

LAPORAN PRATIKUM
ALGORITMA DAN PEMROGRAMAN

“ Perulangan Menggunakan *for loop* ”

disusun Oleh:

Muhammad Yasin Habiburrahman

2511532016

Dosen Pengampu:

Dr. Wahyudi. S.T.M.T

Asisten Pratikum:

Muhammad Zaki Al Hafiz



DEPARTEMEN INFORMATIKA FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ANDALAS
TAHUN 2025

KATA PENGANTAR

Puji dan syukur kami panjatkan ke hadirat Allah SWT atas segala rahmat dan karunia-Nya, sehingga kami dapat menyelesaikan tugas kelompok ini dengan baik. Shalawat serta salam semoga senantiasa tercurah kepada junjungan kita Nabi Muhammad SAW, keluarga, sahabat, serta para pengikutnya hingga akhir zaman.

Tugas kelompok ini kami susun dengan tujuan untuk memenuhi salah satu tugas mata kuliah Algoritma dan Pemrograman, dengan tema “Perulangan Menggunakan *for loop*”, yang mencakup bagaimana jalan dari *for loop* dan *nested-for loops*

Kami menyadari bahwa penyusunan tugas ini masih jauh dari sempurna, baik dari segi materi maupun penyajiannya. Oleh karena itu, kami sangat mengharapkan kritik dan saran yang membangun dari semua pihak demi kesempurnaan tugas kami di masa mendatang.

Akhir kata, kami mengucapkan terima kasih kepada dosen pembimbing yang telah memberikan arahan, serta kepada seluruh anggota kelompok yang telah bekerja sama dengan baik sehingga tugas ini dapat terselesaikan tepat waktu. Semoga tugas ini dapat bermanfaat bagi kami khususnya, dan bagi pembaca pada umumnya.

Padang, 27 Oktober 2025

Muhammad Yasin Habiburrahman

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI	ii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Tujuan.....	1
1.3 Manfaat Praktikum.....	1
BAB II PEMBAHASAN.....	3
2.1 Perulangan <i>for</i> (<i>for looping</i>) Sederhana	3
2.2 Perulangan <i>for</i> dengan Operasi Penjumlahan	4
2.3 Perulangan <i>for</i> dengan Menghitung Angka dari 1 Sampai n	5
2.4 Perulangan <i>for</i> di dalam Perulangan <i>for</i> lainnya (<i>nested for</i>)	6
BAB III KESIMPULAN.....	8
DAFTAR PUSTAKA.....	9

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam pemrograman, perulangan (*looping*) merupakan salah satu konsep dasar yang sangat penting karena memungkinkan suatu proses dijalankan berulang kali tanpa perlu menulis kode yang sama secara berulang. Salah satu bentuk perulangan yang umum digunakan dalam bahasa Java adalah *for loop*. Struktur *for loop* memudahkan programmer untuk mengontrol jumlah iterasi dengan menentukan nilai awal, kondisi, dan perubahan variabel secara jelas dalam satu baris.

Selain *for loop* tunggal, terdapat juga *nested for loop* atau perulangan bersarang, yaitu perulangan di dalam perulangan lain. Konsep ini sering digunakan untuk menyelesaikan masalah yang melibatkan struktur berulang dua dimensi, seperti menampilkan pola bintang (*pattern*), mengakses elemen dalam matriks, atau membangun tabel data.

Melalui praktikum ini, mahasiswa diharapkan dapat memahami cara kerja perulangan *for* dan *nested for*, serta menerapkannya untuk menyelesaikan berbagai permasalahan logika dasar dalam pemrograman.

1.2 Tujuan

Tujuan dari praktikum ini adalah:

1. Memahami konsep dasar perulangan (*looping*) pada bahasa pemrograman Java.
2. Mampu menggunakan struktur **for loop** dengan benar.
3. Mengetahui cara kerja dan penerapan **nested loop** dalam berbagai kasus.
4. Melatih kemampuan logika berpikir dalam menyusun algoritma menggunakan struktur perulangan.

1.3 Manfaat Praktikum

Melalui pelaksanaan praktikum ini, mahasiswa diharapkan dapat memperoleh manfaat sebagai berikut:

1. Meningkatkan kemampuan analisis logika dan pemecahan masalah secara sistematis.
2. Memahami pentingnya efisiensi kode melalui penggunaan perulang

3. Mampu mengembangkan program sederhana yang menggunakan konsep *looping*.
4. Menjadi dasar bagi pemahaman konsep algoritma yang lebih kompleks di tahap selanjutnya, seperti array, matriks, dan algoritma pencarian/pengurutan.

BAB II PEMBAHASAN

2.1 Perulangan for (*for looping*) Sederhana

Perulangan for (*for looping*) dapat digunakan saat kita ingin mengulang kode program yang sama berkali-kali. Perulangan for dapat digunakan saat kita mengetahui secara pasti batas, atau kapan perulangan itu akan berhenti.

For loop terdiri dari 3 bagian utama, yaitu:

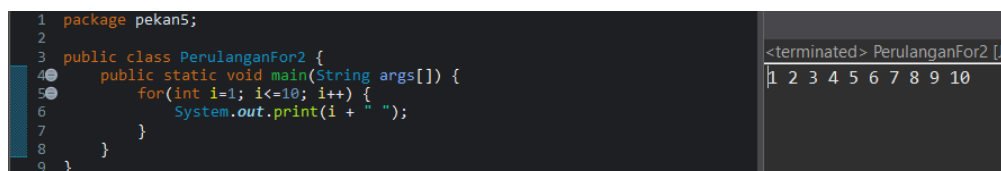
```
for(inisialisasi; kondisi; increment/decrement) {  
    //isi dari kode yang akan diulang}
```

Pada struktur di atas, bagian di dalam tanda kurung kurawal { } disebut *body of loop*, yaitu kumpulan perintah yang akan dijalankan secara berulang selama kondisi bernilai true.

Pada tahap inisialisasi, kita menggunakan dan mendeklarasi variabel yang hanya berlaku pada lingkup *for loop* itu saja.

Pada bagian kondisi, *for loop* akan terus berjalan sampai kondisinya tidak lagi memenuhi.

Pada bagian *increment/decrement*, terjadi perubahan nilai terhadap variabel terjadi perubahan nilai terhadap variabel yang digunakan dalam perulangan. Biasanya, nilai variabel tersebut akan bertambah (*increment*) atau berkurang (*decrement*) setiap kali satu siklus perulangan selesai dijalankan.



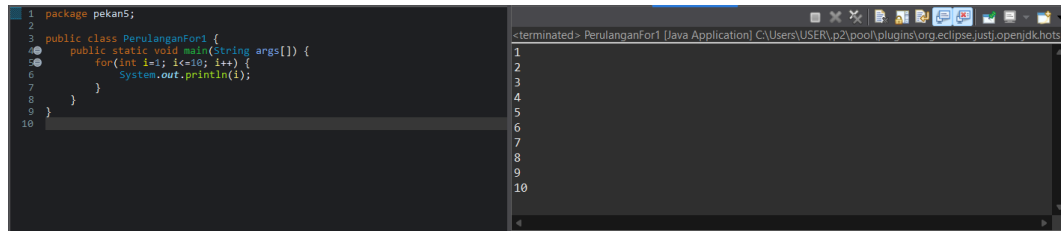
```
1 package pekan5;  
2  
3 public class PerulanganFor2 {  
4     public static void main(String args[]) {  
5         for(int i=1; i<=10; i++) {  
6             System.out.print(i + " ");  
7         }  
8     }  
9 }
```

<terminated> PerulanganFor2 [J
1 2 3 4 5 6 7 8 9 10

Kode Program 2.1

Pada Kode Program 2.1, di mana variabel *i* diinisialisasi dengan nilai 1, kemudian program akan terus mencetak nilai *i* ke layar selama kondisi *i* <= 10 terpenuhi. Setelah setiap iterasi, nilai *i* akan bertambah satu melalui proses increment *i++*. Ketika nilai *i* mencapai 11, kondisi menjadi false sehingga perulangan berhenti secara otomatis.

Kode program 2.2 di bawah menampilkan kode program yang mirip dengan Kode Program 2.1, tetapi dibuat baris baru tiap kali program mengeluarkan nilai *i*.

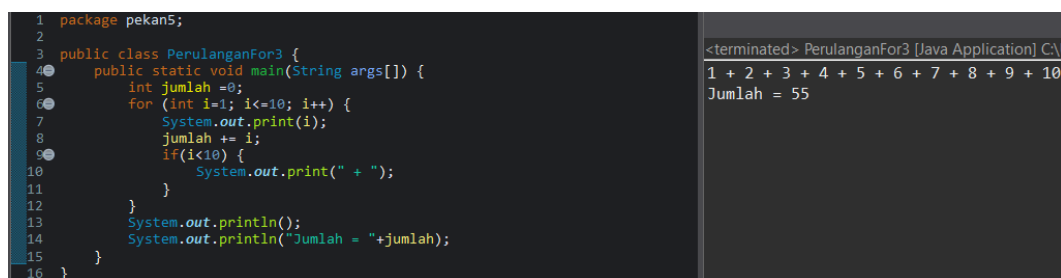


```
1 package pekan5;
2
3 public class PerulanganFor1 {
4     public static void main(String args[]) {
5         for (int i=1; i<=10; i++) {
6             System.out.println(i);
7         }
8     }
9 }
10
```

<terminated> PerulanganFor1 [Java Application] C:\Users\USER\p2\pool\plugins\org.eclipse.justi.openjdkhotsp
1
2
3
4
5
6
7
8
9
10

Kode Program 2.2

2.2 Perulangan *for* dengan Operasi Penjumlahan



```
1 package pekan5;
2
3 public class PerulanganFor3 {
4     public static void main(String args[]) {
5         int jumlah =0;
6         for (int i=1; i<=10; i++) {
7             System.out.print(i);
8             jumlah += i;
9             if(i<10) {
10                 System.out.print(" + ");
11             }
12         }
13         System.out.println();
14         System.out.println("Jumlah = "+jumlah);
15     }
16 }
```

<terminated> PerulanganFor3 [Java Application] C:\U
1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10
Jumlah = 55

Kode Program 2.3

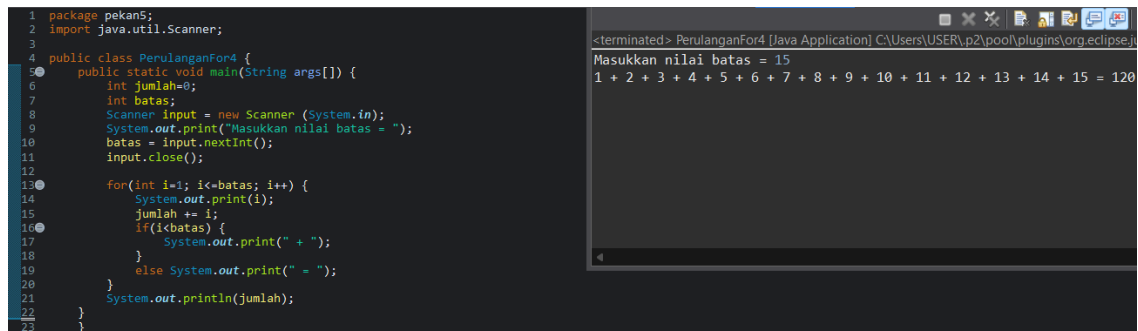
Program di atas merupakan pengembangan dari *for loop* sederhana yang telah dibahas sebelumnya. Tujuan program ini adalah untuk menampilkan deretan angka dari 1 sampai 10, kemudian menghitung total penjumlahan dari semua angka tersebut.

Variabel *jumlah* digunakan untuk menyimpan akumulasi hasil penjumlahan setiap nilai *i*. Pada setiap iterasi, perintah *jumlah += i*; menambahkan nilai *i* ke dalam variabel *jumlah*.

Kondisi *if (i < 10)* digunakan untuk menampilkan tanda “+” di antara angka, kecuali pada angka terakhir. Setelah perulangan selesai, program mencetak hasil total dengan menampilkan kalimat “*Jumlah = ...*”.

2.3 Perulangan *for* dengan Menghitung Angka dari 1 Sampai *n*

Kita bisa membuat Kode Program 2.3 menjadi lebih dinamis dengan menerima input dari pengguna, kemudian menghitung semua angka dari 1 hingga angka yang dimasukkan oleh pengguna melalui kelas *Scanner*.

The image shows a screenshot of an Eclipse IDE. On the left, a Java file named 'PerulanganFor4.java' is open, showing the following code:

```
1 package pekan5;
2 import java.util.Scanner;
3
4 public class PerulanganFor4 {
5     public static void main(String args[]) {
6         int jumlah=0;
7         int batas;
8         Scanner input = new Scanner (System.in);
9         System.out.print("Masukkan nilai batas = ");
10        batas = input.nextInt();
11        input.close();
12
13        for(int i=1; i<=batas; i++) {
14            System.out.print(i);
15            jumlah += i;
16            if(i<batas) {
17                System.out.print(" + ");
18            }
19            else System.out.print(" = ");
20        }
21        System.out.println(jumlah);
22    }
23 }
```

On the right, the 'Run and Debug' console shows the output of the program:

```
<terminated> PerulanganFor4 [Java Application] C:\Users\USER\p2\pool\plugins\org.eclipse.j
Masukkan nilai batas = 15
1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 + 11 + 12 + 13 + 14 + 15 = 120
```

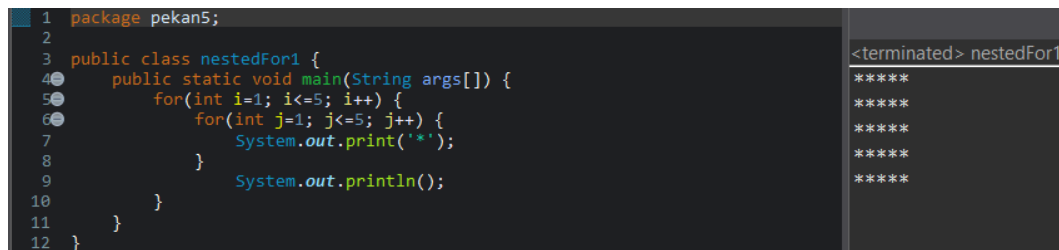
Kode Program 2.4

Nilai yang dimasukkan pengguna disimpan dalam variabel *batas*. Selanjutnya, perulangan *for* akan berjalan mulai dari $i = 1$ hingga $i \leq \textit{batas}$. Pada setiap iterasi, nilai i dijumlahkan ke variabel *jumlah*, dan hasil akhirnya ditampilkan ke layar. Contoh di atas adalah pengguna memasukkan nilai *batas* = 15.

2.4 Perulangan *for* di dalam Perulangan *for* lainnya (*nested for*)

Dalam pemrograman *Java*, sebuah struktur perulangan dapat ditempatkan di dalam perulangan lain. Konsep ini disebut *nested loop* atau perulangan bersarang. Penggunaan *nested for* memungkinkan program menjalankan proses berulang secara bertingkat, di mana setiap satu iterasi dari perulangan luar akan mengeksekusi seluruh perulangan di dalamnya.

1. Perulangan *for* Bersarang untuk Menampilkan Pola Persegi



```
1 package pekan5;
2
3 public class nestedFor1 {
4     public static void main(String args[]) {
5         for(int i=1; i<=5; i++) {
6             for(int j=1; j<=5; j++) {
7                 System.out.print('*');
8             }
9             System.out.println();
10        }
11    }
12 }
```

The output on the right shows a 5x5 grid of asterisks: ***** on five separate lines.

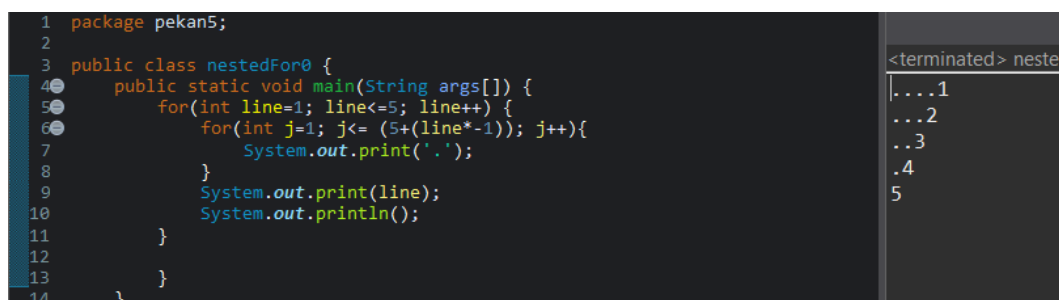
Kode Program 2.5

Program di atas merupakan contoh penggunaan *nested for* atau perulangan *for* di dalam *for* lainnya. Tujuannya adalah untuk menampilkan pola berbentuk persegi berukuran 5×5 menggunakan simbol bintang (*).

Pada bagian luar, terdapat perulangan dengan variabel *i* (`for (int i = 1; i <= 5; i++)`) yang berfungsi mengatur jumlah baris. Di dalamnya, terdapat perulangan kedua dengan variabel *j* (`for (int j = 1; j <= 5; j++)`) yang berfungsi mengatur jumlah kolom pada setiap baris.

Perintah `System.out.print('*');` akan mencetak satu simbol bintang tanpa berpindah ke baris baru. Setelah satu baris penuh dicetak (yakni ketika perulangan dalam selesai), perintah `System.out.println();` pada perulangan luar digunakan untuk memindahkan posisi pencetakan ke baris berikutnya.

2. Perulangan *nested for* untuk Menampilkan Pola dengan Simbol dan Angka



```
1 package pekan5;
2
3 public class nestedFor0 {
4     public static void main(String args[]) {
5         for(int line=1; line<=5; line++) {
6             for(int j=1; j<= (5+(line*-1)); j++){
7                 System.out.print('.');
8             }
9             System.out.print(line);
10            System.out.println();
11        }
12    }
13 }
14 }
```

The output on the right shows a pattern of dots and numbers:1, ...2, ..3, .4, 5 on five separate lines.

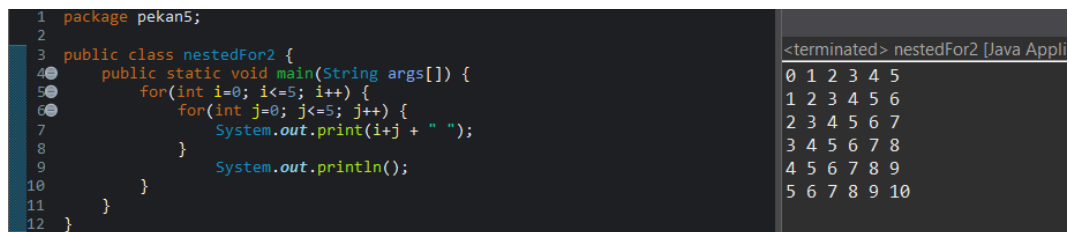
Kode Program 2.6

Program di atas menggunakan struktur *nested for* untuk menampilkan pola kombinasi titik (.) dan angka. Perulangan luar (for (int line = 1; line <= 5; line++)) mengatur jumlah baris yang akan dicetak, yaitu sebanyak lima kali.

Pada setiap baris, perulangan dalam (for (int j = 1; j <= (5 + (line * -1)); j++)) menentukan berapa banyak simbol titik yang akan dicetak. Rumus (5 + (line * -1)) menyebabkan jumlah titik berkurang satu di setiap baris berikutnya, karena nilai *line* bertambah.

Setelah titik-titik dicetak, program menampilkan angka sesuai nilai *line* dengan perintah System.out.print(line);. Kemudian, System.out.println(); digunakan untuk berpindah ke baris baru.

3. Perulangan *nested for* untuk Operasi Penjumlahan Dua Variabel (Tabel Penjumlahan)



```
1 package pekan5;
2
3 public class nestedFor2 {
4     public static void main(String args[]) {
5         for(int i=0; i<=5; i++) {
6             for(int j=0; j<=5; j++) {
7                 System.out.print(i+j + " ");
8             }
9             System.out.println();
10        }
11    }
12 }
```

<terminated> nestedFor2 [Java Appli
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

Kode Program 2.7

Program ini menampilkan hasil penjumlahan dua variabel *i* dan *j* menggunakan struktur *nested for*. Perulangan luar (for (int i = 0; i <= 5; i++)) berfungsi untuk mengatur baris, sedangkan perulangan dalam (for (int j = 0; j <= 5; j++)) menghasilkan kolom berisi hasil penjumlahan antara *i* dan *j*.

Pada setiap iterasi, nilai dari *i* + *j* dicetak diikuti dengan spasi agar hasilnya tersusun dalam bentuk tabel. Setelah satu baris selesai, System.out.println(); digunakan untuk berpindah ke baris berikutnya. Hasil keluaran dari program ini berupa tabel penjumlahan yang di mana elemen ke-(*i,j*) merupakan penjumlahan baris (*i*,1) dan kolom (1,*j*).

BAB III

KESIMPULAN

Berdasarkan hasil praktikum yang telah dilakukan mengenai perulangan *for* dan *nested for* pada bahasa pemrograman *Java*, dapat disimpulkan bahwa struktur perulangan *for* digunakan ketika jumlah iterasi sudah diketahui secara pasti. Struktur ini terdiri atas tiga bagian utama, yaitu *inisialisasi*, *kondisi*, dan *increment/decrement*, yang bekerja bersama untuk mengontrol proses perulangan agar berjalan sesuai logika program. Dengan menggunakan *for loop*, penulisan kode menjadi lebih efisien karena satu blok perintah dapat dijalankan berulang kali tanpa perlu menulis ulang instruksi yang sama.

Secara keseluruhan, praktikum ini membantu mahasiswa memahami bagaimana logika perulangan bekerja serta bagaimana interaksi antarvariabel dalam *loop* dapat memengaruhi hasil keluaran (*output*). Pemahaman ini menjadi dasar penting untuk mempelajari konsep pemrograman yang lebih lanjut, seperti penggunaan *array*, *function*, dan algoritma pengulangan bersyarat yang lebih kompleks.

DAFTAR PUSTAKA

- [1] Oracle. *Java Documentation*. <https://docs.oracle.com/javase/>