

LAPORAN PRATIKUM
ALGORITMA DAN PEMROGRAMAN
“PERULANGAN NESTED FOR”

Disusun Oleh:

Rifqi Aditya

2511533002

Dosen Pengampu :

Dr. Wahyudi, S.T, M.T

Asisten Praktikum :

Rahmad Dwirizki Olders



DEPARTEMEN INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Puji syukur ke hadirat Allah SWT karena atas rahmat dan karunia-Nya saya dapat menyelesaikan laporan praktikum mata kuliah Algoritma dan Pemrograman dengan judul "*Laporan Cuaca*".

Laporan ini disusun untuk memenuhi salah satu tugas praktikum sekaligus sebagai sarana pembelajaran dalam memahami dasar-dasar pemrograman menggunakan bahasa Java. Melalui praktikum ini, penulis dapat mempelajari penggunaan tipe data dasar seperti *integer*, *float*, *char*, dan *boolean*, serta bagaimana menampilkan informasi ke layar menggunakan perintah `System.out.println`.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, penulis mengucapkan terima kasih kepada dosen pengampu, asisten praktikum, serta rekan-rekan yang telah membantu dalam proses penyusunan laporan ini.

Akhir kata, semoga laporan ini dapat memberikan manfaat, baik bagi penulis sendiri maupun bagi pembaca, khususnya dalam memperdalam pemahaman tentang pemrograman dasar dengan Java.

Padang, 26 September 2025

Penulis

Daftar Pustaka

BAB I PENDAHULUAN	4
1.1 Latar Belakang	4
1.2 Tujuan	4
1.3 Manfaat.....	4
BAB II PEMBAHASAN.....	5
2.1 Program PerulanganFor1.....	5
2.2 Program PerulanganFor2.....	6
2.3 Program PerulanganFor3.....	7
2.4 Program PerulanganFor4.....	9
2.5 Program nestedFor0	10
2.6 Program nestedFor1	11
2.7 Program nestedFor2	12
BAB III KESIMPULAN	14
DAFTAR PUSTAKA	15

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pemrograman Java merupakan landasan penting dalam mempelajari algoritma. Untuk menguasainya, pemahaman mendalam tentang elemen dasar seperti tipe data sangatlah diperlukan. Praktikum ini bertujuan untuk memberikan pengalaman praktis dalam penggunaan tipe data di Java, meliputi *integer*, *float*, *char*, *double* dan *Boolean*.

1.2 Tujuan

1. Melatih keterampilan menampilkan output ke layar dengan format yang rapi.
2. Membiasakan penggunaan penginputan sebagai media tampilan informasi.
3. Menerapkan deklarasi variabel dengan berbagai tipe data dasar di Java.

1.3 Manfaat

1. Mahasiswa mampu membuat program sederhana.
2. Mahasiswa memahami penggunaan tipe data dasar dalam Java.
3. Mahasiswa dapat mengaitkan pembelajaran pemrograman dengan kasus nyata di dunia industri.

BAB II PEMBAHASAN

Bab ini berfokus pada pembahasan dan analisis implementasi struktur kontrol perulangan dalam bahasa pemrograman Java. Perulangan (looping) adalah mekanisme fundamental yang memungkinkan program untuk mengeksekusi blok kode secara berulang kali selama kondisi tertentu terpenuhi. Praktikum ini secara spesifik mengkaji penggunaan perulangan for, baik dalam bentuk perulangan tunggal maupun perulangan bersarang (*nested loop*).

2.1 Program PerulanganFor1

Program PerulanganFor1 dirancang untuk mendemonstrasikan struktur perulangan for paling dasar, yaitu mencetak deret angka dari 1 sampai 10 secara vertikal.

2.1.1 Uraian Kode Program

```
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
```

2.1.2 Penjelasan Langkah Kerja

1. Program dimulai pada metode main.
2. Struktur perulangan for diinisialisasi. Variabel kontrol i dideklarasikan dan diinisialisasi dengan nilai 1 (int i = 1).

3. Program memeriksa kondisi perulangan ($i \leq 10$). Saat i bernilai 1, kondisi ini bernilai benar (*true*).
4. Karena kondisi benar, badan perulangan (baris 7) dieksekusi: `System.out.println(i);`. Perintah ini mencetak nilai i saat ini (yaitu 1) diikuti dengan karakter baris baru (*newline*).
5. Setelah badan perulangan selesai, bagian pembaruan (*update*) dieksekusi: $i++$. Nilai i sekarang menjadi 2.
6. Langkah 3 sampai 5 diulang kembali. Proses ini berlanjut selama kondisi $i \leq 10$ bernilai benar.
7. Ketika i bernilai 11, kondisi $i \leq 10$ bernilai salah (*false*). Perulangan `for` berhenti.
8. Program selesai.

2.1.3 Analisis Hasil

Keluaran dari program ini adalah deret angka dari 1 hingga 10, di mana setiap angka dicetak pada baris baru.

2.2 Program PerulanganFor2

Program `PerulanganFor2` memiliki fungsionalitas yang hampir identik dengan `PerulanganFor1`, namun dengan sedikit modifikasi pada perintah pencetakan.

2.2.1 Uraian Kode Program

```
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.println(i+" ");
8         }
9     }
10
11 }
12 |
```

2.2.2 Penjelasan Langkah Kerja

1. Langkah kerja program ini identik dengan PerulanganFor1 (poin 2.1.2), dengan perbedaan pada langkah ke-4.
2. Pada badan perulangan (baris 7), perintah yang dieksekusi adalah `System.out.println(i + " ");`.
3. Di sini, terjadi konkatenasi string (penggabungan string). Nilai integer dari variabel `i` secara implisit diubah menjadi *string* dan digabungkan dengan string literal `" "`.
4. Hasil penggabungan (misal, `"1 "`, `"2 "`, dst.) dicetak ke konsol, diikuti dengan karakter baris baru.

2.2.3 Analisis Hasil

Keluaran program ini tetap berupa deret angka 1 hingga 10 secara vertikal, namun setiap angka kini diikuti oleh satu karakter spasi.

2.3 Program PerulanganFor3

Program PerulanganFor3 dikembangkan untuk menghitung dan menampilkan proses penjumlahan deret angka dari 1 hingga 10, serta menampilkan hasil akhirnya.

2.3.1 Uraian Kode Program

```

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.println(i);
9             jumlah= jumlah+i;
10            if (i<10) {
11                System.out.println();
12                System.out.println("  + ");
13            }
14        }
15        System.out.println();
16        System.out.println("Jumlah = "+jumlah);
17    }
18
19 }
20

```

2.3.2 Penjelasan Langkah Kerja

1. Program menginisialisasi variabel jumlah dengan nilai 0. Variabel ini berfungsi sebagai akumulator.
2. Perulangan for diatur untuk beriterasi dari $i = 1$ sampai $i = 10$.
3. Di dalam perulangan:
 - `System.out.println(i);` mencetak nilai i saat ini.
 - `jumlah = jumlah + i;` menambahkan nilai i ke variabel jumlah.
 - Struktur kondisional `if (i < 10)` memeriksa apakah i belum mencapai nilai akhir (10).
 - Jika $i < 10$ (benar), program mencetak baris baru (untuk spasi) dan string " +" pada baris berikutnya, sebagai representasi visual dari operasi penjumlahan.
4. Setelah perulangan selesai (ketika i mencapai 11), program melanjutkan ke baris 15.
5. `System.out.println();` mencetak baris baru untuk memberi jarak.
6. `System.out.println("Jumlah = " + jumlah);` mencetak hasil akhir dari total penjumlahan yang tersimpan dalam variabel jumlah.

2.3.3 Analisis Hasil

Keluaran program akan menampilkan proses penjumlahan $1+2+\dots+10$ secara vertikal, dan diakhiri dengan total penjumlahannya, yaitu 55.

2.4 Program PerulanganFor4

Program PerulanganFor4 merupakan pengembangan dari PerulanganFor3 yang membuatnya lebih dinamis dengan menerima masukan batas perulangan dari pengguna.

2.4.1 Uraian Kode Program

```
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.println(i);
16            jumlah = jumlah+i;
17            if (i<batas) {
18                System.out.println(" + ");
19            } else {
20                System.out.println(" = ");
21            }
22        }
23        System.out.println(jumlah);
24    }
25
26 }
27
```

2.4.2 Penjelasan Langkah Kerja

1. Program mengimpor kelas Scanner dari paket java.util untuk menangani masukan pengguna.
2. Variabel jumlah (akumulator) dan batas (penyimpan nilai masukan) dideklarasikan.
3. Objek Scanner bernama input dibuat untuk membaca dari System.in (konsol).
4. Program mencetak prompt "Masukkan nilai batas".
5. batas = input.nextInt(); program berhenti sejenak, menunggu pengguna memasukkan sebuah bilangan bulat dan menekan Enter. Nilai ini disimpan di variabel batas.
6. input.close(); menutup aliran masukan, ini adalah praktik yang baik untuk mencegah kebocoran sumber daya (*resource leak*).
7. Perulangan for diinisialisasi untuk beriterasi dari i = 1 hingga i <= batas.
8. Di dalam perulangan:
 - o Nilai i dicetak.

- Nilai i ditambahkan ke jumlah.
 - Struktur kondisional if-else digunakan:
 - Jika $i < \text{batas}$ (bukan iterasi terakhir), cetak "+".
 - Jika $i == \text{batas}$ (iterasi terakhir), cetak "=".
9. Setelah perulangan selesai, `System.out.println(jumlah);` mencetak total akhir.

2.4.3 Analisis Hasil

Keluaran program ini serupa dengan PerulanganFor3, tetapi batas atasnya ditentukan oleh pengguna. Jika pengguna memasukkan 5, program akan menghitung $1+2+3+4+5 = 15$

2.5 Program nestedFor0

Program `nestedFor0` mendemonstrasikan penggunaan perulangan bersarang (*nested loop*) untuk menciptakan pola keluaran yang kompleks, dalam hal ini segitiga siku-siku terbalik yang diisi dengan angka.

2.5.1 Uraian Kode Program

```

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 }
16

```

2.5.2 Penjelasan Langkah Kerja

1. Program menggunakan dua perulangan for yang bersarang.
2. Perulangan Luar (dikontrol variabel `line`): Beriterasi dari 1 hingga 5. Perulangan ini mengontrol jumlah baris yang akan dicetak.
3. Pada setiap iterasi perulangan luar:

- Perulangan Dalam (dikontrol variabel j): Perulangan ini dieksekusi. Batas atasnya adalah ekspresi matematis $(-1 * \text{line} + 5)$.
 - Saat line = 1, kondisi $j \leq 4$. Perulangan dalam berjalan 4 kali, mencetak
 - Saat line = 2, kondisi $j \leq 3$. Perulangan dalam berjalan 3 kali, mencetak ...
 - ...
 - Saat line = 5, kondisi $j \leq 0$. Perulangan dalam tidak berjalan sama sekali (0 kali).
- `System.out.print(line);` mencetak nilai line saat ini (nomor baris) setelah perulangan dalam selesai.
- `System.out.println();` memindahkan kursor ke baris baru.

2.5.3 Analisis Hasil

Kombinasi dari perulangan dalam yang jumlah iterasinya menurun dan pencetakan nomor baris di akhir menghasilkan pola segitiga yang rata kanan.

2.6 Program nestedFor1

Program nestedFor1 adalah contoh klasik penggunaan perulangan bersarang untuk mencetak bentuk persegi (kotak) solid.

2.6.1 Uraian Kode Program

```

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            // tp end the line
12        }
13    }
14
15 }
16

```

2.6.2 Penjelasan Langkah Kerja

1. Perulangan Luar (variabel i): Beriterasi dari 1 hingga 5. Perulangan ini mengontrol baris.
2. Untuk setiap iterasi i:
 - Perulangan Dalam (variabel j): Beriterasi dari 1 hingga 5. Perulangan ini mengontrol kolom.
 - Pada setiap iterasi j, `System.out.print("*");` dieksekusi, mencetak satu karakter bintang tanpa baris baru.
 - Setelah perulangan dalam selesai (telah mencetak 5 bintang), `System.out.println();` dieksekusi (baris 10). Ini memindahkan kursor ke baris berikutnya, "mengakhiri" baris saat ini.
3. Perulangan luar berlanjut ke iterasi i berikutnya, mengulangi proses.

2.6.3 Analisis Hasil

Program ini akan mencetak 5 baris, dan setiap baris berisi 5 karakter bintang, menghasilkan sebuah persegi berukuran 5x5

2.7 Program nestedFor2

Program `nestedFor2` menggunakan perulangan bersarang untuk menghasilkan *grid* atau tabel, di mana isi selnya merupakan hasil operasi aritmetika dari indeks baris dan kolom.

2.7.1 Uraian Kode Program

```
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
```

2.7.2 Penjelasan Langkah Kerja

1. Perulangan Luar (variabel i): Beriterasi dari 0 hingga 5 (total 6 iterasi).
2. Untuk setiap iterasi i:
 - Perulangan Dalam (variabel j): Beriterasi dari 0 hingga 5 (total 6 iterasi).
 - Pada setiap iterasi j, program menghitung $i + j$.
 - `System.out.print(i + j + " ");` mencetak hasil penjumlahan tersebut, diikuti oleh spasi.
 - Setelah perulangan dalam selesai, `System.out.println();` dieksekusi untuk pindah ke baris baru.

2.7.3 Analisis Hasil

Keluaran program adalah sebuah tabel berukuran 6x6. Setiap sel dalam tabel berisi nilai $i+j$, di mana i adalah indeks baris (0-5) dan j adalah indeks kolom (0-5). Ini mirip dengan tabel penjumlahan.

BAB III

KESIMPULAN

Berdasarkan pembahasan dan analisis program pada bab sebelumnya, dapat ditarik kesimpulan mengenai penggunaan dan karakteristik dari berbagai struktur kontrol percabangan dalam bahasa Java:

1. Struktur if-else merupakan mekanisme untuk percabangan dua arah. Struktur ini efektif untuk menangani satu kondisi logis dengan dua kemungkinan hasil yang saling eksklusif.
2. Struktur if-else if-else adalah pengembangan untuk percabangan multi-arrah. Keunggulannya terletak pada efisiensi, di mana evaluasi kondisi dilakukan secara sekuensial dan akan berhenti segera setelah salah satu kondisi terpenuhi.
3. Penggunaan multi if cocok untuk situasi di mana terdapat beberapa kondisi yang tidak saling terikat dan perlu dievaluasi secara mandiri. Berbeda dengan if-else if, semua kondisi if akan diperiksa tanpa terkecuali, yang memungkinkan lebih dari satu blok kode dapat dieksekusi jika kondisinya terpenuhi.
4. Struktur switch-case menawarkan alternatif yang lebih terstruktur dan sering kali lebih mudah dibaca dibandingkan rantai if-else if yang panjang, terutama saat membandingkan sebuah peubah tunggal terhadap serangkaian nilai konstanta yang spesifik.

DAFTAR PUSTAKA

- [1] Deitel, P. J. (2018). *Java: How to Program, Early Objects*.
- [2] Horstmann, C. S. (2019). *Core Java Volume I--Fundamentals*.
- [3] Rosihan Ari Yuana, S. M. (2022). *Pemrograman Java*.