

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 1
MODUL 5 & 6
“FOR-LOOP”



DISUSUN OLEH:
SHEILA STEPHANIE ANINDYA
103112400086
S1 IF-12-01
DOSEN:
Yohani Setiya Rafika Nur, M. Kom.

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

DASAR TEORI

I. 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.

Sebagai contoh adalah menuliskan suatu teks "CAK1BAB3 Algoritma Pemrograman 1" sebanyak 1000 baris. Maka tidak mungkin kita menuliskan instruksi `fmt.Println("CAK1BAB3 Algoritma Pemrograman 1")` sebanyak 1000x (walaupun memungkinkan menggunakan copy-paste).

Instruksi for-loop memiliki beberapa komponen, yaitu:

- **Inisialisasi** merupakan assignment variabel iterasi yang bertipe integer. Contoh biasanya variabel iterasi = 0 atau 1, artinya iterasi dimulai dari 0 atau 1.
- **Kondisi** merupakan suatu operasi bernilai boolean yang menyatakan kapan perulangan harus dilakukan. Contohnya kondisi adalah variabel iterasi $\leq n$ (kurang dari atau sama dengan).
- **Update** merupakan ekspresi yang menyatakan perubahan nilai dari variabel iterasi. Contohnya update adalah variabel iterasi = variabel iterasi + 1.

Beberapa situasi di mana penggunaan looping sangat dianjurkan:

- **Melakukan tugas berulang:** *looping* sangat berguna ketika perlu melakukan tugas yang sama berulang kali. Misalnya, jika ingin mencetak angka dari 1 sampai 100, menggunakan *looping* akan lebih efisien daripada menulis perintah print sebanyak 100 kali.
- **Menelusuri elemen dalam koleksi:** saat bekerja dengan array atau koleksi data lainnya, looping dipakai untuk menelusuri setiap elemen. Hal ini berguna untuk mencari elemen tertentu, melakukan operasi pada setiap elemen, atau mengumpulkan informasi dari seluruh koleksi.
- **Implementasi algoritma:** banyak algoritma dalam pemrograman, seperti algoritma pengurutan atau pencarian, memerlukan penggunaan looping untuk memproses data secara efisien.
- **Pengolahan data berkelanjutan:** dalam kasus di mana program perlu terus menerus memproses data yang masuk, seperti dalam aplikasi server atau game, looping berfungsi untuk menjalankan proses tersebut secara berkelanjutan sampai kondisi tertentu terpenuhi.
- **Pembuatan menu interaktif:** dalam pengembangan aplikasi dengan menu interaktif, looping sering digunakan untuk menampilkan menu dan menangani pilihan *user* secara berulang sampai user memutuskan untuk keluar dari aplikasi.

CONTOH SOAL

1. Latihan 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:

```
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> go run "c:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou\coso\c1.go"
2 5
2 3 4 5
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> |
```

Deskripsi Program:

Program ini membaca dua bilangan bulat, lalu akan mencetak semua angka yang diinput secara urut seperti pada output diatas.

Pseudocode:

- 1) Mulai
- 2) Deklarasikan variabel a, b, dan j bertipe integer.
- 3) Input dua bilangan bulat dan simpan dalam a dan b.
- 4) Melakukan loop dari a sampai b. Setiap iterasi, nilai j akan dicetak dengan spasi setelahnya.
- 5) Program selesai.

2. Latihan 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:

```
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> go run "c:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou\coso\c2.go"
3
12 32
192
231 234
27027
43 34
731
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> |
```

Deskripsi Program:

Program ini menghitung luas beberapa segitiga dengan rumus

$$\text{Luas} = \frac{1}{2} \times \text{alas} \times \text{tinggi}$$

Pseudocode :

- 1) Mulai
- 2) Deklarasikan variabel `j`, `alas`, `tinggi`, `n` bertipe integer.
- 3) Deklarasikan variabel `luas` bertipe float64
- 4) Input jumlah segitiga dengan variable `n`.
- 5) Melakukan loop dari 1 hingga `n`. Pada setiap iterasi:
 - 1) Meminta input alas dan tinggi segitiga.
 - 2) Menghitung luas segitiga menggunakan rumus `0.5 * alas * tinggi`.
 - 3) Mencetak hasil perhitungan luas segitiga.
- 6) Program selesai.

3. Latihan 3

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:

```
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> go run "c:\Users\sheila\OneDrive\
res\Sheila SA\rindou\coso\c3.go"
2 100
200
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> |
```

Deskripsi Program:

Program ini menghitung hasil perkalian dua buah bilangan tanpa menggunakan operator kali.

Pseudocode :

- 1) Mulai.
- 2) Deklarasikan variable j, hasil, v1, v2 bertipe integer.
- 3) Input v1 dan v2.
- 4) Program melakukan loop v2 kali, setiap iterasi v1 ditambahkan ke variable hasil.
- 5) Cetak hasil.
- 6) Program selesai.

SOAL LATIHAN

Statement perulangan

1. Buatlah program untuk menjumlahkan sekumpulan bilangan. Masukan terdiri dari suatu bilangan bulat positif n. Keluaran berupa bilangan hasil penjumlahan dari 1 sampai dengan n.

Source Code:

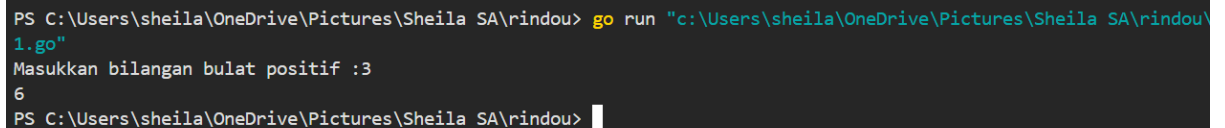
```
package main

import (
    "fmt"
)

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

    sum := 0
    for j := 1; j <= n; j++ {
        sum += j
    }
    fmt.Print(sum)
}
```

Output:



```
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> go run "c:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou\1.go"
Masukkan bilangan bulat positif :3
6
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> 
```

Deskripsi Program:

Program ini menghitung jumlah (penjumlahan) dari semua bilangan bulat mulai dari 1 hingga bilangan tersebut, lalu mencetak hasilnya.

Pseudocode :

- 1) Mulai.
- 2) Deklarasikan variabel `n` sebagai integer.
- 3) Print "Masukkan bilangan bulat positif:".
- 4) Ambil input dari pengguna dan simpan di variabel `n`.
- 5) Deklarasikan variabel `sum` dan inisialisasi dengan nilai 0.
- 6) Tampilkan nilai `sum` (jumlah dari semua bilangan 1 hingga `n`).
- 7) Program selesai.

Statement perulangan

2. Buatlah program yang digunakan untuk menghitung volume sejumlah n kerucut, apabila diketahui panjang jari-jari alas kerucut dan tinggi dari kerucut. Masukan terdiri dari beberapa baris. Baris pertama adalah bilangan bulat n , selanjutnya n baris berikutnya masing-masing merupakan panjang jari-jari alas kerucut dan tinggi dari kerucut. Keluaran terdiri dari beberapa baris, yang masing-masingnya menyatakan volume dari n kerucut

Source Code:

```
package main

import (
    "fmt"
    "math"
)

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

    var jariJari, tinggi, volume, volumeTotal float64

    for j := 0; j < n; j++ {
        fmt.Print("Masukkan jari-jari kerucut: ", j+1)
        fmt.Scan(&jariJari)

        fmt.Print("Masukkan tinggi kerucut: ", j+1)
        fmt.Scan(&tinggi)

        volume = (1.0 / 3.0) * math.Pi * math.Pow(jariJari, 2) * tinggi
        volumeTotal += volume

        fmt.Printf("Volume kerucut ke-%d: %.2f\n", j+1, volume)
    }
    fmt.Print(volumeTotal)
}
```

Output:

```
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> go run "c:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou\main.go"
Masukkan n kerucut: 1
Masukkan jari-jari kerucut: 13
Masukkan tinggi kerucut: 14
Volume kerucut ke-1: 37.70
37.699111843077524
```

Deskripsi Program:

Program tersebut menghitung volume beberapa kerucut berdasarkan input dari pengguna, seperti jari-jari dan tinggi kerucut. Program meminta jumlah kerucut yang akan dihitung, kemudian menghitung volume setiap kerucut menggunakan rumus volume kerucut dan menampilkan total volume dari semua kerucut tersebut.

Pseudocode :

- 1) Mulai.
- 2) Deklarasikan variabel `n` bertipe integer.
- 3) Print "Masukkan n kerucut:".
- 4) Input `n` (jumlah kerucut).
- 5) Deklarasikan variabel `jariJari`, `tinggi`, `volume`, dan `volumeTotal` bertipe float64.
- 6) Print "Masukkan jari-jari kerucut ke-x:".
- 7) Input `jariJari`.
- 8) Print "Masukkan tinggi kerucut ke-x:".
- 9) Input `tinggi`.
- 10) Hitung volume kerucut menggunakan rumus:
$$\text{volume} = \frac{1}{3} \times \pi \times r^2 \times t$$
- 11) Tambahkan volume kerucut tersebut ke `volumeTotal`.
- 12) Print volume kerucut ke-x.
- 13) Print total volume dari semua kerucut.
- 14) Program selesai.

Statement perulangan

3. Buatlah program yang digunakan untuk menghitung hasil pemangkatan dari dua buah bilangan. Program dibuat dengan menggunakan operator perkalian dan struktur kontrol perulangan. Masukan terdiri dari dua bilangan bulat positif. Keluaran terdiri dari suatu bilangan yang menyatakan hasil bilangan pertama dipangkatkan dengan bilangan kedua.

Source Code:

```
package main

import "fmt"

func main() {
    var basis, pangkat int

    fmt.Print("Masukkan bilangan pertama (basis): ")
    fmt.Scan(&basis)

    fmt.Print("Masukkan biangan kedua (pangkat): ")
    fmt.Scan(&pangkat)

    hasil := 1
    for j := 0; j < pangkat; j++ {
        hasil *= basis
    }
    fmt.Print(basis, pangkat, hasil)
}
```


Output:

```
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> go run "c:\Users\sheila\OneDrive\Pictures\Sheila SA\rind
Masukkan bilangan pertama (basis): 4
Masukkan bilangan kedua (pangkat): 2
4 2 16
```

Deskripsi Program:

Program ini menghitung hasil pemangkatan dari dua buah bilangan, dengan menggunakan operator perkalian dan struktur kontrol perulangan. Tidak menggunakan fungsi bawaan matematika seperti `math.Pow`, melainkan loop sederhana.

Pseudocode :

- 1) Mulai.
- 2) Deklarasikan variabel `basis` dan `pangkat` bertipe integer.
- 3) Print "Masukkan bilangan pertama (basis):".
- 4) Input `basis`.
- 5) Print "Masukkan bilangan kedua (pangkat):"
- 6) Input `pangkat`.
- 7) Inisialisasi variabel `hasil` dengan nilai 1.
- 8) Kalikan `hasil` dengan `basis` (`hasil *= basis`).
- 9) Tampilkan nilai `basis`, `pangkat`, dan `hasil`.
- 10) Program selesai.

Statement perulangan

4. Buatlah program yang digunakan untuk menghitung hasil faktorial dari suatu bilangan. Masukan terdiri dari suatu bilangan bulat non negatif. Keluaran terdiri dari hasil faktorial dari bilangan bulat n.

Source Code:

```
package main

import "fmt"

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

    faktorial := 1
    for j := 1; j <= n; j++ {
        faktorial *= j
    }
    fmt.Print(n, faktorial)
}
```

Output:

```
PS C:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou> go run "c:\Users\sheila\OneDrive\Pictures\Sheila SA\rindou"
Masukkan bilangan bulat non negatif: 5
5 120
```

Deskripsi Program:

Program ini menghitung faktorial dari bilangan bulat non-negatif yang diinput pengguna.

Pseudocode :

- 1) Mulai.
- 2) Deklarasikan variabel `n` bertipe integer.
- 3) Print "Masukkan bilangan bulat non negatif:".
- 4) Input `n` dari pengguna.
- 5) Inisialisasi variabel `faktorial` dengan nilai 1.
- 6) Kalikan `faktorial` dengan `j` (faktorial *= j).
- 7) Print nilai `faktorial` (hasil faktorial dari `n`).
- 8) Program selesai.

DAFTAR PUSTAKA

- Modul Algoritma dan Pemrograman 5 & 6.
- RevoU. (2024). *Apa itu Looping? Arti, Fungsi, Contoh, FAQs*
<https://revou.co/kosakata/looping>