

LAPORAN PRAKTIKUM
PEMROGRAMAN ALGORITMA DAN PEMROGRAMAN
“PERULANGAN FOR”



disusun Oleh:

FILZI JELILA INDA ROBBANI

2511533019

Dosen Pengampu:

Dr. WAHYUDI, S.T, M.T.

Asisten Praktikum:

JOVANTRI IMMANUEL GULO

DEPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Dengan memanjatkan puji syukur kehadiran Allah SWT, atas limpahan rahmat dan karunia-Nya sehingga praktikum Algoritma dan Pemrograman kali ini dapat saya selesaikan dengan baik dan lancar. Shalawat serta salam semoga senantiasa tercurahkan kepada junjungan kita Nabi Muhammad SAW.

Laporan ini saya susun untuk memenuhi salah satu tugas pada mata kuliah Praktikum Algoritma dan Pemrograman di Universitas Andalas. Laporan ini diharapkan dapat menambah wawasan mengenai *perulangan for* dalam bahasa pemrograman Java.

Saya menyampaikan terima kasih kepada dosen pengampu dan asisten praktikum yang telah membimbing serta memberikan arahan selama proses pembelajaran di kelas maupun di laboratorium komputer. Selain itu, saya juga berterima kasih kepada teman-teman praktikan atas bantuan dan dukungan yang diberikan, sehingga laporan ini dapat terselesaikan dengan baik.

Saya menyadari laporan ini masih jauh dari sempurna. Oleh karena itu, saya mohon maaf apabila terdapat kekurangan maupun kesalahan dalam penyusunan laporan ini. Harapan saya, semoga laporan ini dapat memberikan manfaat serta menambah pengetahuan bagi pembaca sekalian.

DAFTAR ISI

KATA PENGANTAR.....	ii
DAFTAR ISI.....	iii
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Tujuan	1
1.3 Manfaat	2
BAB II PEMBAHASAN.....	3
2.1 Langkah Kerja Praktikum.....	3
2.1.1 Membuat program pertama (perulangan For)	3
2.1.2 Membuat program kedua (perulangan for)	4
2.1.4 Membuat program keempat (perulanganFor).....	6
2.1.5 Membuat program kelima (nested for).....	8
2.1.6 Membuat program keenam (nested for)	9
2.1.7 Membuat program ketujuh (nested for)	11
BAB III PENUTUP	13
3.1 Kesimpulan.....	13
DAFTAR PUSTAKA.....	14

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Pada praktikum sebelumnya, telah dipelajari mengenai struktur percabangan dalam bahasa pemrograman Java seperti *if else* dan *switch case*. Struktur percabangan merupakan konsep dasar yang digunakan untuk menentukan alur program berdasarkan kondisi tertentu. Setelah memahami konsep percabangan, langkah selanjutnya dalam mempelajari pemrograman adalah memahami konsep *perulangan (looping)*.

Perulangan atau *looping* merupakan salah satu konsep penting dalam pemrograman yang digunakan untuk mengeksekusi suatu blok kode secara berulang sesuai dengan jumlah iterasi yang ditentukan atau hingga syarat tertentu terpenuhi. Dalam bahasa pemrograman Java, salah satu bentuk perulangan yang sering digunakan adalah *perulangan for*. *Perulangan for* sangat efektif digunakan ketika jumlah pengulangan sudah diketahui sebelumnya.

Dengan menguasai struktur *perulangan for*, kita dapat membuat program yang lebih efisien, terutama untuk proses yang bersifat berulang seperti menampilkan data berurutan, melakukan perhitungan berulang, serta memproses elemen dalam struktur data.

Oleh sebab itu, pada praktikum kali ini kita akan mempelajari penggunaan *perulangan for* dalam penyelesaian berbagai permasalahan sederhana dalam pemrograman. Praktikum ini bertujuan agar mahasiswa mampu memahami cara kerja perulangan dan dapat mengimplementasikannya dalam membangun program yang lebih sistematis, efisien, dan terstruktur.

1.2 Tujuan

Tujuan dilaksanakannya praktikum ini adalah sebagai berikut:

- a. Memahami konsep dasar perulangan (looping) dalam bahasa pemrograman Java
- b. Mempelajari cara kerja dan penggunaan struktur perulangan for
- c. Mampu menerapkan perulangan for untuk menyelesaikan masalah pemrograman yang melibatkan proses berulang
- d. Mengembangkan kemampuan logika pemrograman agar dapat menulis kode secara lebih efisien dan terstruktur

1.3 Manfaat

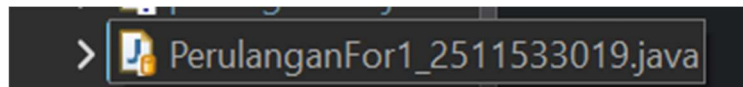
- a. Dapat memahami cara kerja perulangan for dalam pemrograman Java.
- b. Menambah bekal dasar mempelajari konsep pemrograman yang lebih lanjut.
- c. Melatih keterampilan mahasiswa dalam menuliskan kode yang efektif dan efisien.

BAB II PEMBAHASAN

2.1 Langkah Kerja Praktikum

2.1.1 Membuat program pertama (perulangan For)

- 1) Pertama buka apl Eclipse IDE for Java Developers, lalu buatlah package baru dengan mengklik kanan di folder src. Setelah itu beri nama “pekan5”. Lalu buat new class dan beri nama class “PerulanganFor1_NIM”.

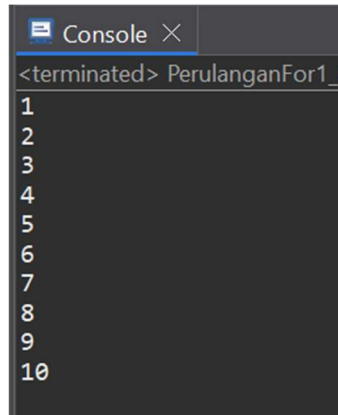


- 2) Lalu, masukkan syntax seperti pada gambar.

```
1 package pekan5_2511533019;
2
3 public class PerulanganFor1_2511533019 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         for(int i=1; i<=10; i++) {
8             System.out.println(i);
9         }
10    }
11
12 }
13
```

Pada program di atas, saya menggunakan struktur *perulangan for* untuk menampilkan angka dari 1-10. Pertama, saya mendeklarasikan variabel *i* dengan tipe data *int* di dalam *perulangan for*. Nilai awal variabel *i* adalah 1, kemudian diberikan kondisi *i <= 10* yang berarti perulangan akan terus berjalan selama nilai *i* kurang dari atau sama dengan 10. Setelah itu, nilai *i* akan ditambah satu dengan operasi *i++*. Di dalam blok perulangan, terdapat perintah *System.out.println(i);* yang berfungsi untuk mencetak nilai *i* ke layar setiap kali perulangan berjalan. Dengan demikian, program akan menampilkan angka 1-10 secara berurutan pada output.

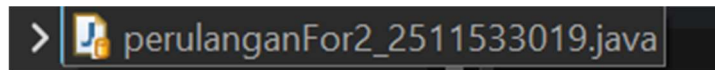
- 3) Run program dengan mengklik tombol run (bulatan hijau di bar menu) dan nanti akan muncul output dari program.



```
<terminated> PerulanganFor1_2
1
2
3
4
5
6
7
8
9
10
```

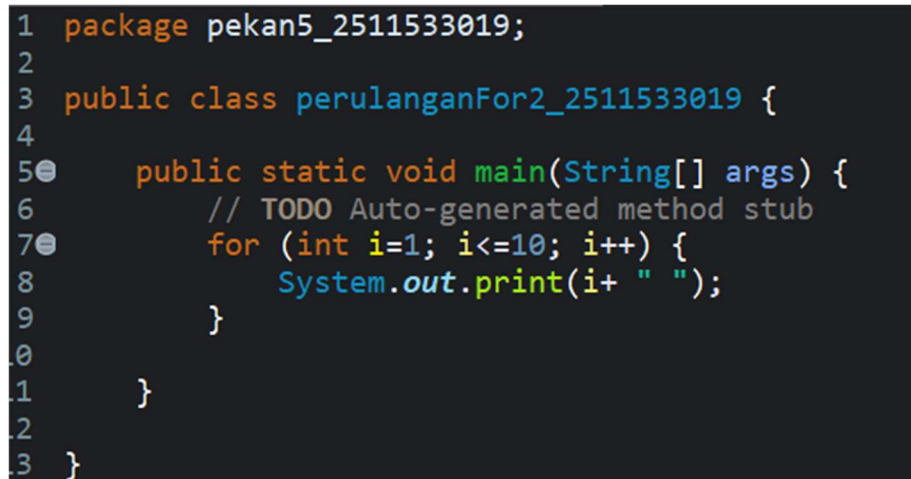
2.1.2 Membuat program kedua (perulangan for)

- 1) Buat new class dan beri nama class “PerulanganFor2_NIM”.



```
> perulanganFor2_2511533019.java
```

- 2) Masukkan syntax seperti berikut.

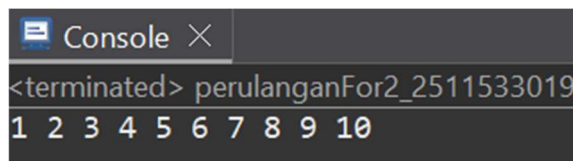


```
1 package pekan5_2511533019;
2
3 public class perulanganFor2_2511533019 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         for (int i=1; i<=10; i++) {
8             System.out.print(i+ " ");
9         }
10
11     }
12
13 }
```

Pada program di atas, saya menggunakan struktur *perulangan for* untuk menampilkan angka dari 1-10 lagi secara berurutan namun dalam satu baris. Di dalam perulangan, saya mendeklarasikan variabel *i* dengan nilai awal 1, kemudian memberikan kondisi *i* <= 10 yang berarti perulangan akan terus berjalan sampai nilai *i* mencapai 10. Setiap kali perulangan selesai dieksekusi, nilai *i* akan ditambah satu dengan operasi *i++*.

Perintah yang digunakan untuk menampilkan hasil adalah `System.out.print(i + " ");`. Berbeda dengan `println`, perintah `print` tidak membuat baris baru setelah mencetak nilai, sehingga semua angka dicetak dalam satu baris dengan spasi setelah setiap angkanya. Dengan begitu, output program akan menampilkan angka 1 -10 dalam satu baris.

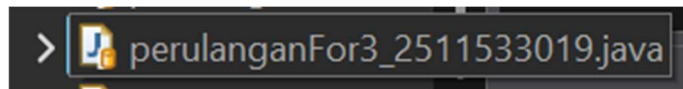
- 3) Run program dengan mengklik tombol run (bulatan hijau di bar menu) dan nanti akan muncul output dari program.



```
<terminated> perulanganFor2_2511533019
1 2 3 4 5 6 7 8 9 10
```

2.1.3 Membuat program ketiga (perulangan For)

- 1) Buat new class dan beri nama “perulanganFor3_NIM”



- 2) Masukkan syntax seperti berikut.

```
1 package pekan5_2511533019;
2
3 public class perulanganFor3_2511533019 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         int jumlah=0;
8         for (int i=1; i<=10; i++) {
9             System.out.print(i);
10            jumlah= jumlah+i;
11            if(i<10) {
12                System.out.print("+");
13            }
14        }
15        System.out.println();
16        System.out.println("Jumlah="+jumlah);
17    }
18 }
19
20 }
```

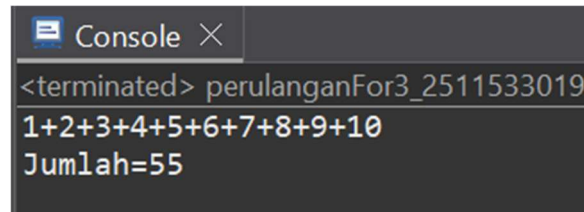

Pada program di atas, saya menggunakan *perulangan for* untuk menampilkan deretan angka dari 1-10 sekaligus menghitung total penjumlahan dari semua angka tersebut. Sebelum perulangan dimulai, saya mendeklarasikan variabel jumlah dengan nilai awal 0, karena variabel ini akan digunakan untuk menyimpan hasil penjumlahan secara bertahap.

Di dalam perulangan, setiap iterasi program akan mencetak nilai *i* menggunakan *System.out.print(i);*. Lalu, nilai *i* ditambahkan ke variabel jumlah dengan perintah *jumlah = jumlah + i;*. Dengan begitu, setiap angka dari 1- 10 akan dijumlahkan secara berurutan.

Selanjutnya, saya menggunakan struktur *if* untuk memberi tanda "+" di antara angka-angka, namun hanya sampai angka sebelum 10. Jadi jika *i* < 10, program akan mencetak simbol "+", sehingga hasil outputnya berbentuk seperti proses penjumlahan, misalnya 1+2+3+...+10.

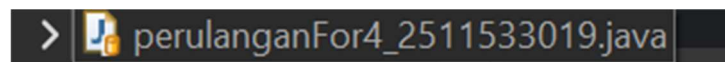
Setelah perulangan selesai, program melakukan *println()* untuk membuat baris baru, kemudian mencetak hasil total penjumlahan dengan perintah *System.out.println("Jumlah=" + jumlah);*. Dengan begitu, program menampilkan output deretan angka penjumlahan beserta hasil akhirnya.

- 3) Run program dengan mengklik tombol run (bulatan hijau di bar menu) dan nanti akan muncul output dari program.



2.1.4 Membuat program keempat (perulanganFor)

- 1) Buat new class lalu beri nama class "perulanganFor4_NIM"



- 2) Masukkan syntax seperti berikut.

```

1 package pekan5_2511533019;
2
3 import java.util.Scanner;
4
5 public class perulanganFor4_2511533019 {
6
7     public static void main(String[] args) {
8         // TODO Auto-generated method stub
9         int jumlah= 0;
10        int batas;
11        Scanner input=new Scanner(System.in);
12        System.out.print("masukkan nilai batas =");
13        batas= input.nextInt();
14        input.close();
15        for (int i=1; i<=batas; i++) {
16            System.out.print(i);
17            jumlah= jumlah+i;
18            if (i<batas) {
19                System.out.print("+");
20            }else {
21                System.out.print("=");
22            }
23        }
24        System.out.print(jumlah);
25    }
26
27 }

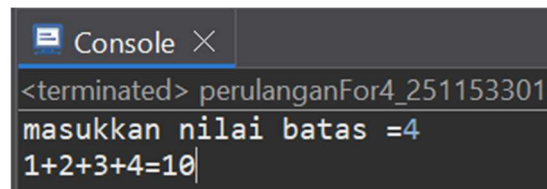
```

Untuk program ini, saya menggunakan *perulangan for* yang digabungkan dengan input dari pengguna. Pertama, saya mendeklarasikan variabel jumlah dengan nilai awal 0, lalu mendeklarasikan variabel batas yang nantinya akan diisi oleh user melalui input keyboard. Untuk membaca input tersebut, saya menggunakan syntax *Scanner*. Program meminta pengguna memasukkan nilai batas dengan perintah *System.out.print("masukkan nilai batas =");*, kemudian nilai tersebut disimpan ke dalam variabel batas menggunakan *input.nextInt()*. Setelah nilai batas diterima, objek *Scanner* ditutup menggunakan *input.close()*. Selanjutnya, *perulangan for* digunakan untuk mencetak angka dimulai dari 1 sampai angka sesuai batas yang dimasukkan user. Pada setiap iterasi, angka *i* dicetak dengan *System.out.print(i);*, lalu nilai *i* ditambahkan ke variabel jumlah untuk menghitung total penjumlahan.

Di dalam perulangan juga digunakan *if else* untuk memberikan tanda antara angka-angka. Jika nilai *i* masih lebih kecil dari batas, maka program akan mencetak tanda "+", dan jika sudah mencapai angka terakhir (sama dengan batas), program mencetak tanda "=" sebagai penutup sebelum hasil jumlah ditampilkan.

Setelah perulangan selesai, program mencetak hasil penjumlahan dengan `System.out.print(jumlah);`. Dengan demikian, output yang dihasilkan menyerupai proses penjumlahan dari angka 1 sampai batas yang diinput, diikuti dengan hasil penjumlahannya.

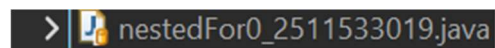
- 3) Run program, nanti user akan diminta menginputkan nilai batas dan setelah itu program akan menjalankan perintah dan output akan terlihat seperti berikut



```
<terminated> perulanganFor4_2511533019
masukkan nilai batas =4
1+2+3+4=10
```

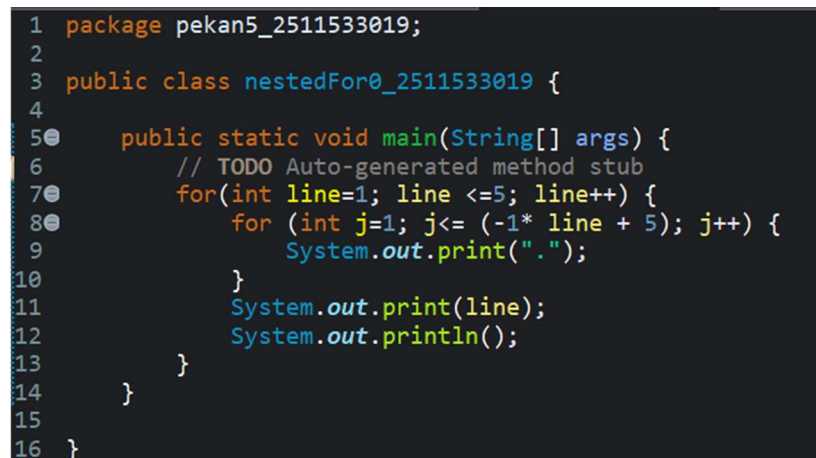
2.1.5 Membuat program kelima (nested for)

- 1) Buat new class dan beri nama class “nestedFor0_NIM”



```
> nestedFor0_2511533019.java
```

- 2) Masukkan syntax seperti berikut.



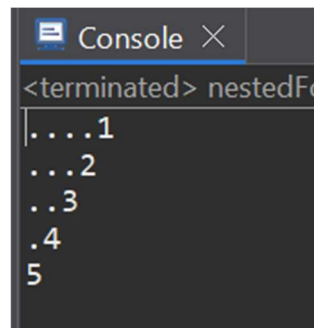
```
1 package pekan5_2511533019;
2
3 public class nestedFor0_2511533019 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         for(int line=1; line <=5; line++) {
8             for (int j=1; j<= (-1* line + 5); j++) {
9                 System.out.print(".");
10            }
11            System.out.print(line);
12            System.out.println();
13        }
14    }
15
16 }
```

Pada program di atas, saya menggunakan struktur *nested loop* untuk mencetak pola kombinasi titik dan angka. Perulangan luar menggunakan variabel *line* yang dimulai dari 1 sampai 5, sehingga program akan mencetak pola sebanyak 5 baris.

Di dalam perulangan luar, terdapat perulangan dalam menggunakan variabel *j*. Perulangan *j* berjalan dari nilai 1 hingga hasil ekspresi $(-1 * \text{line} + 5)$. Ekspresi tersebut akan menghasilkan jumlah titik yang semakin berkurang setiap barisnya. Pada setiap iterasi perulangan dalam, program mencetak karakter titik `.` menggunakan `System.out.print(".");`. Dengan demikian, tiap baris akan lebih sedikit titik dibandingkan baris sebelumnya.

Setelah perulangan dalam selesai untuk satu baris, program mencetak nilai *line* menggunakan `System.out.print(line);` sehingga angka yang ditampilkan akan bertambah setiap barisnya seiring bertambahnya nilai variabel *line*. Kemudian `System.out.println()` digunakan untuk memindahkan cursor ke baris berikutnya agar pola tercetak vertikal secara teratur.

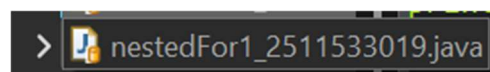
- 3) Run program , lalu akan terlihat output seperti berikut.



```
<terminated> nestedFor1
....1
...2
..3
.4
5
```

2.1.6 Membuat program keenam (nested for)

- 1) Buat new class lalu beri nama class “nestedFor1_NIM”



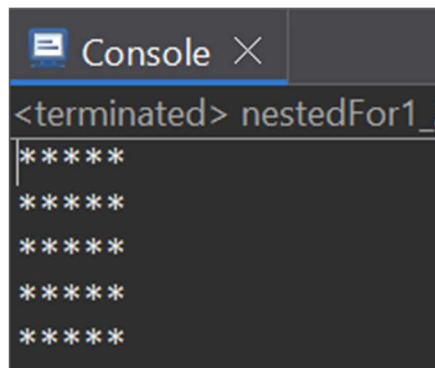
```
> nestedFor1_2511533019.java
```

- 2) Masukkan syntax seperti berikut.

```
1 package pekan5_2511533019;
2
3 public class nestedFor1_2511533019 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         for(int i=1; i<=5; i++) {
8             for(int j=1; j <=5; j++) {
9                 System.out.print("*");
10            }
11            System.out.println();
12            //to end the line
13        }
14    }
15
16 }
17
```

Pada program di atas, saya menggunakan *nested loop* (*nested for*) untuk mencetak pola bintang * berbentuk persegi. Perulangan luar menggunakan variabel i yang berjalan dari 1-5. Artinya, perulangan luar akan mengontrol jumlah baris yang akan dicetak, sehingga program akan menghasilkan 5 baris output. Di dalam perulangan luar, terdapat perulangan dalam menggunakan variabel j yang juga berjalan dari 1 sampai 5. Perulangan dalam inilah yang mengatur jumlah kolom bintang pada setiap baris. Setiap kali perulangan dalam dijalankan, program akan mencetak satu karakter * menggunakan `System.out.print("*");`. Karena perulangan dalam berjalan sebanyak 5 kali untuk setiap baris, maka setiap baris akan berisi 5 simbol *. Setelah perulangan dalam selesai untuk satu baris, program menjalankan `System.out.println();` untuk memindahkan ke baris berikutnya, sehingga pola bintang tercetak dalam format kotak 5x5. Dengan menggunakan struktur *nested loop* ini, program mampu menghasilkan pola yang teratur tanpa perlu menuliskan setiap baris secara manual.

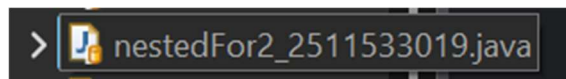
- 3) Run program, nanti akan terlihat output seperti berikut.



```
<terminated> nestedFor1_2
*****
*****
*****
*****
*****
```

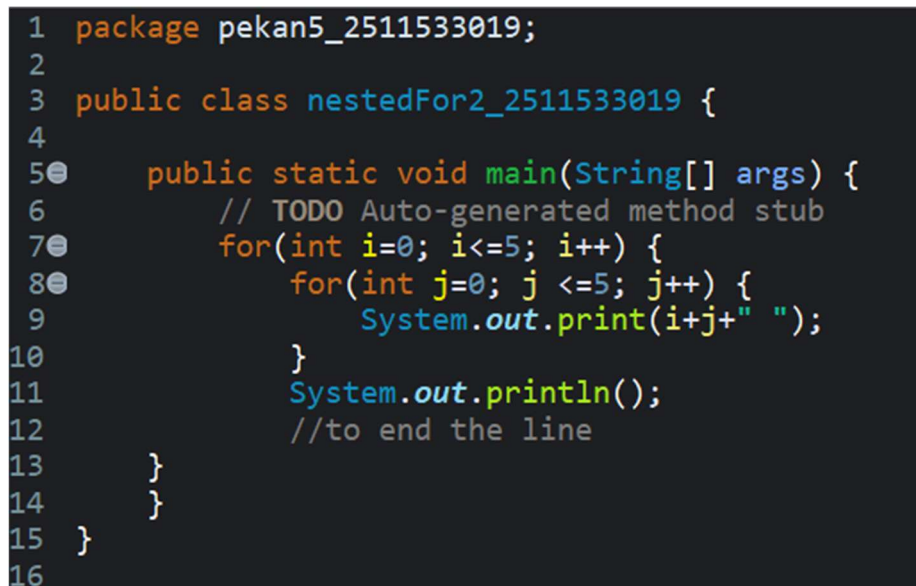
2.1.7 Membuat program ketujuh (nested for)

- 1) Buat new class dan beri nama class “nestedFor2_NIM”



```
> nestedFor2_2511533019.java
```

- 2) Masukkan syntax seperti berikut.

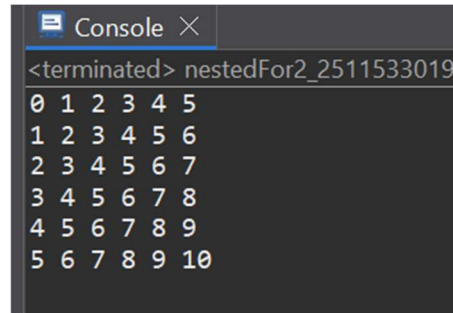


```
1 package pekan5_2511533019;
2
3 public class nestedFor2_2511533019 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         for(int i=0; i<=5; i++) {
8             for(int j=0; j <=5; j++) {
9                 System.out.print(i+j+" ");
10            }
11            System.out.println();
12            //to end the line
13        }
14    }
15 }
16
```

Pada program ini, saya menggunakan *nested loop*(*nested for*) yang terdiri dari dua perulangan for. Perulangan luar menggunakan variabel i, sedangkan perulangan dalam menggunakan variabel j. Setiap kali perulangan luar dijalankan, perulangan dalam akan berjalan sepenuhnya terlebih dahulu. Di dalam perulangan dalam, program mencetak hasil penjumlahan dari i + j diikuti dengan spasi agar hasilnya terlihat berderet secara horizontal. Setelah perulangan dalam selesai untuk satu

nilai *i*, perintah *System.out.println()*; digunakan untuk pindah ke baris berikutnya agar hasil dari perulangan berikutnya ditampilkan di baris baru. Dengan demikian, program ini akan menampilkan deretan angka hasil penjumlahan dari dua variabel *i* dan *j* dalam bentuk tabel.

- 3) Run program, hingga terlihat output seperti berikut.



```
<terminated> nestedFor2_2511533019
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
```

BAB III

PENUTUP

3.1 Kesimpulan

Berdasarkan praktikum ini, dapat disimpulkan bahwa pemahaman mengenai struktur perulangan (*looping*) dalam pemrograman Java memiliki peran penting dalam mengotomatisasi proses yang dilakukan secara berulang. Dengan menggunakan *perulangan for*, program dapat menjalankan serangkaian perintah secara berulang dalam jumlah tertentu tanpa harus menuliskan kode yang sama berulang kali. Hal ini membuat program lebih efisien, ringkas, dan mudah dipelihara.

Struktur *perulangan for* sangat cocok digunakan ketika jumlah pengulangan sudah diketahui sejak awal, misalnya mencetak angka 1 sampai 10 atau melakukan proses penjumlahan bertahap. Pada praktikum ini, *perulangan for* digunakan untuk menampilkan deret angka, menghitung jumlah nilai, serta mengatur format keluaran sesuai kebutuhan pengguna. Selain itu, penggunaan input dari user memungkinkan program menjadi lebih dinamis, di mana batas perulangan dapat ditentukan secara langsung oleh pengguna.

Praktikum ini juga mempelajari *nested for* (*didalam perulangan ada perulangan lagi/perulangan bersarang*). Dengan *nested for*, program dapat membentuk pola tertentu seperti pola bintang maupun susunan karakter lainnya. *Nested For* ini memungkinkan satu perulangan menjalankan perulangan lain di dalamnya, sehingga menghasilkan kombinasi logika dan pola yang lebih kompleks.

Dengan memahami konsep perulangan tunggal hingga perulangan bersarang, kita dapat membangun program yang lebih sistematis, efisien, dan mampu menangani proses berulang secara optimal. Penguasaan teknik looping juga menjadi dasar penting untuk pemrograman yang lebih lanjut, seperti algoritma, struktur data, dan pemrosesan data dalam skala besar.

DAFTAR PUSTAKA

- [1] Wahyudi, “*Perulangan For di Java*” PowerPoint slides, Mata Kuliah Algoritma dan pemrograman, Universitas Andalas, Padang, 2025.
- [2] Muzanni, “*Perulangan Pada Java*,” MalasNgoding.com, 24 January 2017. [Online]. Tersedia: <https://www.malasngoding.com/perulangan-pada-java/>
- [3] Muhardian, A., “*Belajar Java: Memahami 2 Jenis Perulangan dalam Java*,” PetaniKode.com, 03 Jan 2016. [Online]. Tersedia: <https://www.petanikode.com/java-perulangan/>
- [4] Oracle, “*The for Statement (Java SE Documentation)*,” Oracle Docs, [Online]. Tersedia: <https://docs.oracle.com/javase/tutorial/java/nutsandbolts/for.html>