

**LAPORAN PRAKTIKUM**  
**AGORITMA PEMROGRAMAN**

**“PERULANGAN FOR”**

**DISUSUN OLEH:**

**NABIL FIKRI**

**2511533011**

**DOSEN PENGAMPU:**

**Dr. WAHYUDI, S.T, M.T**

**ASISTEN PRAKTIKUM:**

**JOVANTRI IMMANUEL GULO**



**DEPARTEMEN INFORMATIKA**  
**FAKULTAS TEKNOLOGI INFORMASI**  
**UNIVERSITAS ANDALAS**

**2025**

## KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT yang telah memberikan kelimpahan rahmat dan hidayahnya yaitu kesehatan dan kekuatan , atas rahmat tersebut penulis dapat mengerjakan laporan pratikum pekan ke-5 ini dengan tepat waktu. Laporan ini disusun sebagai tugas mingguan mata kuliah Pratikum Algoritma Pemrograman yaitu dengan topik “Perulangan *for*” pada Java.

Melalui laporan ini, penulis bertujuan untuk mempelajari serta memahami konsep pada perulangan For dalam bahasa Java, khususnya penggunaan struktur *for* dan *Nested for*. Kedua jenis perulangan tersebut memiliki peranan penting dalam pengembangan program yang efisien, terstruktur, dan mudah dikelola. Pratikum ini juga diharapkan dapat membantu mahasiswa dalam mengasah kemampuan logika serta pemahaman terhadap alur eksekusi program.

Penulis menyadari laporan ini masih jauh dari sempurna, oleh karena itu kritik dan saran sangat diharapkan demi perbaikan ke depannya. Semoga laporan ini dapat bermanfaat bagi penulis maupun pembaca.

Padang, 31 Oktober 2025

Nabil Fikri

## DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI .....	ii
BAB I PENDAHULUAN .....	3
1.1 Latar Belakang .....	3
1.2 Tujuan.....	3
1.3 Manfaat.....	4
BAB II PEMBAHASAN .....	5
2.1 Pengertian .....	5
2.2 Kode Program.....	6
2.2.1 PerulanganFor1 .....	6
2.2.2 PerulanganFor2 .....	6
2.2.3 PerulanganFor3 .....	7
2.2.4 PerulanganFor4 .....	8
2.2.5 Perulangan Nested For 0.....	9
2.2.6 Perulangan Nested For 1.....	10
2.2.7 Perulangan Nested For 2.....	11
BAB III PENUTUP .....	12
3.1 Kesimpulan .....	12
3.2 Saran.....	12
DAFTAR PUSTAKA.....	13

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Dalam pemrograman, perulangan merupakan salah satu konsep dasar yang berperan penting dalam efisiensi dan efektivitas penulisan kode. Perulangan digunakan untuk menjalankan suatu blok perintah secara berulang sesuai dengan jumlah yang ditentukan atau selama kondisi tertentu terpenuhi. Salah satu bentuk perulangan yang paling umum dan sering digunakan dalam bahasa pemrograman Java adalah perulangan *for*. Struktur perulangan *for* memudahkan programmer dalam mengontrol jumlah iterasi karena memiliki inisialisasi, kondisi, dan perubahan nilai variabel yang terorganisir di satu baris perintah.

Dalam praktik pemrograman Java, perulangan *for* sangat berguna ketika kita perlu melakukan tugas yang berulang, seperti menampilkan data dalam jumlah tertentu, menghitung hasil operasi matematis secara bertahap, atau mengakses elemen dalam array. Penggunaan *for loop* tidak hanya membuat kode lebih ringkas, tetapi juga meningkatkan keterbacaan dan meminimalkan potensi kesalahan dibandingkan dengan menulis perintah berulang secara manual. Oleh karena itu, pemahaman terhadap konsep dan penerapan perulangan *for* menjadi dasar penting bagi mahasiswa atau programmer pemula dalam mempelajari logika dan alur eksekusi program yang efisien.

### **1.2 Tujuan**

1. Memberikan mahasiswa pengetahuan baru tentang penggunaan perulangan *for* dan nested *for*.
2. Memahami penggunaan perulangan *for* di berbagai situasi pada program Java, serta penulisan sintaks, kelas, dan metode utama.
3. Melatih keterampilan mahasiswa dalam menulis, mengompilasi, dan mengeksekusi program sederhana menggunakan perulangan *for*.
4. Memberikan pemahaman yang matang dengan memberikan mahasiswa Latihan menggunakan program perulangan *for*.

### 1.3 Manfaat

1. Mahasiswa dapat memahami konsep perulangan *for* secara praktis, bukan hanya teori.
2. Melatih keterampilan logika dan analisis dalam menyusun serta mengeksekusi kode.
3. Memberikan pengalaman langsung dalam menggunakan bahasa Java sebagai salah satu bahasa pemrograman populer.
4. Menjadi bekal awal untuk memahami materi pemrograman yang lebih kompleks pada praktikum berikutnya.
5. Membiasakan mahasiswa dengan proses debugging dan pemecahan masalah dalam pemrograman.

## BAB II

### PEMBAHASAN

#### 2.1 Pengertian

Perulangan *for* dalam Java adalah suatu bentuk proses pengulangan kegiatan pada suatu statement atau lebih secara berulang ulang-ulang selama yang dijadikan acuan tersebut terpenuhi dengan baik. Perulangan *for* ini kebalikan dari perulangan *while*, dimana penggunaan perulangan *while* digunakan ketika jumlah perulangannya tidak ditentukan, sedangkan perulangan *for* digunakan bila jumlah perulangan sudah ditentukan. Jadi perulangan ini dapat dibagi menjadi 2 jenis yang berbeda, yang mana dapat kita bedakan menjadi *counted loop*, dan *uncounted loop*.

*Counted loop* sendiri adalah perulangan yang jumlah pengulangannya sudah diketahui atau ditentukan sejak awal. Lebih tepatnya kita tahu pasti berapa kali loop tersebut akan dijalankan. Sedangkan *uncounted loop* adalah perulangan yang jumlah perulangannya tidak diketahui sebelumnya. *Loop* akan terus berjalan selama suatu kondisi berjalan benar (*true*), dan berhenti ketika kondisi itu menjadi salah (*false*). Dalam kondisi ini biasanya menggunakan struktur *while* atau *do-while*.

Dalam perulangan *for* terdapat 3 bentuk parameter yang harus diketahui, yaitu:

- Inisiasi, adalah bagian yang mana harus ada nilai awal yang dilakukan untuk memulai perulangan
- Kondisi, adalah suatu bagian yang menjelaskan bentuk kondisi perulangan agar perulangan tersebut dapat berjalan
- Modifer, adalah bagian yang melakukan increment atau decrement dari nilai awal yang telah diinisiasi

Selain itu, *Nested for* adalah perulangan *for* yang diletakkan di dalam perulangan *for* lainnya. Dengan kata lain, setiap kali perulangan luar dijalankan satu kali, perulangan dalam akan berjalan penuh. Struktur ini biasanya digunakan untuk membuat pola data bertingkat, seperti tabel, segitiga bintang (*pattern*), atau untuk mengakses elemen dua dimensi seperti array 2D.

## 2.2 Kode Program

Pada pekan-5 ini, mahasiswa telah melakukan pratikum mengenai berbagai program Perulangan *for* dalam bahasa Java. Berikut program – program yang telah dipelajari :

### 2.2.1 PerulanganFor1

```
1 package pekan5_2511533011;
2
3 public class PerulanganFor1_2511533011 {
4
5     public static void main(String[] args) {
6         for (int i = 1; i <= 10; i++) {
7             System.out.println(i);
8         }
9     }
10 }
11
12 }
13
```

Kode program 2.2.1

Program ini merupakan awal dari perulangan *for* dengan mengetahui batas *loop* yaitu dimulai dari angka 1–10. Perulangan akan terus terjadi jika hasil yang dikeluarkan  $\leq 10$  dan bernilai true. Program ini memberikan output satuan angka per baris dari 1-10.

### 2.2.2 PerulanganFor2

```
1 package pekan5_2511533011;
2
3 public class PerulanganFor2_2511533011 {
4
5     public static void main(String[] args) {
6         for (int i = 1; i <= 10; i++) {
7             System.out.print(i+" ");
8         }
9     }
10 }
11
12 }
13
```

Kode Program 2.2.2

Sama dengan program sebelumnya, hanya saja yang membedakan yaitu program ini menggunakan `System.out.print()` yang hanya menggunakan satu barisan dimulai dari angka 1-10.

### 2.2.3 PerulanganFor3

```

1 package pekan5_2511533011;
2
3 public class PerulanganFor3_2511533011 {
4
5     public static void main(String[] args) {
6         int jumlah = 0;
7         for (int i=1; i<=10;i++) {
8             System.out.print(i);
9             jumlah = jumlah+i;
10            if (i < 10) {
11                System.out.print(" + ");
12            }
13        }
14        System.out.println();
15        System.out.println("Jumlah = " + jumlah);
16    }
17
18 }
19

```

Kode Program 2.2.3

Pada program ini mahasiswa diminta untuk membuat operasi penjumlahan dengan *loop* yang telah dibatasi. Dengan menginisialisasi `jumlah = 0`, batas perulangan dimulai dari 1-10 dengan menggunakan ekspresi (`jumlah = jumlah+i`). Selanjutnya menggunakan If (jika) `i<10` maka output yang dikeluarkan yaitu tanda “+” disetiap perulangan. Jika nilai `i` pada if bernilai 10 maka bernilai false dan berhenti. Jika perulangan sudah mencapai batas maka output yang dikeluarkan “=” sebelum hasil akhir ditampilkan. Secara otomatis program akan mengeluarkan hasil penjumlahan angka 1-10 secara berurutan.



### 2.2.4 PerulanganFor4

```

1 package pekan5_2511533011;
2
3 import java.util.Scanner;
4
5 public class PerulanganFor4_2511533011 {
6
7     public static void main(String[] args) {
8         int jumlah=0;
9         int batas;
10        Scanner input = new Scanner (System.in);
11        System.out.print("Masukkan nilai batas = ");
12        batas = input.nextInt();
13        input.close();
14        for (int i = 1; i<=batas; i++) {
15            System.out.print(i);
16            jumlah = jumlah+i;
17            if (i < batas) {
18                System.out.print(" + ");
19            } else {
20                System.out.print(" = ");
21            }
22        }
23        System.out.println(jumlah);
24    }
25
26 }
27

```

Kode Program 2.2.4

Program di atas merupakan program Java yang menggunakan perulangan *For* untuk menghitung dan menampilkan hasil penjumlahan dari angka 1 hingga angka batas yang dimasukkan oleh pengguna. Pada awal program, digunakan class *Scanner* untuk menerima input dari keyboard. Mahasiswa diminta memasukkan nilai batas, kemudian program akan melakukan perulangan dari angka 1 sampai batas tersebut. Setiap angka yang dilewati perulangan akan ditampilkan di layar dan ditambahkan ke variabel *jumlah* sebagai total penjumlahan. Di dalam perulangan juga terdapat percabangan *if* yang berfungsi untuk menampilkan tanda “+” di antara angka, dan tanda “=” sebelum hasil akhir ditampilkan. Setelah perulangan selesai, program mencetak hasil total penjumlahan seluruh angka tersebut. Misalnya, jika pengguna memasukkan angka 5, maka output yang ditampilkan adalah 1 + 2 + 3 + 4 + 5 = 15. Program ini sederhana namun menggambarkan cara kerja perulangan, penjumlahan, serta penggunaan input dan output dalam Java secara efektif.

### 2.2.5 Perulangan Nested For 0

```
1 package pekan5_2511533011;
2
3 public class nestedFor0_2511533011 {
4
5     public static void main(String[] args) {
6         for (int line = 1; line <= 5; line++) {
7             for (int j = 1; j <= (-1 * line + 5); j++) {
8                 System.out.print(".");
9             }
10            System.out.print(line);
11            System.out.println();
12        }
13    }
14 }
15
16 }
17
```

Kode Program 2.2.5

Program Java pada gambar tersebut menggunakan *nested loop* (perulangan bersarang) untuk menampilkan pola tertentu di layar. Di dalam kode, terdapat dua perulangan *for*. Perulangan pertama dengan variabel *line* berfungsi sebagai pengontrol jumlah baris yang akan ditampilkan, yang berjalan dari 1 sampai 5. Di dalamnya terdapat perulangan kedua dengan variabel *j* yang mengatur jumlah karakter titik (.) yang dicetak di setiap baris. Nilai batas perulangan dalam *j* ditentukan oleh ekspresi  $(-1 * line + 5)$ , sehingga semakin besar nilai *line*, semakin sedikit jumlah titik yang dicetak.

Setelah perulangan dalam selesai, program mencetak angka sesuai nilai *line* di akhir setiap baris, lalu pindah ke baris berikutnya menggunakan `System.out.println()`. Hasil akhirnya akan membentuk pola seperti segitiga menurun.

### 2.2.6 Perulangan Nested For 1

```
1 package pekan5_2511533011;
2
3 public class nestedFor1_2511533011 {
4
5     public static void main(String[] args) {
6         for (int i = 1; i <= 5; i++) {
7             for (int j = 1; j <= 5; j++) {
8                 System.out.print("*");
9             }
10            System.out.println();
11            // to end the line
12        }
13    }
14 }
15
16 }
17
```

Kode Program 2.2.6

Program di atas merupakan contoh penggunaan perulangan bersarang (*nested loop*) dalam bahasa Java untuk mencetak pola bintang berbentuk persegi. Program ini memiliki dua perulangan for di dalam method main. Perulangan pertama menggunakan variabel i yang berfungsi untuk mengatur jumlah baris, dimulai dari 1 hingga 5. Di dalamnya terdapat perulangan kedua dengan variabel j yang juga berjalan dari 1 sampai 5, dan bertugas mencetak karakter bintang (\*) pada setiap baris. Setelah lima bintang dicetak, perintah `System.out.println();` dijalankan untuk berpindah ke baris berikutnya, sehingga bintang pada iterasi selanjutnya akan muncul di bawahnya. Hasil akhirnya, program akan menampilkan lima baris yang masing-masing berisi lima bintang, membentuk pola persegi berukuran 5x5. Program ini sederhana tetapi efektif untuk memahami konsep dasar perulangan ganda dan bagaimana cara menampilkan pola menggunakan karakter dalam Java.

### 2.2.7 Perulangan Nested For 2

```

1 package pekan5_2511533011;
2
3 public class nestedFor2_2511533011 {
4
5     public static void main(String[] args) {
6         for (int i = 0; i <= 5; i++) {
7             for (int j = 0; j <= 5; j++) {
8                 System.out.print(i + j + " ");
9             }
10            System.out.println();
11            // to end the line
12        }
13    }
14 }
15
16 }
17

```

Kode Program 2.2.7

Program Java pada gambar tersebut merupakan contoh penggunaan *nested loop* (perulangan bersarang) untuk menampilkan pola angka berdasarkan hasil penjumlahan dua variabel perulangan. Program ini memiliki dua perulangan *for*. Perulangan luar menggunakan variabel *i* yang berjalan dari 0 hingga 5 untuk mengatur baris, sedangkan perulangan dalam menggunakan variabel *j* yang juga berjalan dari 0 hingga 5 untuk mengatur kolom. Pada setiap iterasi perulangan dalam, program mencetak hasil dari operasi *i + j*, yaitu penjumlahan nilai dari kedua variabel tersebut, diikuti dengan spasi agar hasilnya rapi. Setelah perulangan dalam selesai, perintah `System.out.println();` digunakan untuk berpindah ke baris berikutnya, sehingga hasil berikutnya dicetak di baris baru.

Sebagai contoh, pada baris pertama (saat *i* = 0), nilai yang dicetak adalah 0 1 2 3 4 5, lalu pada baris berikutnya (saat *i* = 1), nilai yang dicetak menjadi 1 2 3 4 5 6, dan seterusnya hingga *i* = 5. Hasil akhir program ini membentuk tabel angka yang menampilkan penjumlahan dari dua indeks perulangan. Program ini membantu memahami bagaimana dua loop bekerja secara bersamaan dan bagaimana nilai dari variabel perulangan bisa dikombinasikan untuk menghasilkan pola numerik tertentu.

## **BAB III**

### **PENUTUP**

#### **3.1 Kesimpulan**

Berdasarkan hasil praktikum yang telah dilakukan, dapat disimpulkan bahwa perulangan *for* merupakan salah satu struktur dasar penting dalam bahasa pemrograman Java yang berfungsi untuk mengeksekusi suatu perintah secara berulang dengan jumlah iterasi yang telah ditentukan. Dengan memahami cara kerja perulangan *for*, penulisan kode menjadi lebih efisien, terstruktur, dan mudah dipahami. Selain itu, penggunaan *nested for* atau perulangan bersarang memberikan fleksibilitas lebih dalam membentuk pola dan melakukan operasi dua dimensi, seperti mencetak tabel angka, pola bintang, maupun proses penjumlahan bertingkat. Melalui praktikum ini, mahasiswa tidak hanya memahami teori perulangan, tetapi juga mampu menerapkannya secara langsung dalam program yang logis dan terarah, sehingga keterampilan berpikir algoritmik dapat meningkat.

#### **3.2 Saran**

Untuk meningkatkan efektivitas kegiatan praktikum ke depannya, disarankan agar penyampaian materi dilakukan secara lebih interaktif dengan disertai contoh-contoh program yang relevan dan bervariasi. Dosen maupun asisten laboratorium sebaiknya memberikan penjelasan yang lebih mendalam mengenai kesalahan umum yang sering terjadi saat menulis kode agar mahasiswa dapat belajar dari kesalahan tersebut. Selain itu, waktu pelaksanaan praktikum diharapkan dapat disesuaikan agar peserta memiliki kesempatan yang cukup untuk mencoba, memahami, dan mendiskusikan hasil programnya. Praktikum berikutnya juga sebaiknya disertai latihan tambahan atau mini proyek agar mahasiswa dapat memperkuat pemahaman konsep dan penerapannya dalam kasus nyata.

## DAFTAR PUSTAKA

- [1] Wikipedia, “Pengulangan for ,”[Daring]. Tersedia pada:  
[https://id.wikipedia.org/wiki/Perulangan\\_for](https://id.wikipedia.org/wiki/Perulangan_for) [Diakses: 31-Oktober-2025]