

**LAPORAN PRATIUM**  
**PEMROGRAMAN ALGORITMA PEMROGRAMAN**  
**PERULANGAN**

**Disusun Oleh:**

Gina Ramadhani

Nim: 2511533014

Dosen Pengampu: DR. Wahyudi, S.T, M.T

Asisten Pratikum: Rahmad Dwi Rizki Olders



**DAPERTEMEN INFORMATIKA**  
**FAKULTAS TEKNOLOGI INFORMASI**  
**UNIVERSITAS ANDALAS**  
**TAHUN 2025**

## **KATA PENGANTAR**

Laporan praktikum ini dibuat sebagai bentuk pertanggung jawaban atas kegiatan praktikum mata kuliah Algoritma Pemrograman, khususnya mengenai Perulangan dalam Bahasa Pemrograman Java. Melalui penyusunan laporan ini, penulis dapat memperdalam pemahaman terhadap materi praktikum sekaligus melatih ketelitian, keteraturan, serta kemampuan menulis sesuai dengan kaidah akademik dasar. Dengan demikian, laporan ini diharapkan tidak hanya menjadi dokumentasi kegiatan praktikum, tetapi juga dapat berfungsi sebagai sarana pembelajaran dan latihan keterampilan menulis ilmiah yang bermanfaat untuk proses belajar berikutnya.

Penulis menyadari bahwa laporan ini masih memiliki banyak kekurangan. Oleh sebab itu, penulis sangat mengharapkan adanya kritik dan saran yang bersifat membangun agar kualitas laporan dan pemahaman penulis dapat lebih baik pada kesempatan selanjutnya.

Padang, 2025

Penulis

## DAFTAR ISI

<b>KATA PENGANTAR.....</b>	<b>i</b>
<b>DAFTAR ISI.....</b>	<b>ii</b>
<b>BAB I.....</b>	<b>1</b>
<b>PENDAHULUAN .....</b>	<b>1</b>
1.1 Latar Belakang .....	1
1.2 Tujuan Praktikum.....	1
1.3 Manfaat Praktikum.....	2
<b>BAB II .....</b>	<b>3</b>
<b>PEMBAHASAN .....</b>	<b>3</b>
2.1 Praktikum “nestedFor0”.....	3
2.2 Praktikum “nestedFor1”.....	5
2.3 Praktikum “nestedFor2”.....	7
2.4 Praktikum “PerulanganFor1” .....	10
2.5 Praktikum “PerulanganFor2”.....	12
2.6 Praktikum “PerulanganFor3” .....	13
2.7 Praktikum “PerulanganFor4” .....	16
<b>BAB III.....</b>	<b>19</b>
<b>KESIMPULAN .....</b>	<b>19</b>
3.1 Kesimpulan .....	19
<b>DAFTAR PUSTAKA .....</b>	<b>20</b>

## **BAB I**

### **PENDAHULUAN**

#### **1.1 Latar Belakang**

Perulangan, atau yang sering disebut looping, dalam pemrograman adalah sebuah proses di mana suatu perintah atau blok kode dapat dijalankan secara terus-menerus berdasarkan kondisi yang sudah ditentukan. Dalam bahasa pemrograman Java, perulangan memiliki peranan penting karena dapat membantu programmer menjalankan instruksi yang sama berulang kali tanpa perlu menuliskan kode secara berulang. Hal ini sangat berguna untuk membuat program menjadi lebih sederhana serta efisien dalam penggunaan waktu dan memori.

Java memiliki tiga jenis perulangan utama, yaitu `for`, `while`, dan `do-while`. Perulangan `for` biasanya diaplikasikan ketika jumlah iterasi sudah diketahui sebelumnya, sedangkan `while` dan `do-while` digunakan saat jumlah iterasi tidak pasti dan bergantung pada kondisi tertentu. Dengan memahami perulangan ini, programmer dapat menyelesaikan berbagai masalah pemrograman yang memerlukan pelaksanaan kode secara berulang, seperti pengolahan data dan pengambilan input berulang.

Praktikum ini diselenggarakan untuk memberi kesempatan belajar secara praktis tentang konsep, jenis, serta penerapan perulangan dalam Java, sehingga peserta dapat menguasai teknik dasar looping dan memperbaiki keterampilan pemrograman mereka secara langsung lewat latihan coding.

#### **1.2 Tujuan Praktikum**

1. Memahami konsep dasar perulangan (looping) dalam bahasa pemrograman Java.
2. Mengenal dan menguasai berbagai jenis perulangan seperti `for`, `while`, dan `do-while`.
3. Mampu mengimplementasikan perulangan dalam program Java secara tepat sesuai kebutuhan.
4. Memahami cara kerja logika perulangan dan mengontrol agar perulangan berhenti pada kondisi yang tepat.
5. Mengembangkan kemampuan pemrograman melalui latihan langsung penggunaan perulangan dalam penulisan kode.

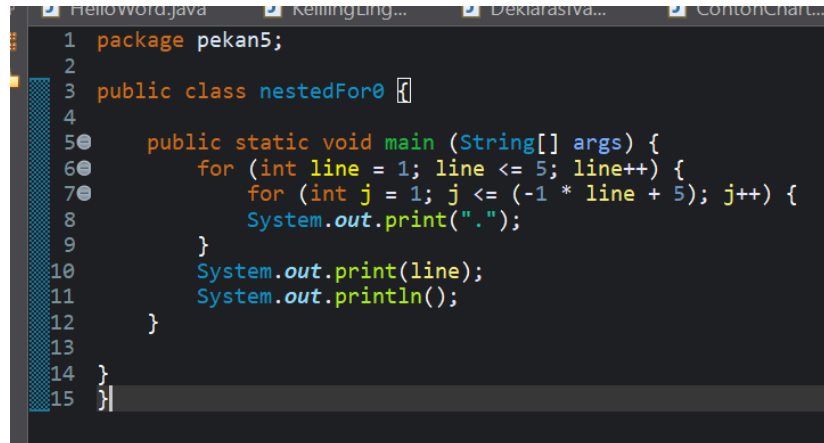
### **1.3 Manfaat Praktikum**

1. Membantu peserta memahami konsep dasar dan mekanisme perulangan yang digunakan dalam pemrograman Java.
2. Meningkatkan kemampuan peserta dalam menulis dan mengimplementasikan
3. Memperkuat keterampilan logika pemrograman, terutama dalam mengatur kondisi pengulangan agar proses program berjalan sesuai harapan tanpa mengalami loop tak berujung.
4. Memberikan pengalaman praktis yang dapat membantu peserta mengatasi masalah pengulangan pada pengolahan data dan pengambilan input secara berulang.
5. Mempersiapkan peserta untuk membuat program yang lebih kompleks dan optimal dengan memanfaatkan teknik perulangan sebagai bagian dari algoritma pemrograman.

## BAB II

### PEMBAHASAN

#### 2.1 Praktikum “nestedFor0”



```
1 package pekan5;
2
3 public class nestedFor0 {
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 }
```

**Gambar 1.1** Kode Program Praktikum nestedFor0

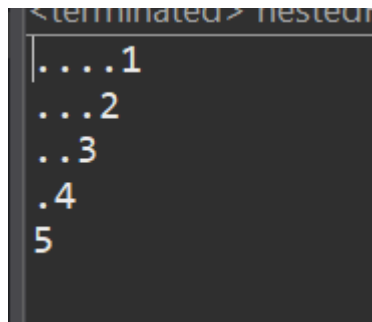
Program ini menggunakan nested for loop untuk mencetak pola teks ke layar. Loop luar berjalan dari 1 sampai 5, dan untuk setiap nilai dari loop luar, loop dalam mencetak karakter titik (".") sebanyak jumlah yang berkurang setiap barisnya. Setelah mencetak titik-titik, program mencetak nomor baris tersebut dan pindah ke baris baru. Hasilnya adalah pola yang terdiri dari titik yang jumlahnya menurun diikuti angka baris dari 1 hingga 5.

Berikut penjelasan langkah-langkah kerjanya:

1. Baris 1: Mendeklarasikan bahwa kelas ini berada dalam package bernama pekan5
2. Baris 3: Mendefinisikan kelas publik dengan nama nestedFor0.
3. Baris 5: Mendefinisikan metode main sebagai titik awal eksekusi program.
4. Baris 6: Membuat loop luar yang menginisialisasi variabel line mulai dari 1, berjalan hingga line kurang dari atau sama dengan 5, dan menambah line sebanyak 1 setiap iterasi.

5. Baris 7: Membuat loop dalam yang menginisialisasi variabel *j* mulai dari 1, berjalan hingga *j* kurang dari atau sama dengan hasil perhitungan  $(-1 * \text{line} + 5)$ , dan menambah *j* sebanyak 1 setiap iterasi. Ini mengatur jumlah titik yang dicetak menurun setiap baris.
6. Baris 8: Mencetak karakter titik . di baris yang sama tanpa pindah ke baris baru.
7. Baris 9: Menutup blok loop dalam.
8. Baris 10: Mencetak nilai line pada baris yang sama setelah titik-titik.
9. Baris 11: Mencetak baris baru, sehingga output berikutnya tampil di baris selanjutnya.

Dari Langkah-langkah diatas kita akan mendapatkan output:



```
<terminated> nested
....1
...2
..3
.4
5
```

**Gambar 2.2** Output dari kode program nestedFor0

### 2.1.1 Analisis

- Program berjalan dengan benar dan menampilkan output sesuai logika perulangan bersarang (*nested for*).
- Tidak error dalam eksekusi
- Output yang dihasilkan sesuai dengan logika program, yaitu pola menurun dari ....1 hingga 5.
- Secara keseluruhan, program telah berfungsi dengan benar dan efisien, mencetak pola berbentuk segitiga terbalik dari titik dan angka tanpa kesalahan.

## 2.2 Praktikum “nestedFor1”

```

1 package pekan5;
2
3 public class nestedFor1 {
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

```

**Gambar 2.3** Kode program pratikum nestedFor1

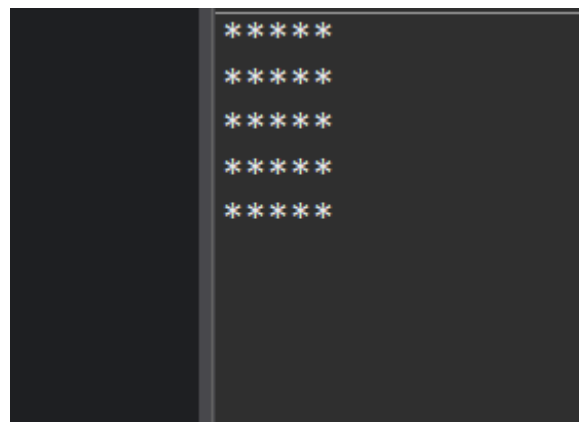
Kode program tersebut bertujuan untuk menampilkan pola bintang berbentuk persegi berukuran 5x5 di layar. Program menggunakan dua perulangan bersarang (*nested for*), di mana perulangan luar mengatur jumlah baris, dan perulangan dalam mencetak bintang (\*) sebanyak lima kali di setiap baris. Setelah satu baris penuh tercetak, program berpindah ke baris berikutnya menggunakan `System.out.println()`. Hasil akhirnya adalah tampilan lima baris bintang yang masing-masing terdiri dari lima simbol \*, membentuk pola persegi.

Berikut penjelasan langkah-langkah kerjanya:

1. Perulangan luar `for (int i = 1; i <= 5; i++)` berfungsi untuk mengatur jumlah baris yang akan dicetak, yaitu sebanyak 5 baris.
2. Di dalam perulangan luar terdapat perulangan dalam `for (int j = 1; j <= 5; j++)` yang bertugas mencetak karakter bintang (\*) sebanyak 5 kali dalam setiap baris.

3. Setelah perulangan dalam selesai (artinya satu baris bintang telah tercetak), program menjalankan perintah `System.out.println();` untuk pindah ke baris baru.
4. Langkah-langkah tersebut berulang hingga perulangan luar selesai dijalankan seluruhnya.
5. Hasil akhir dari program adalah pola bintang berbentuk kotak 5x5.

Dari Langkah-langkah diatas kita akan mendapatkan output:



**Gambar 2.4** Output dari kode program `nestedFor1`

### 2.1.2 Analisis

- Program berjalan dengan benar dan menampilkan output sesuai dengan logika perulangan bersarang (*nested for*).
- Tidak ada error selama proses kompilasi maupun eksekusi program.
- Program tidak memerlukan input pengguna, karena nilai batas perulangan sudah ditentukan langsung di dalam kode ( $i \leq 5$  dan  $j \leq 5$ ).
- Perulangan luar berfungsi untuk mengatur jumlah baris, sedangkan perulangan dalam mencetak bintang (\*) sebanyak lima kali di setiap baris.
- Setelah satu baris selesai, program berpindah ke

baris baru menggunakan `System.out.println()`;

- Output yang dihasilkan sesuai logika, yaitu pola bintang berbentuk persegi 5x5.

### 2.3 Praktikum “nestedFor2”

```

1 package pekan5;
2
3 public class nestedFor2 {
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 }
```

**Gambar 2.5** Kode program praktikum nestedFor2

Program ini menggunakan perulangan bersarang (nested loop) untuk menampilkan hasil penjumlahan antara dua variabel, yaitu *i* dan *j*. Variabel *i* mewakili baris, sedangkan *j* mewakili kolom. Di setiap iterasi, program menjumlahkan nilai *i* dan *j*, lalu mencetak hasilnya di layar dengan spasi sebagai pemisah. Setelah satu baris selesai dicetak, program membuat baris baru menggunakan `System.out.println()`. Hasil akhirnya berupa tampilan berbentuk tabel angka yang merupakan hasil penjumlahan indeks baris dan kolom.

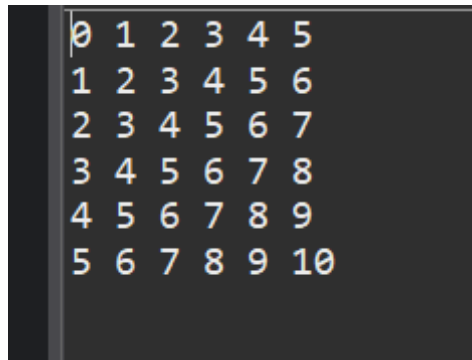
Berikut penjelasan langkah-langkah kerjanya:

1. `package pekan5;`

- Menunjukkan bahwa file Java ini berada dalam package (folder) bernama *pekan5*. Package digunakan untuk mengelompokkan kelas-kelas agar lebih terstruktur.
2. `public class nestedFor2 {`
    - Mendefinisikan sebuah kelas publik bernama `nestedFor2`. Semua kode program utama akan ditulis di dalam kelas ini.
  3. `public static void main(String[] args) {`
    - Ini adalah *method utama* dalam program Java. Program selalu dimulai dari baris ini saat dijalankan.
  4. `for (int i = 0; i <= 5; i++) {`
    - Loop pertama (luar) digunakan untuk mengatur jumlah baris.
    - Nilai `i` dimulai dari 0 dan akan bertambah 1 setiap iterasi hingga mencapai 5.
  5. `for (int j = 0; j <= 5; j++) {`
    - Loop kedua (dalam) digunakan untuk mengatur jumlah kolom.
    - Nilai `j` dimulai dari 0 hingga 5 pada setiap iterasi `i`.
  6. `System.out.print(i + j + " ");`
    - Pada setiap kombinasi `i` dan `j`, program menjumlahkan keduanya (`i + j`).
    - Hasilnya dicetak di layar diikuti dengan spasi agar hasilnya rapi berjajar.
  7. `}`

- Menandai akhir dari loop dalam (loop j).
- 8. `System.out.println();`
  - Setelah satu baris (loop j) selesai, perintah ini membuat baris baru.
  - Tujuannya agar hasil dari iterasi i berikutnya ditulis di baris baru.
- 9. `}`
  - Menandai akhir dari loop luar (loop i).
- 10. `}` (terakhir)
  - Menandai akhir dari kelas `nestedFor2`.

Dari Langkah-langkah diatas kita akan mendapatkan output:



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 2.6** Output kode program `nestedFor2`

### 2.1.3 Analisis

- Program berjalan dengan baik tanpa error dan menampilkan output sesuai logika nested loop.
- Perulangan luar (i) mengatur baris dan perulangan dalam (j) mengatur kolom.
- Setiap nilai hasil dari  $i + j$  membentuk pola tabel 6x6 dengan angka berurutan.
- Kode efisien, mudah dibaca, dan tidak

memerlukan input karena batas perulangan ditentukan statis.

## 2.4 Praktikum “PerulanganFor1”

```
1 package pekan5;  
2  
3 public class PerulanganFor1 {  
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 2.7** Kode program praktikum PerulanganFor1

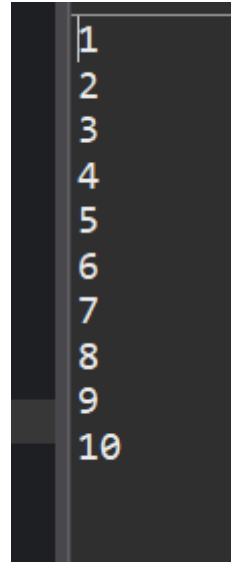
Program ini berfungsi untuk menampilkan angka dari 1 sampai 10 di layar secara berurutan ke bawah. Hal tersebut dilakukan dengan menggunakan perulangan for, di mana variabel *i* berfungsi sebagai penghitung (counter) yang dimulai dari 1 dan akan terus bertambah satu per satu hingga mencapai 10. Pada setiap iterasi, nilai *i* dicetak menggunakan `System.out.println(i)`, sehingga setiap angka muncul di baris baru. Program ini sederhana namun penting untuk memahami konsep dasar looping dalam Java.

Berikut penjelasan langkah-langkah kerjanya:

1. Program dimulai dari bagian utama saat dijalankan.
2. Variabel *i* diberi nilai awal 1.
3. Perulangan for berjalan selama nilai *i* kurang dari atau sama dengan 10.
4. Setiap kali perulangan, nilai *i* ditampilkan ke layar menggunakan `System.out.println(i)`.

5. Setelah dicetak, nilai *i* bertambah 1.
6. Perulangan berhenti ketika nilai *i* melebihi 10 dan program selesai dijalankan.

Dari Langkah-langkah diatas kita akan mendapatkan output:



**Gambar 2.8** Output kode program PerulanganFor1

#### 2.1.4 Analisis

- Gambar menampilkan hasil output berupa angka 1 sampai 10.
- Setiap angka dicetak pada baris baru secara berurutan ke bawah.
- Output dihasilkan dari perulangan for yang menambah nilai *i* dari 1 hingga 10.
- Program berjalan dengan benar tanpa error.
- Menunjukkan penggunaan perulangan sederhana untuk mencetak deret angka berurutan.

## 2.5 Praktikum “PerulanganFor2”

```
1 package pekan5;  
2  
3 public class perulanganFor2 {  
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
```

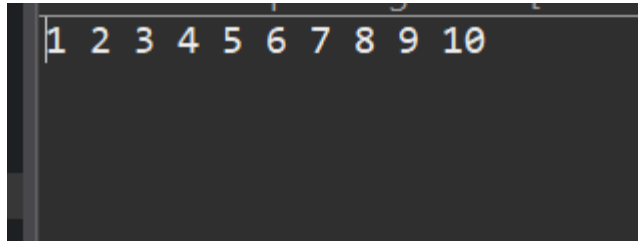
**Gambar 2.9** Kode program praktikum PerulanganFor2

Program ini berfungsi untuk menampilkan angka 1 sampai 10 dalam satu baris secara berurutan. Proses ini dilakukan dengan menggunakan perulangan for yang mengatur agar variabel *i* dimulai dari 1 dan terus bertambah satu hingga mencapai 10. Setiap nilai *i* dicetak menggunakan `System.out.print(i + " ")`, yang berarti setiap angka akan ditampilkan dalam satu baris dengan spasi sebagai pemisah, bukan baris baru. Program ini merupakan contoh sederhana untuk memahami penggunaan perulangan dan fungsi print tanpa pindah baris.

Berikut penjelasan langkah-langkah kerjanya:

1. Program dimulai dari bagian utama saat dijalankan.
2. Variabel *i* diberi nilai awal 1.
3. Perulangan for berjalan selama *i* kurang dari atau sama dengan 10.
4. Setiap iterasi mencetak nilai *i* dalam satu baris dengan spasi menggunakan `System.out.print(i + " ")`.
5. Setelah mencapai nilai 10, perulangan berhenti dan program selesai dijalankan.

Dari Langkah-langkah diatas kita akan mendapatkan output:



**Gambar 2.10** Output kode program PerulanganFor2

### 2.1.5 Analisis

- Program menggunakan perulangan for untuk mencetak angka dari 1 hingga 10 secara berurutan.
- Tidak terdapat kesalahan sintaks dan program berjalan
- Logika perulangan sederhana dan efisien untuk menampilkan deret angka berurutan.
- Struktur kode jelas, mudah dibaca, serta sesuai dengan prinsip dasar penggunaan perulangan dalam Java.

## 2.6 Praktikum “PerulanganFor3”

```

1 package pekan5;
2
3 public class perulanganFor3 {
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 }
20

```

**Gambar 2.11** Kode program praktikum PerulanganFor3

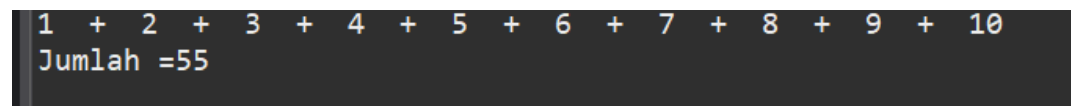
Program di atas berfungsi untuk menampilkan deret angka dari 1 sampai 10 dan menghitung jumlah total dari angka-angka tersebut. Variabel *jumlah* digunakan untuk menyimpan hasil penjumlahan seluruh angka dalam perulangan. Struktur `for` digunakan untuk mengulangi proses dari angka 1 hingga 10. Di dalam perulangan, program mencetak nilai *i* dan menambahkannya ke variabel *jumlah*. Kondisi `if (i < 10)` digunakan agar tanda “+” hanya muncul di antara angka, bukan setelah angka terakhir. Setelah perulangan selesai, program mencetak hasil total penjumlahan dengan menampilkan teks “Jumlah =” diikuti nilai totalnya. Program ini berjalan dengan benar dan efisien, menampilkan hasil yang rapi di layar.

Berikut penjelasan langkah-langkah kerjanya:

1. `package pekan5;` → Menentukan bahwa program berada dalam paket bernama *pekan5*.
2. `public class perulanganFor3 {` → Mendefinisikan kelas utama dengan nama *perulanganFor3*.
3. `public static void main(String[] args) {` → Menentukan fungsi utama tempat program mulai dieksekusi.
4. `int jumlah = 0;` → Mendeklarasikan variabel *jumlah* untuk menyimpan total penjumlahan dan menginisialisasinya dengan 0.
5. `for (int i = 1; i <= 10; i++) {` → Membuat perulangan dari *i* = 1 hingga *i* = 10.
6. `System.out.print(i);` → Mencetak nilai *i* ke layar tanpa pindah baris.
7. `jumlah = jumlah + i;` → Menambahkan nilai *i* ke dalam variabel *jumlah*.
8. `if (i < 10) {` → Mengecek apakah *i* belum mencapai angka 10.

9. `System.out.print(" + ");` → Jika kondisi benar, menampilkan tanda “+” di antara angka.
10. `System.out.println();` → Setelah perulangan selesai, berpindah ke baris baru.
11. `System.out.println("Jumlah = " + jumlah);` → Menampilkan hasil total penjumlahan dari 1 sampai 10.
12. `}` → Menutup blok kode main dan kelas.

Dari Langkah-langkah diatas kita akan mendapatkan output:



```
1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10
Jumlah =55
```

**Gambar 2.12** Output kode program PerulanganFor3

#### 2.1.6 Analisis.

- Program menggunakan perulangan *for* untuk menampilkan deret angka dari 1 sampai 10.
- Di dalam perulangan, setiap angka dijumlahkan ke dalam variabel jumlah untuk menghitung total keseluruhan.
- Kondisi `if (i < 10)` digunakan agar tanda “+” hanya muncul di antara angka dan tidak setelah angka terakhir.
- Hasil penjumlahan ditampilkan setelah perulangan selesai menggunakan `System.out.println("Jumlah =" + jumlah);`.
- Program berjalan dengan benar dan efisien dalam menampilkan deret angka serta menghitung totalnya.
- Struktur logikanya jelas, mudah dipahami, dan sesuai dengan prinsip dasar penggunaan perulangan serta kondisi dalam Java.

## 2.7 Praktikum “PerulanganFor4”

```

1 package pekan5;
2
3 import java.util.Scanner;
4
5 public class PerulanganFor4 {
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

```

**Gambar 2.13** Kode program praktikum PerulanganFor4

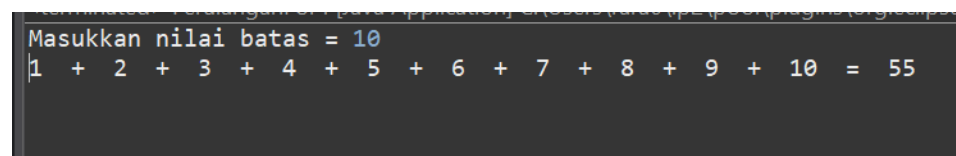
Program di atas berfungsi untuk menampilkan deret angka dari 1 hingga batas yang dimasukkan oleh pengguna, serta menghitung total penjumlahan dari angka-angka tersebut. Program memanfaatkan perulangan *for* dan struktur *if-else* untuk mengatur tampilan tanda “+” di antara angka dan tanda “=” di akhir sebelum hasil jumlah ditampilkan. Program ini juga menggunakan *Scanner* agar pengguna bisa menentukan sendiri nilai batas angka yang akan dijumlahkan.

Berikut penjelasan langkah-langkah kerjanya:

1. Mendeklarasikan variabel *jumlah* untuk menyimpan total penjumlahan dan variabel *batas* untuk menyimpan angka batas dari input pengguna.
2. Membuat objek *Scanner* untuk membaca input dari keyboard.
3. Menampilkan pesan “Masukkan nilai batas = ” di layar.

4. Menerima input angka dari pengguna dan menyimpannya ke dalam variabel batas.
5. Menutup objek Scanner setelah input selesai dibaca.
6. Memulai perulangan *for* dari  $i = 1$  hingga  $i \leq$  batas.
7. Mencetak nilai  $i$  di setiap perulangan.
8. Menambahkan nilai  $i$  ke dalam variabel jumlah untuk menghitung total.
9. Menggunakan *if* untuk memeriksa apakah  $i$  masih lebih kecil dari batas:
  - Jika ya, mencetak tanda “+”.
  - Jika tidak, mencetak tanda “=”.
10. Setelah perulangan selesai, mencetak total hasil penjumlahan (jumlah).

Dari Langkah-langkah diatas kita akan mendapatkan output:



```

Masukkan nilai batas = 10
1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = 55
  
```

**Gambar 2.14** Output kode program PerulanganFor4

### 2.1.7 Analisis

- Program meminta praktikan untuk memasukkan nilai batas sebagai akhir dari deret bilangan yang akan dijumlahkan.
- Menggunakan perulangan (looping) untuk menampilkan setiap angka dari 1 hingga nilai batas yang dimasukkan.
- Setiap angka ditampilkan diikuti tanda “+” kecuali angka terakhir, agar format output terlihat rapi seperti deret matematika.
- Program juga melakukan penjumlahan total seluruh angka di dalam perulangan dan menampilkan hasilnya di akhir.

- Logika perulangan berjalan urutan dari 1 sampai  $n$ , sehingga setiap iterasi menambah nilai ke total secara berurutan.
- Struktur logika sederhana, mudah dipahami dan efisien, karena hanya menggunakan satu perulangan dan operasi penjumlahan dasar.
- Output yang dihasilkan menunjukkan urutan bilangan dan hasil akhir penjumlahan dengan benar, sesuai logika yang diharapkan.

## BAB III

### KESIMPULAN

#### 3.1 Kesimpulan

Kesimpulan dari pembahasan mengenai perulangan *for* dan *nested for* dalam Java adalah bahwa kedua struktur ini berfungsi untuk mengulang eksekusi perintah secara otomatis berdasarkan kondisi tertentu. Perulangan *for* digunakan jika jumlah pengulangan sudah diketahui, karena memiliki bagian inisialisasi, kondisi, dan perubahan nilai yang jelas. Dengan cara ini, program dapat berjalan lebih efisien dan terkontrol tanpa perlu menulis perintah berulang secara manual.

Perulangan bersarang atau *nested for* digunakan saat dibutuhkan pengulangan di dalam pengulangan lain, misalnya untuk membuat pola angka, tabel, atau bentuk tertentu. Struktur ini bekerja dengan menjalankan satu perulangan di dalam perulangan lainnya, di mana perulangan luar mengatur baris dan perulangan dalam mengatur kolom. Penggunaan *nested loop* membuat program mampu menampilkan data secara terstruktur dan teratur sesuai kebutuhan.

Secara keseluruhan, perulangan *for* dan *nested for* merupakan dasar penting dalam logika pemrograman yang sering digunakan dalam berbagai aplikasi. Dengan memahami konsep dan cara kerjanya, seseorang dapat menulis program yang lebih rapi, efisien, serta mudah dipahami. Penguasaan materi ini juga membantu dalam membangun algoritma yang sistematis dan efektif untuk menyelesaikan berbagai permasalahan komputasi.

**DAFTAR PUSTAKA**

- [1] A. Nugroho, Pemrograman Java untuk Pemula. Bandung: Informatika, 2019
- [2] P. J. Deitel and H. M. Deitel, Java: How to Program, 10th ed. Pearson Education, 2017
- [3] Oracle, “The Java Tutorials,” [Online]. Available: <https://docs.oracle.com/javase/tutorial/> .[Diakses: 2-Nov-2025]