

# **LAPORAN PRATIKUM ALGORITMA PEMROGRAMAN**

## **PERULANGAN PADA JAVA**

DI SUSUN OLEH :

ABDUL KARIM ALGAZALI

NIM 2511532029

DOSEN PENGAMPU : Dr. WAHYUDI, S.T, M.T

ASISTEN LABORATORIUM: AUFAN TAUFIQURRAHMAN



**DEPARTEMEN INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**UNIVERSITAS ANDALAS**

**PADANG**

**2025**

## **KATA PENGANTAR**

Sebelumnya saya Panjatkan Puji syukur atas kehadiran Tuhan Yang Maha Esa, berkat rahmat dan izinnya juga laporan praktikum “Perulangan Pada Java” ini dapat diselesaikan dengan baik. Saya ucapkan terima kasih yang sebesar besarnya kepada Dr. Wahyudi, S.T, M.T selaku dosen pengampu yang telah membimbing Mata kuliah Algoritma dan Pemrograman. Dan tidak lupa saya ucapkan terima kasih kepada Uda Aufan Tafiqurrahman selaku asisten labor yang telah membimbing praktikum Perulangan Pada Java

Laporan ini disusun sebagai Bentuk hasil kegiatan praktikum yang bertujuan untuk memahami konsep dasar Perulangan Pada java dalam pemrograman. Penulis menyadari bahwa pemahaman mendalam terhadap kedua topik ini merupakan fondasi essential bagi pengembangan keterampilan pemrograman yang lebih advanced.

Saya menyadari bahwa penulisan laporan praktikum ini masih jauh dari kata sempurna baik dari segi pembahasan dan penulisannya. namun dengan demikian saya telah berupaya dengan segala kemampuan dan pengetahuan yang dimiliki supaya laporan ini dapat selesai dengan baik dan oleh karenanya saya dengan rendah hati menerima saran, masukan dan kritikan guna penyempurnaan laporan ini.

Padang, 30 Oktober 2025

Penulis

## DAFTAR ISI

<b>KATA PENGANTAR .....</b>	<b>i</b>
<b>DAFTAR ISI .....</b>	<b>ii</b>
<b>BAB I PENDAHULUAN .....</b>	<b>3</b>
1.1 Latar Belakang .....	3
1.2 Tujuan.....	3
1.3 Manfaat .....	3
<b>BAB II PEMBAHASAN .....</b>	<b>4</b>
2.1 Perulangan Pada Java .....	4
2.2 Langkah pengerjaan .....	4
<b>BAB III KESIMPULAN .....</b>	<b>13</b>
3.1 Kesimpulan .....	13
3.2 Saran .....	13
<b>DAFTAR PUSTAKA .....</b>	<b>14</b>

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Perulangan adalah melakukan perintah yang ada di dalam blok perulangan tersebut secara berulang-ulang sesuai dengan nilai yang ditentukan atau sampai mencapai batas yang diinginkan. Dalam bahasa pemrograman JAVA terdapat beberapa statement perulangan yang dapat digunakan salah satunya yaitu “*for*”.

Perulangan dengan teknik ini dikontrol oleh tiga bagian yang ada dalam tanda kurung dan masing-masing bagian ini dipisahkan oleh titik-koma. Pada bagian pertama (inisialisasi ekspresi), sebuah variabel akan di deklarasikan sebagai sebuah titik awal dari perulangan, biasanya variable ini mempunyai tipe data *integer* atau *float*. Sementara pada bagian kedua perulangan diperiksa apakah masih memenuhi syarat atau tidak, jika memenuhi syarat maka statement dibawahnya akan di eksekusi. Sedangkan bagian ketiga adalah bagian dimana jika bagian kedua memenuhi syarat maka nilai variabel ditambahkan sesuai syarat yang dituliskan. Bagian ketiga, ini otomatis akan tidak dibaca oleh program jika kondisi pada bagian ke-dua sudah tidak lagi memenuhi syarat, dan perulangan pun menjadi terhenti.

### **1.2 Tujuan**

1. Menjelaskan cara mengulang kode eksekusi secara efisien
2. Dapat mengontrol alur eksekusi program dengan lebih baik.
3. Mengaplikasikan perulangan *for* dalam bahasa java

### **1.3 Manfaat**

1. Menambah pemahaman tentang struktur perulangan *for* dalam bahasa Java.
2. Menambah wawasan serta bekal mahasiswa mengenai pemrograman

## BAB II

### PEMBAHASAN

#### 2.1 Perulangan Pada Java

- Perulangan  
Adalah struktur kontrol yang digunakan untuk mengulang eksekusi blok kode dengan banyak kondisi tertentu
- Jenis perulangan
  - For : digunakan untuk mengulang eksekusi kode berdasarkan kondisi tertentu
  - While : digunakan untuk mengulang eksekusi kode selama kondisi tertentu terpenuhi
  - Do-while : digunakan untuk mengulang eksekusi kode minimal sekali sebelum kondisi diperiksa

#### 2.2 Langkah pengerjaan

1. Program Perulangan *for* 1
  - a) Pemeriksaan kondisi:  $i \leq 10$ 
    - Variabel lokal  $i$  dibuat dan diberi nilai awal 1
  - b) Pemeriksaan kondisi:  $i \leq 10$ 
    - Program memeriksa apakah nilai  $i$  kurang dari atau sama dengan 10.
    - Jika kondisi benar, program masuk ke badan loop.
    - Jika kondisi salah, program keluar dari loop dan eksekusi selesai.
  - c) Badan loop: `System.out.println(i);`
    - Nilai variabel  $i$  saat ini dicetak ke konsol, diikuti dengan pindah ke baris baru (karena menggunakan `println`).
  - d) Update/Increment:  $i++$ 
    - Setelah badan loop selesai dieksekusi, nilai  $i$  ditambah 1 ( $i = i + 1$ ).
  - e) Pengulangan proses
    - Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
    - Karena nilai awal  $i = 1$  dan setiap iterasi  $i$  bertambah 1, maka loop akan berjalan untuk nilai  $i = 1, 2, 3, \dots, 10$  (total 10 iterasi). Saat  $i$  menjadi 11, kondisi  $i \leq 10$  bernilai false, sehingga loop berhenti dan program selesai

## Kode program

```
1 package pekan5;
2
3 public class PerulanganFor1_2522532029 {
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 }
```

Gambar 2.1

## Output yang dihasilkan

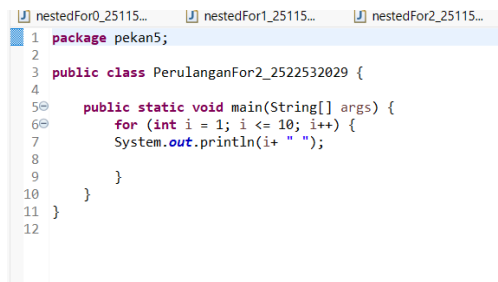
```
<terminated> PerulanganFor1_2522532029 [Java Application] C:\Users\USER\p2\pc
1
2
3
4
5
6
7
8
9
10
```

Gambar 2.2

## 2. Program Perulangan *for* 2

- a) Pemeriksaan kondisi:  $i \leq 10$ 
  - Variabel lokal  $i$  dibuat dan diberi nilai awal 1
- b) Pemeriksaan kondisi:  $i \leq 10$ 
  - Program memeriksa apakah nilai  $i$  kurang dari atau sama dengan 10.
  - Jika kondisi benar, program masuk ke badan loop.
  - Jika kondisi salah, program keluar dari loop dan eksekusi selesai.
- c) Badan loop: `System.out.print(i);`
  - Nilai variabel  $i$  saat ini dicetak ke konsol, tidak diikuti dengan pindah ke baris baru (karena menggunakan `print`).
- d) Update/Increment: `i++`
  - Setelah badan loop selesai dieksekusi, nilai  $i$  ditambah 1 ( $i = i + 1$ ).
- e) Pengulangan proses
  - Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
  - Karena nilai awal  $i = 1$  dan setiap iterasi  $i$  bertambah 1, maka loop akan berjalan untuk nilai  $i = 1, 2, 3, \dots, 10$  (total 10 iterasi). Saat  $i$  menjadi 11, kondisi  $i \leq 10$  bernilai false, sehingga loop berhenti dan program selesai


Kode program:



```
1 package pekan5;
2
3 public class PerulanganFor2_2522532029 {
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
```

Gambar 2.3

Output yang dihasilkan:



```
<terminated> PerulanganFor2_2522532029 [Java Application] C:\Users\USER\AppData\Local\Temp\pluginio
1 2 3 4 5 6 7 8 9 10
```

Gambar 2.4

### 3. Program Perulangan *for* 3

- a) Inisialisasi : `int i = 1`
  - Variabel lokal `i` dibuat dan diberi nilai awal 1
- b) Pemeriksaan kondisi: `i <= 10`
  - Program memeriksa apakah nilai `i` kurang dari atau sama dengan 10.
  - Jika kondisi benar, program masuk ke badan loop.
  - Jika kondisi salah, program keluar dari loop dan eksekusi selesai.
- c) Badan loop: `System.out.print(i);`
  - Nilai variabel `i` saat ini dicetak ke konsol, diikuti dengan pindah ke baris baru (karena menggunakan `println`).
- d) Update/Increment: `i++`
  - Setelah badan loop selesai dieksekusi, nilai `i` ditambah 1 (`i = i + 1`).
- e) Pengulangan proses
  - Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
  - Karena nilai awal `i = 1` dan setiap iterasi `i` bertambah 1, maka loop akan berjalan untuk nilai `i = 1, 2, 3, ..., 10` (total 10 iterasi). Saat `i` menjadi 11, kondisi `i <= 10` bernilai false, sehingga loop berhenti dan program selesai

Kode program:

```
1 package pekan5;
2
3 public class PerulanganFor3_2522532029 {
4     public static void main(String[] args) {
5         int jumlah = 0;
6         for (int i=1; i<=10; i++) {
7             System.out.println(i);
8             jumlah = jumlah + i;
9             if (i<10) {
10                 System.out.print(" + ");
11             }
12         }
13         System.out.println();
14         System.out.println("jumlah "+ jumlah);
15     }
16 }
17 }
```

Gambar 2.5

Output yang dihasilkan:

```
Console X
<terminated> PerulanganFor3_2522532029 [Java Applicat
1
+ 2
+ 3
+ 4
+ 5
+ 6
+ 7
+ 8
+ 9
+ 10
jumlah 55
```

Gambar 2.6

#### 4. Program Perulangan *for* 4

##### a) Inisialisasi : `int i = 1`

- Variabel lokal `i` dibuat dan diberi nilai awal 1

##### b) Pemeriksaan kondisi: `i <= batas`

- Program memeriksa apakah nilai `i` kurang dari atau sama dengan batas(nilai yang diinput user).
- Jika kondisi benar, program masuk ke badan loop.
- Jika kondisi salah, program keluar dari loop dan eksekusi selesai.

##### c) Badan loop: `System.out.print(i);`

- Nilai variabel `i` saat ini dicetak ke konsol, diikuti dengan pindah ke baris baru (karena menggunakan `println`).
- Nilai `i` ditambahkan ke variabel `jumlah`
- Jika `i` kurang dari batas, cetak “ + “
- Jika `i` sama dengan batas, cetak “ = “

##### d) Update/Increment: `i++`

- Setelah badan loop selesai dieksekusi, nilai `i` ditambah 1 (`i = i + 1`).

##### e) Pengulangan proses

- Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
- Karena nilai awal `i = 1` dan setiap iterasi `i` bertambah 1, maka loop akan berjalan untuk nilai `i = 1, 2, 3, ..., 10` (total 10 iterasi). Saat `i` menjadi 11, kondisi `i <= 10` bernilai false, sehingga loop berhenti dan program selesai



Kode program:

```

1 package pekan5;
2
3 import java.util.Scanner;
4
5 public class PerulanganFor4_2511532029 {
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.println("Masukan nilai batas = ");
12        batas = input.nextInt();
13        input.close();
14        for (int i = 1; i<batas; i++) {
15            System.out.println(i);
16            jumlah = jumlah + i ;
17            if (i==batas) {
18                System.out.println(" = ");
19            }
20        }
21        System.out.println(jumlah);
22    }
23 }
24
25
26
27
28
29

```

Gambar 2.7

Output yang dihasilkan:

```

<terminated> PerulanganFor4_2511532029 [Java Application] C:\Users\USER\AppData\Local\Temp\org.ec
Masukan nilai batas =
5
1
+
2
+
3
+
4
+
5
=
15

```

Gambar 2.8

## 5. Program nestedFor0

### 1. Loop luar (Outer Loop)

#### a) Inisialisasi : int line = 1

- Variabel lokal line dibuat dan diberi nilai awal 1

#### b) Pemeriksaan kondisi: line <= 5

- Program memeriksa apakah nilai line kurang dari atau sama dengan 5.
- Jika kondisi benar, program masuk ke badan loop.
- Jika kondisi salah, program keluar dari loop dan eksekusi selesai.

#### c) Badan loop luar

- Berisi loop dalam dan perintah cetak

#### d) Update/Increment: line++

- Setelah badan loop luar selesai dieksekusi, nilai line ditambah 1 (line = line + 1).

#### e) Pengulangan proses loop luar

- Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
- Loop berjalan untuk nilai line = 1 , 2 , 3 , 4 ,5
- Saat Line menjadi 6, kondisi line <= 5 bernilai false, loop luar berhenti

### 2. Loop dalam (Inner Loop)

#### a) Inisialisasi : int j = 1

- Variabel lokal j dibuat dan diberi nilai awal 1

#### b) Pemeriksaan kondisi: j <= (-1 \* line + 5)

- Program memeriksa apakah nilai j kurang dari atau sama dengan  $(-1 * \text{line} + 5)$ .
  - Jika kondisi benar, program masuk ke badan loop.
  - Jika kondisi salah, program keluar dari loop dan eksekusi selesai.
- c) Badan loop dalam
- Berisi String “.” Tanpa pindah garis
- d) Update/Increment:  $j++$
- Setelah badan loop luar selesai dieksekusi, nilai j ditambah 1 ( $j = j + 1$ ).
- e) Pengulangan proses loop luar
- Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
  - Jumlah literasi dalam loop dalam bergantung pada nilai line :
    - Line = 1:  $j \leq 4$  (4 literasi)
    - Line = 2:  $j \leq 3$  (3 literasi)
    - Line = 3:  $j \leq 2$  (2 literasi)
    - Line = 4:  $j \leq 1$  (1 literasi)
    - Line = 5:  $j \leq 0$  (0 literasi)

### 3. Perintah setelah Loop dalam

- a) `System.out.print(line);`
- Mencetak nilai line saat ini tanpa pindah baris
- b) `System.out.println( );`
- Pindah ke baris baru setelah menyelesaikan satu baris pola

Kode program :

```

1 package pekan5;
2
3 public class nestedFor0_2511532029 {
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

```

Gambar 2.9

Output yang dihasilkan :

```

<terminated> nestedFor0_2511532029 [Java Application] C:\Users\U
. . . . 1
. . . 2
. . 3
. 4
5

```

Gambar 2.10

## 6. Program nestedFor1

### 1. Loop luar (Outer Loop)

- a) Inisialisasi : `int i = 1`
  - Variabel lokal `i` dibuat dan diberi nilai awal 1
- b) Pemeriksaan kondisi: `i <= 5`
  - Program memeriksa apakah nilai `i` kurang dari atau sama dengan 5.
  - Jika kondisi benar, program masuk ke badan loop.
  - Jika kondisi salah, program keluar dari loop dan eksekusi selesai.
- c) Badan loop luar
  - Berisi loop dalam dan perintah cetak
- d) Update/Increment: `i++`
  - Setelah badan loop luar selesai dieksekusi, nilai `i` ditambah 1 (`i = i + 1`).
- e) Pengulangan proses loop luar
  - Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
  - Loop berjalan untuk nilai `i = 1, 2, 3, 4, 5`
  - Saat `i` menjadi 6, kondisi `i <= 5` bernilai false, loop luar berhenti

### 2. Loop dalam (inner Loop)

- a) Inisialisasi : `int j = 1`
  - Variabel lokal `j` dibuat dan diberi nilai awal 1
- b) Pemeriksaan kondisi: `j <= 5`
  - Program memeriksa apakah nilai `j` kurang dari atau sama dengan 5.
  - Jika kondisi benar, program masuk ke badan loop.
  - Jika kondisi salah, program keluar dari loop dan eksekusi selesai.
- c) Badan loop dalam
  - Mencetak karakter “ \* ” Tanpa pindah garis
- d) Update/Increment: `j++`
  - Setelah badan loop luar selesai dieksekusi, nilai `j` ditambah 1 (`j = j + 1`).
- e) Pengulangan proses loop luar
  - Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
  - Saat `j` menjadi 6, kondisi `j <= 5` bernilai false, loop dalam berhenti

### 3. Perintah setelah Loop dalam

- a) `System.out.print();`
  - Pindah ke garis baru setelah menyelesaikan satu baris bintang

Kode program :

```
1 package pekan5;
2
3 public class nestedFor1_2511532029 {
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 }
```

Gambar 2.11

Output yang dihasilkan :

```
<terminated> nestedFor1_2511532029 [Java Application] C:\Users\USER\p2\pool\g
*****
*****
*****
*****
*****
```

Gambar 2.12

## 7. Program nestedFor2

### 1. Loop luar (Outer Loop)

- a) Inisialisasi : int i = 0
  - Variabel lokal i dibuat dan diberi nilai awal 0
- b) Pemeriksaan kondisi: i <= 5
  - Program memeriksa apakah nilai i kurang dari atau sama dengan 5.
  - Jika kondisi benar, program masuk ke badan loop.
  - Jika kondisi salah, program keluar dari loop dan eksekusi selesai.
- c) Badan loop luar
  - Berisi loop dalam dan perintah cetak
- d) Update/Increment: i++
  - Setelah badan loop luar selesai dieksekusi, nilai i ditambah 1 (i = i + 1).
- e) Pengulangan proses loop luar
  - Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
  - Loop berjalan untuk nilai i = 0, 1, 2, 3, 4, 5
  - Saat i menjadi 6, kondisi line <= 5 bernilai false, loop luar berhenti

### 2. Loop dalam (inner Loop)

- a) Inisialisasi : int j = 0
  - Variabel lokal j dibuat dan diberi nilai awal 1
- b) Pemeriksaan kondisi: j <= 5
  - Program memeriksa apakah nilai i kurang dari atau sama dengan 5.
  - Jika kondisi benar, program masuk ke badan loop.
  - Jika kondisi salah, program keluar dari loop dan eksekusi selesai.
- c) Badan loop dalam

- Mencetak hasil penjumlahan  $i + j$  diikuti dengan spasi , tanpa pindah baris
  - d) Update/Increment:  $j++$ 
    - Setelah badan loop luar selesai dieksekusi, nilai  $j$  ditambah 1 ( $j = j + 1$ ).
  - e) Pengulangan proses loop luar
    - Program kembali ke langkah 2 (pemeriksaan kondisi) dan mengulangi langkah 2–4 terus-menerus.
    - Loop dalam berjalan untuk nilai  $j = 0, 1, 2, 3, 4, 5$
    - Saat  $j$  menjadi 6, kondisi  $j \leq 5$  bernilai false, loop dalam berhenti
3. Perintah setelah Loop dalam
- b) `System.out.print();`
- Pindah ke garis baru setelah menyelesaikan satu baris penjumlahanj

Kode program :

```

1 package pekan5;
2
3 public class nestedFor2_2511532029 {
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 }

```

Gambar 2.13

```

<terminated> nestedFor2_2511532029 [Java Application] C:\Users\USER\p2\pool\plugins\org.eclipse.j
0 1 2 3 4 5
1 2 3 4 5 6
2 3 4 5 6 7
3 4 5 6 7 8
4 5 6 7 8 9
5 6 7 8 9 10

```

Gambar 2.14

## **BAB III**

### **KESIMPULAN**

#### **3.1 Kesimpulan**

Berdasarkan program-program praktikum yang dibuat, dapat disimpulkan bahwa praktikum ini berhasil mengajarkan konsep fundamental perulangan for dalam Java secara komprehensif, dimulai dari penguasaan dasar perulangan dengan perbedaan output menggunakan `println()` untuk tampilan vertikal dan `print()` untuk tampilan horizontal, hingga implementasi teknik akumulator untuk perhitungan beruntun.

Lebih lanjut, praktikum ini telah mengembangkan pemahaman nested loop melalui pembuatan berbagai pola seperti persegi dengan karakter '\*' dan segitiga terbalik dengan kombinasi titik-angka, serta demonstrasi operasi matematika dalam loop bersarang. Melalui manipulasi kondisi dalam iterasi dan formatting output, praktikum ini tidak hanya membangun logika pemrograman yang solid tetapi juga melatih kemampuan pattern recognition dan variable manipulation, yang secara keseluruhan membentuk fondasi esensial dalam penyelesaian masalah pemrograman yang lebih kompleks.

#### **3.2 Saran**

1. Akan lebih baik bila dosen menyelenggarakan sesi pra-praktikum di kelas agar mahasiswa dapat memperoleh pemahaman awal yang lebih memadai, sehingga dapat menghindari kepanikan atau kesalahan saat praktikum
2. Sebaiknya dosen membagikan materi praktikum terlebih dahulu melalui iLearn agar mahasiswa bisa mempersiapkan diri sebelum pelaksanaan praktikum

## **DAFTAR PUSTAKA**

[1] “The for Statement (The Java™ Tutorials > Learning the Java Language)”, Oracle, 2024. Available:

<https://docs.oracle.com/javase/tutorial/java/nutsandbolts/for.html>