LAPORAN PRAKTIKUM

ALGORITMA DAN PEMROGRAMAN

“laporan Praktikum Pekan 5”

Disusun Oleh:

Faiz Fikri Satria

2511533026

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

Asisten Praktikum : Rahmad Dwirizki Olders

DAPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Puji syukur saya panjatkan ke hadirat Tuhan Yang Maha Esa atas rahmat dan karunia-Nya, sehingga dapat menyelesaikan laporan praktikum ini dengan baik. Tidak lupa juga saya ucapkan terima kasih kepada bapak Wahyudi. Dr. S.T.M.T sebagai dosen pembimbing dan Rahmad Dwirizki Olders yang telah membantu dalam pelaksanaan praktikum ini. Laporan ini saya susun untuk memenuhi tugas praktikum Algoritma dan Pemrograman dari praktikum pertemuan kelima mengenai pemahaman dasar pemrograman bahasa Java menggunakan IDE Eclipse. Praktikum ini bertema pembuatan kode Perulangan For, dan Nasted If. Saya berharap laporan ini dapat memberikan manfaat bagi pembaca dalam memahami konsep dasar pemrograman Java.

Laporan ini dibuat dengan harapan dapat memberikan pemahaman dasar mengenai penggunaan Bahasa pemrograman Java, khususnya mengenai hal struktur dasar program dan fungsi.

Saya menyadari bahwa laporan ini masih jauh dari sempurna, oleh sebab itu, kritik dan saran sangat diharapkan demi penyempurnaan laporan di masa mendatang.

Padang 02 November, 2025

Penulis

DAFTAR ISI

BAB I

PENDAHULUAN

1.3 Tujuan Praktikum

Adapun tujuan dari pelaksanaan praktikum ini adalah sebagai berikut:

1. Memahami sintaks dan mekanisme kerja dari perulangan tunggal (for loop) dalam Java.
2. Mampu mengimplementasikan perulangan bersarang (Nested For) untuk mencetak pola dua dimensi atau matriks.
3. Mampu mengintegrasikan pernyataan kondisional (if dan if-else) di dalam struktur perulangan untuk mengontrol alur program dan format *output*.
4. Mendapatkan pengalaman praktis dalam menulis, menjalankan, dan menganalisis kode Java menggunakan Eclipse IDE.

1.4 Manfaat Praktikum

Manfaat yang diperoleh dari praktikum ini meliputi:

1. Manfaat Teoritis: Memperkuat pemahaman konseptual tentang struktur kontrol dan alur eksekusi program.
2. Manfaat Praktis: Meningkatkan keterampilan dalam *debugging* dan analisis kode, serta kemampuan untuk menciptakan algoritma yang lebih kompleks dan terstruktur.
3. Manfaat Pengembangan: Menjadi dasar yang kuat untuk mempelajari topik pemrograman Java yang lebih lanjut, seperti array multidimensi dan pemrograman berorientasi objek.

BAB II

PEMBAHASAN

2.1 Nested For untuk Pola Titik dan Angka

Kode ini merupakan contoh sederhana dari nested for loop dalam Java yang digunakan untuk menghasilkan pola output berupa titik-titik yang berkurang diikuti oleh angka yang meningkat pada setiap baris. Konsep ini berguna untuk memahami bagaimana perulangan bersarang dapat mengontrol struktur dua dimensi, seperti pola grafik atau matriks. Dengan loop luar mengatur jumlah baris dan loop dalam menghitung jumlah titik berdasarkan rumus aritmatika, kode ini membantu mahasiswa melatih logika iterasi bertingkat dan penggunaan System.out untuk output konsol. Secara keseluruhan, ini memperkenalkan aplikasi dasar nested loop dalam pemrograman visual sederhana.

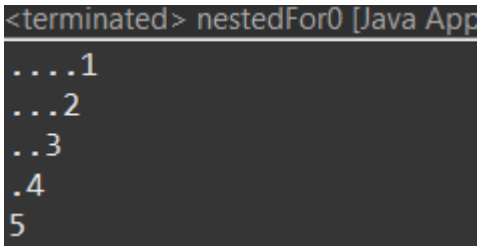
2.1.1 Langkah-langkah

- Buka Eclipse dan buat project baru.
- Buat class baru dengan method main.
- Tulis kode nested for: loop luar untuk baris (line), loop dalam untuk mencetak titik berdasarkan rumus (-1 * line + 5).
- Cetak angka line setelah titik, lalu pindah baris.
- Jalankan program dan amati output.

2.1.2 Contoh Pemrograman

```
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 }
```

2.1.3 Hasil Output



2.1.4 Analisis

Kode ini menghasilkan pola segitiga terbalik dengan titik yang berkurang dan angka yang meningkat. Loop luar mengontrol baris, sementara loop dalam menghitung jumlah titik menggunakan rumus aritmatika. Tidak ada error, output sesuai ekspektasi untuk 5 baris.

2.1.5 Teori Mengenai Kode

Nested for adalah perulangan bersarang di mana satu loop berada di dalam loop lain. Loop luar dieksekusi sekali, lalu loop dalam dieksekusi penuh sebelum kembali ke loop luar. Teori ini berdasarkan pada konsep iterasi berjenjang, berguna untuk array 2D atau pola grafik. Dalam Java, System.out.print() digunakan untuk output tanpa pindah baris, sementara println() untuk pindah baris.

2.2 Nested For untuk Penjumlahan i + j

Kode ini menunjukkan penggunaan nested for loop untuk membuat matriks output di mana setiap elemen adalah hasil penjumlahan variabel i dari loop luar dan j dari loop dalam, menciptakan pola angka yang meningkat secara diagonal. Hal ini membantu dalam memahami iterasi dua dimensi, yang sering diterapkan dalam pengolahan data seperti array atau tabel. Dengan rentang dari 0 hingga 5, kode ini melatih kemampuan dalam mengontrol alur eksekusi loop bersarang dan menggunakan System.out untuk format output berbaris. Secara keseluruhan, ini memperkuat konsep bagaimana nested loop dapat menghasilkan struktur data kompleks dari operasi sederhana.

2.2.1 Langkah-langkah

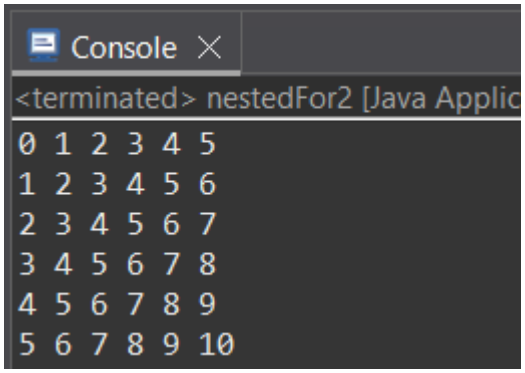
- Buat class baru bernama nestedFor2.

- Tulis nested for: loop luar untuk i (0-5), loop dalam untuk j (0-5).
- Cetak i + j diikuti spasi.
- Pindah baris setelah loop dalam selesai.
- Eksekusi dan verifikasi output.

2.2.2 Contoh Pemrograman

```
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();
0              //to end the line
1          }
2      }
3  }
```

2.2.3 Hasil Output



2.2.4 Analisis

Output berupa matriks 6x6 di mana setiap elemen adalah penjumlahan indeks baris dan kolom. Kode berjalan efisien tanpa infinite loop, menghasilkan pola aritmatika yang meningkat. Analisis menunjukkan nested for cocok untuk operasi matriks.

2.2.5 Teori Mengenai Kode

Nested for sering digunakan untuk iterasi dua dimensi, seperti pada array atau tabel. Teori dasarnya adalah eksekusi inner loop penuh untuk setiap iterasi outer loop, dengan kompleksitas O(n^2). Di Java, inisialisasi variabel loop bisa dari 0 untuk indeks berbasis nol.

2.3 Nested For untuk Pola Bintang

Kode ini adalah implementasi nested for loop yang sederhana untuk menghasilkan pola persegi berisi bintang (*), dengan loop luar mengatur baris dan loop dalam mengatur kolom, sehingga menciptakan grid 5x5. Konsep ini berguna untuk memahami dasar pembuatan pola tetap dalam pemrograman, yang dapat dikembangkan untuk grafik ASCII atau simulasi matriks. Dengan penggunaan package Pekan_lima, kode ini juga memperkenalkan organisasi struktur proyek di Java. Secara keseluruhan, ini membantu mahasiswa dalam melatih kontrol alur program untuk output visual statis dan memahami pentingnya pindah baris setelah setiap iterasi baris.

2.3.1 Langkah-langkah

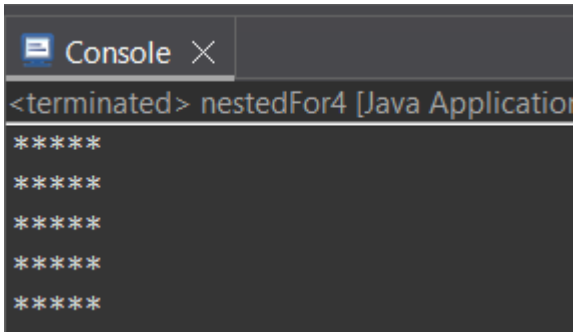
1. Buat package Pekan_lima dan class nestedFor4.
2. Tulis nested for: loop luar untuk baris (1-5), loop dalam untuk kolom (1-5).
3. Cetak "*" untuk setiap iterasi dalam.
4. Pindah baris setelah loop dalam.
5. Jalankan program.

2.3.2 Contoh Pemrograman

```java
package Pekan_lima;

public class nestedFor4 {
    public static void main (String[] args) {
        for (int i = 1; i <= 5; i++) {
            for (int j = 1; j <= 5; j++) {
                System.out.print("*");
            }
            System.out.println();
            //to end the line
        }
    }
}
```

2.3.3 Hasil Output



2.3.4 Analisis

Kode menghasilkan persegi 5x5 bintang. Sederhana dan tanpa error, menunjukkan penggunaan nested for untuk pola statis. Analisis: Efisien untuk ukuran kecil, tapi bisa lambat untuk n besar.

2.3.5 Teori Mengenai Kode

Teori nested for mirip subbab sebelumnya, fokus pada pembuatan pola tetap. Di Java, package digunakan untuk organisasi kode, mencegah konflik nama class.

2.4 Perulangan For Sederhana

Kode ini menunjukkan perulangan for dasar dengan kesalahan pada kondisi loop, di mana kondisi selalu true sehingga menyebabkan infinite loop, mencetak angka secara terus-menerus. Hal ini berguna sebagai contoh negatif untuk memahami pentingnya kondisi yang tepat dalam struktur for, yang terdiri dari inisialisasi, kondisi, dan increment. Dengan package Pekan_lima, kode ini melatih identifikasi error umum seperti loop tak berujung. Secara keseluruhan, ini memperkuat pemahaman bahwa variabel loop harus terlibat dalam kondisi untuk menghentikan iterasi pada waktu yang tepat, menghindari konsumsi resource berlebih.

2.4.1 Langkah-langkah

1. **Buat package Pekan_lima dan class PerulanganFor1.**
2. **Tulis for loop dengan kondisi salah (1 <= 10).**
3. **Cetak i di setiap iterasi.**
4. **Jalankan dan amati perilaku (infinite loop).**
5. **Perbaiki kondisi menjadi i <= 10.**

2.4.2 Contoh Pemrograman

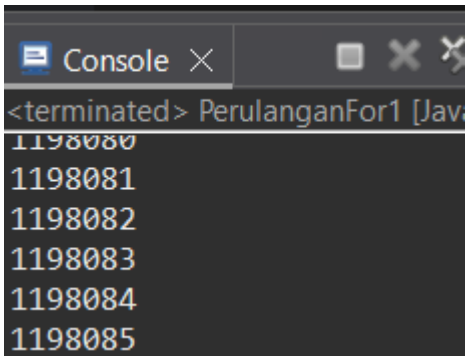
```java
package Pekan_lima;

public class PerulanganFor1 {

    public static void main (String [] args) {
        for (int i = 1; 1 <= 10; i++) {
            System.out.println( i );
        }
    }
}

```

2.4.3 Hasil Output



2.4.4 Analisis

Kode menyebabkan infinite loop karena kondisi selalu true (1 <= 10). Analisis: Kesalahan umum pada penulisan kondisi, menyebabkan program tidak berhenti. Perbaikan: Ganti menjadi i <= 10 untuk output 1 hingga 10.

2.4.5 Teori Mengenai Kode

Perulangan for terdiri dari inisialisasi, kondisi, dan increment. Teori: Kondisi harus bergantung pada variabel loop agar bisa false suatu saat, mencegah infinite loop. Di Java, ini berguna untuk iterasi terbatas.

2.5 Perulangan For untuk Cetak Angka Satu Baris

Kode ini adalah contoh perulangan for sederhana yang mencetak angka 1 hingga 10 dalam satu baris, menggunakan System.out.print() untuk menghindari pindah baris setiap iterasi. Konsep ini membantu memahami perbedaan antara print dan println, serta struktur dasar for loop untuk iterasi numerik. Kode ini melatih kemampuan dalam mengontrol format output konsol tanpa elemen kondisional tambahan. Secara keseluruhan, ini memperkenalkan aplikasi for loop dalam tugas-tugas sederhana seperti generasi urutan data, yang dapat dikembangkan untuk pengolahan array atau list di Java.

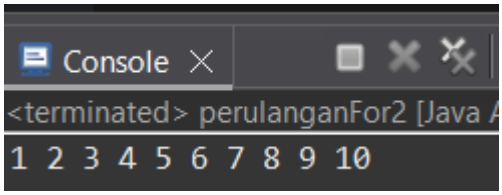
2.5.1 Langkah-langkah

1. **Buat class perulanganFor2.**
2. **Tulis for loop dari 1 hingga 10.**
3. **Cetak i diikuti spasi menggunakan print().**
4. **Jalankan program.**

2.5.2 Contoh Pemrograman

```
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
10     }
```

2.5.3 Hasil Output



2.5.4 Analisis

Output satu baris angka 1-10. Kode efisien, tanpa error. Analisis: Penggunaan print() vs println() memengaruhi format output.

2.5.5 Teori Mengenai Kode

Teori for loop: Struktur dasar untuk iterasi numerik. Di Java, increment i++ meningkatkan variabel setelah eksekusi body.

2.6 Perulangan For dengan If untuk Penjumlahan

Kode ini menggabungkan perulangan for dengan kondisional if untuk menghitung dan mencetak penjumlahan angka 1 hingga 10, di mana if digunakan untuk menambahkan tanda "+" hanya jika bukan angka terakhir. Konsep ini berguna untuk memahami integrasi struktur kontrol kondisional dalam loop, memungkinkan logika dinamis per iterasi. Dengan variabel jumlah untuk akumulasi, kode ini melatih perhitungan aritmatika iteratif. Secara keseluruhan, ini memperkuat pemahaman tentang bagaimana for dan if dapat menciptakan output yang formatted seperti ekspresi matematika, berguna dalam program kalkulator sederhana.

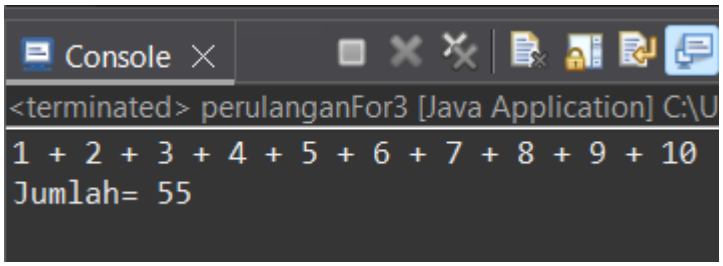
2.6.1 Langkah-langkah

1. Inisialisasi variabel jumlah = 0.
2. Tulis for loop 1-10, tambah i ke jumlah.
3. Gunakan if (i < 10) untuk cetak " + ".
4. Cetak hasil jumlah setelah loop.
5. Jalankan.

2.6.2 Contoh Pemrograman

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

2.6.3 Hasil Output



2.6.4 Analisis

Kode menghitung sum 1-10 sambil mencetak ekspresi. If mencegah "+" di akhir. Analisis: Kombinasi for dan if baik untuk logika kondisional dalam loop.

2.6.5 Teori Mengenai Kode

Teori: If di dalam for memungkinkan keputusan per iterasi. Formula sum n(n+1)/2, tapi dihitung iteratif di sini.

2.7 Perulangan For dengan If dan Input

Kode ini merupakan perulangan for yang dinamis dengan input pengguna melalui Scanner, menghitung penjumlahan hingga batas yang dimasukkan, dan menggunakan if-else untuk format output seperti "1 + 2 + ... = hasil". Konsep ini

memperkenalkan interaksi pengguna, menggabungkan loop, kondisional, dan input handling. Dengan close() pada Scanner, kode ini melatih praktik baik untuk manajemen resource. Secara keseluruhan, ini membantu memahami program interaktif di Java, berguna untuk aplikasi yang memerlukan input variabel seperti kalkulator atau simulator data.

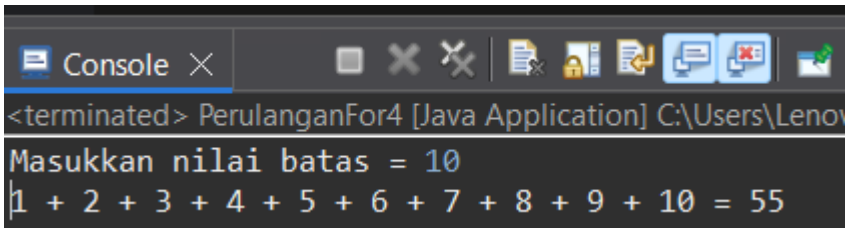
2.7.1 Langkah-langkah

1. Import Scanner untuk input.
2. Ambil input batas.
3. Inisialisasi jumlah = 0.
4. Loop for 1-batas, tambah i, gunakan if untuk " + " atau " = ".
5. Cetak jumlah.

2.7.2 Contoh Pemrograman

```
5  public class PerulanganFor4 {
6      public static void main(String[] args) {
7          int jumlah=0;
8          int batas;
9          Scanner input= new Scanner (System.in);
10         System.out.print("Masukkan nilai batas = ");
11         batas= input.nextInt();
12         input.close();
13         for (int i=1;i<=batas;i++) {
14             System.out.print(i);
15             jumlah= jumlah+i;
16             if (i<batas) {
17                 System.out.print(" + ");
18             } else {
19                 System.out.print(" = ");
20             }
21         }
22         System.out.println(jumlah);
23     }
24 }
```

2.7.3 Hasil Output



2.7.4 Analisis

Kode interaktif, menghitung sum hingga batas input. If else mengatur format. Analisis: Scanner berguna untuk input, close() mencegah leak.

2.7.5 Teori Mengenai Kode

Teori: Kombinasi for, if, dan Scanner untuk program dinamis. Di Java, input harus ditangani dengan exception handling untuk robustitas, tapi sederhana di sini.

BAB III

KESIMPULAN

Praktikum pertemuan kedua ini berhasil memberikan pemahaman dasar mengenai nested for dan perulangan for dengan if dalam bahasa Java menggunakan Eclipse. Melalui pembuatan kode-kode sederhana, mahasiswa dapat melihat bagaimana struktur kontrol ini bekerja untuk menghasilkan pola output dan perhitungan numerik. Nested for terbukti efektif untuk iterasi bersarang seperti pembuatan matriks atau pola, sementara integrasi if dalam loop menambahkan fleksibilitas kondisional, menghindari kesalahan seperti infinite loop pada salah satu contoh. Secara keseluruhan, praktikum ini memperkuat fondasi logika pemrograman.

Dari analisis kode, terlihat bahwa kesalahan kecil seperti kondisi loop yang salah dapat menyebabkan masalah besar, sehingga penting untuk selalu memverifikasi logika sebelum eksekusi. Manfaat praktikum ini tidak hanya pada pemahaman teori, tapi juga pada keterampilan praktis seperti debugging di Eclipse. Penggunaan package dan import seperti Scanner menunjukkan pentingnya organisasi kode untuk program yang lebih kompleks di masa depan.

Akhirnya, praktikum ini mendorong mahasiswa untuk bereksperimen lebih lanjut dengan variasi kode, seperti mengubah batas loop atau menambahkan kondisi lain. Dengan demikian, pemahaman dasar ini dapat diterapkan pada proyek nyata, seperti pengembangan aplikasi sederhana. Saran untuk praktikum selanjutnya adalah menambahkan elemen error handling untuk membuat kode lebih tangguh.

DAFTAR PUSTAKA

1. Schildt, Herbert. (2019). *Java: The Complete Reference*. McGraw-Hill Education.

2. Oracle. (2023). *Java Documentation*.

3. Deitel, Paul & Deitel, Harvey. (2020). *Java How to Program*. Pearson.