fmt.Scan(&n)

    // Menjumlahkan bilangan dari 1 hingga n
    for i := 1; i <= n; i++ {
        total += i
    }

    // Mencetak hasil penjumlahan
    fmt.Println("Hasil penjumlahan dari 1 sampai", n, "adalah", total)
}
```

Output :



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\latsoal1\latsoal1.go"
Masukkan bilangan bulat positif: 3
Hasil penjumlahan dari 1 sampai 3 adalah 6
PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\latsoal1\latsoal1.go"
Masukkan bilangan bulat positif: 5
Hasil penjumlahan dari 1 sampai 5 adalah 15
```

Deskripsi Program :

Program diatas adalah program untuk menghitung jumlah bilangan dari 1 hingga n secara sederhana menggunakan perulangan, di mana nilai-nilai bilangan dijumlahkan satu per satu.

Penjelasan dari setiap bagian program :

1. Deklarasi Variabel :

- Dua variabel dideklarasikan :
  - *n* bertipe *int*, yang menyimpan bilangan bulat positif yang dimasukkan oleh pengguna.
  - *total* bertipe *int*, yang digunakan untuk menyimpan hasil penjumlahan.
- 2. Input :
  - Program meminta pengguna memasukkan sebuah bilangan bulat positif *n* menggunakan `fmt.Scan(&n)`.
  - *n* adalah batas akhir dari sekumpulan bilangan yang akan dijumlahkan (dari 1 hingga *n*).
- 3. Proses Perulangan :
  - Program menggunakan perulangan `for` untuk menjumlahkan bilangan dari 1 hingga *n*.
  - Pada setiap iterasi, nilai *i* (dimulai dari 1 hingga *n*) ditambahkan ke variabel *total*, yang menyimpan hasil penjumlahan.
- 4. Output :
  - Setelah perulangan selesai, hasil penjumlahan dari 1 hingga *n* dicetak menggunakan `fmt.Println`.

## 2.) Latihan Soal 2

Source code :

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var n int
    fmt.Print("Masukkan jumlah kerucut: ")
    fmt.Scan(&n)

    // Menghitung volume untuk setiap kerucut
```

```

for i := 1; i <= n; i++ {
    var jariJari, tinggi float64
    fmt.Printf("Masukkan jari-jari dan tinggi kerucut ke-%d: ", i)
    fmt.Scan(&jariJari, &tinggi)

    // Rumus volume kerucut:  $V = \frac{1}{3} * \pi * r^2 * t$ 
    volume := (1.0 / 3.0) * math.Pi * math.Pow(jariJari, 2) * tinggi

    // Mencetak volume kerucut
    fmt.Printf("Volume kerucut ke-%d adalah: %.14f\n", i, volume)
}
}

```

Output :



```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR
PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\latsoal2\latsoal2.go"
Masukkan jumlah kerucut: 1
Masukkan jari-jari dan tinggi kerucut ke-1: 3
4
Volume kerucut ke-1 adalah: 37.69911184307752

```

Deskripsi Program :

Program di atas merupakan program yang digunakan untuk menghitung volume beberapa kerucut berdasarkan input jari-jari alas dan tinggi masing-masing kerucut. Hasilnya dicetak dalam bentuk volume dengan dua digit desimal di belakang koma untuk setiap kerucut.

Penjelasan dari setiap bagian program :

1. Pustaka yang Digunakan :

- `fmt`: Digunakan untuk input dan output (menampilkan pesan, menerima input dari pengguna, dan mencetak hasil).
- `math`: Digunakan untuk perhitungan matematika, khususnya nilai  $\pi$  (Pi) dan fungsi pangkat (`math.Pow()`).

2. Deklarasi Variabel :

- `n` bertipe `int` digunakan untuk menyimpan jumlah kerucut yang akan dihitung volumenya.

- jariJari dan tinggi bertipe float64 digunakan untuk menyimpan nilai panjang jari-jari alas dan tinggi dari setiap kerucut yang dimasukkan oleh pengguna.

### 3. Input :

- Program pertama-tama meminta pengguna memasukkan bilangan bulat n yang merupakan jumlah kerucut yang akan dihitung volumenya.
- Setelah itu, untuk setiap kerucut, pengguna diminta memasukkan dua nilai: panjang jari-jari alas (jariJari) dan tinggi (tinggi) dari kerucut tersebut.

### 4. Rumus Volume Kerucut :

- Volume kerucut dihitung menggunakan rumus :

$$V = \frac{1}{3} \times \pi \times r \times r \times t$$

Keterangan :

- r adalah jari-jari alas kerucut.
- t adalah tinggi kerucut.
- $\pi$  (Pi) adalah konstanta yang diperoleh dari math.Pi
- math.Pow(jariJari, 2) digunakan untuk menghitung kuadrat dari jari-jari alas kerucut.

### 5. Proses Perulangan :

- Program menggunakan perulangan for untuk menghitung volume sebanyak n kerucut.
- Pada setiap iterasi, pengguna memasukkan nilai jari-jari dan tinggi dari kerucut.
- Volume kerucut kemudian dihitung menggunakan rumus yang telah dijelaskan, dan hasilnya disimpan dalam variabel volume.

### 6. Output :

- Setelah menghitung volume dari setiap kerucut, program mencetak hasil perhitungan dalam format desimal dengan 14 angka di belakang koma (menggunakan format %.14f).

### 3.) Latihan Soal 3

Source code :

```
package main

import "fmt"

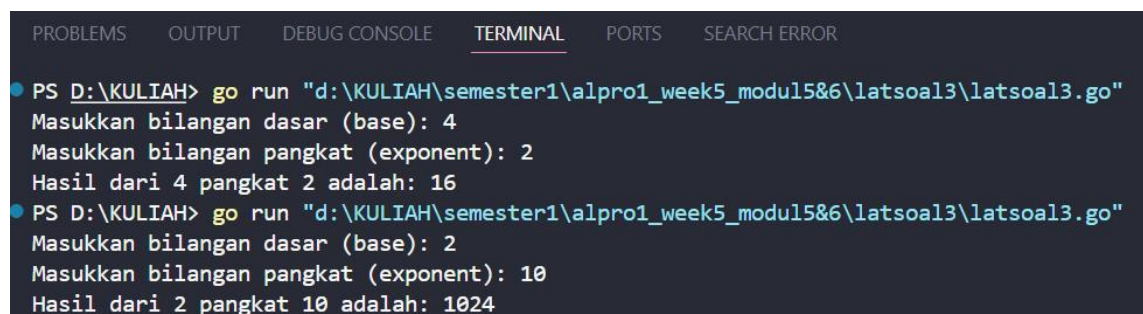
func main() {
    var base, exponent, result int
    fmt.Print("Masukkan bilangan dasar (base): ")
    fmt.Scan(&base)
    fmt.Print("Masukkan bilangan pangkat (exponent): ")
    fmt.Scan(&exponent)

    // Inisialisasi hasil dengan 1 (karena bilangan apapun
    // dipangkatkan 0 hasilnya 1)
    result = 1

    // Menghitung base^exponent menggunakan perkalian berulang
    for i := 1; i <= exponent; i++ {
        result *= base
    }

    // Mencetak hasil pemangkatan
    fmt.Printf("Hasil dari %d pangkat %d adalah: %d\n", base,
    exponent, result)
}
```

Output :



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\latsoal3\latsoal3.go"
Masukkan bilangan dasar (base): 4
Masukkan bilangan pangkat (exponent): 2
Hasil dari 4 pangkat 2 adalah: 16
PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\latsoal3\latsoal3.go"
Masukkan bilangan dasar (base): 2
Masukkan bilangan pangkat (exponent): 10
Hasil dari 2 pangkat 10 adalah: 1024
```

Deskripsi Program :

Program di atas merupakan program yang digunakan untuk menghitung hasil pemangkatan dua bilangan bulat positif dengan menggunakan operator perkalian dan struktur kontrol perulangan.

Penjelasan dari setiap bagian program :

#### 1. Deklarasi Variabel :

- base: Variabel yang menyimpan bilangan dasar (bilangan yang akan dipangkatkan).
- exponent : Variabel yang menyimpan bilangan pangkat (seberapa besar pangkat yang akan dihitung).
- result : Variabel yang menyimpan hasil akhir dari operasi pemangkatan.

#### 2. Input :

- Program meminta pengguna untuk memasukkan dua bilangan bulat positif :
  - base : Bilangan dasar (angka yang dipangkatkan).
  - exponent : Bilangan pangkat (eksponen).

#### 3. Inisialisasi :

- Variabel `result` diinisialisasi dengan nilai 1. Hal ini karena dalam pemangkatan, bilangan apapun dipangkatkan 0 memiliki hasil 1, dan perkalian dimulai dari nilai 1 agar tidak mengubah hasil perhitungan.

#### 4. Proses Perulangan :

- Program menggunakan struktur kontrol for untuk menghitung nilai pangkat dengan menggunakan operator perkalian.
- Perulangan berjalan sebanyak exponent kali (dari 1 hingga nilai exponent).
- Pada setiap iterasi, nilai result dikalikan dengan nilai base. Dengan kata lain, program menghitung hasil pemangkatan dengan mengalikan `base` dengan dirinya sendiri sebanyak exponent kali.

#### 5. Output :

- Setelah perulangan selesai, program mencetak hasil dari pemangkatan menggunakan `fmt.Printf` dengan format yang menyatakan hasil dari operasi pangkat.

#### 4.) Latihan Soal 4

Source code :

```
package main

import "fmt"

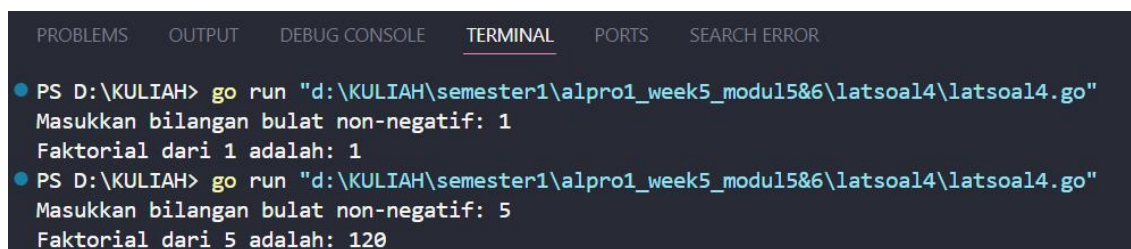
func main() {
    var n, result int
    fmt.Print("Masukkan bilangan bulat non-negatif: ")
    fmt.Scan(&n)

    // Inisialisasi hasil faktorial dengan 1 (karena faktorial 0 = 1)
    result = 1

    // Menghitung faktorial menggunakan perulangan
    for i := 1; i <= n; i++ {
        result *= i
    }

    // Mencetak hasil faktorial
    fmt.Printf("Faktorial dari %d adalah: %d\n", n, result)
}
```

Output :



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\latsoal4\latsoal4.go"
Masukkan bilangan bulat non-negatif: 1
Faktorial dari 1 adalah: 1
PS D:\KULIAH> go run "d:\KULIAH\semester1\alpro1_week5_modul5&6\latsoal4\latsoal4.go"
Masukkan bilangan bulat non-negatif: 5
Faktorial dari 5 adalah: 120
```

Deskripsi Program :

Program di atas merupakan program dalam bahasa Go (Golang) yang digunakan untuk menghitung hasil faktorial dari suatu bilangan bulat non-negatif.

Penjelasan dari setiap bagian program :

### 1. Deklarasi Variabel :

- n: Menyimpan bilangan bulat non-negatif yang akan dihitung faktorialnya. Ini adalah bilangan yang dimasukkan oleh pengguna.
- result: Menyimpan hasil perhitungan faktorial. Variabel ini diinisialisasi dengan nilai awal 1, karena faktorial 0 adalah 1 (secara definisi matematika,  $0! = 1$ ).

### 2. Input :

- Program meminta pengguna untuk memasukkan sebuah bilangan bulat non-negatif melalui fungsi `fmt.Scan(&n)`.
- Bilangan ini akan dihitung faktorialnya.

### 3. Inisialisasi :

- result diinisialisasi dengan nilai 1, karena faktorial bilangan 0 didefinisikan sebagai 1, dan semua faktorial mulai dari 1 akan dikalikan dengan hasil ini.

### 4. Proses Perulangan :

- Program menggunakan struktur perulangan for untuk menghitung nilai faktorial.
- Perulangan berjalan dari nilai 1 hingga nilai n.
- Pada setiap iterasi, variabel result dikalikan dengan nilai dari i, yang bertambah setiap kali iterasi (1, 2, 3, ..., n).
- Ini mensimulasikan proses faktorial, yaitu perkalian berturut-turut dari bilangan 1 hingga bilangan n (misalnya,  $5! = 1 * 2 * 3 * 4 * 5 = 120$ ).

### 5. Output :

- Setelah perulangan selesai, program mencetak hasil faktorial dari bilangan yang dimasukkan oleh pengguna menggunakan `fmt.Printf`.



## **DAFTAR PUSTAKA**

#18: For Loop - Belajar Golang Dari Dasar. (2022). Retrieved from <https://blog.ruangdeveloper.com/golang-for-loop/>

Go for Loop. (n.d.). Retrieved from <https://www.programiz.com/golang/for-loop>