

**LAPORAN PRAKTIKUM**  
**AGORITMA PEMROGRAMAN**  
**“PERULANGAN WHILE dan DO WHILE”**

**DISUSUN OLEH:**

**NOFRI ILHAM**

**2511531013**

**DOSEN PENGAMPU:**

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

**ASISTEN PRAKTIKUM:**

**MUHAMMAD ZAKI AL HAFIZ**



**DEPARTEMEN INFORMATIKA**  
**FAKULTAS TEKNOLOGI INFORMASI**  
**UNIVERSITAS ANDALAS**

**2025**

## KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Puji syukur kehadiran Allah SWT karena atas rahmat dan karunia-Nya penulis dapat menyelesaikan laporan praktikum dengan judul “Perulangan While dan Do-While” tepat pada waktunya. Laporan ini disusun sebagai salah satu tugas dalam mata kuliah Pemrograman Dasar guna memahami konsep dan penerapan struktur perulangan (looping) pada bahasa pemrograman Java.

Melalui praktikum ini, penulis mempelajari bagaimana cara kerja perulangan while dan do-while, serta perbedaan utama di antara keduanya dalam hal kondisi dan pelaksanaan perintah. Diharapkan laporan ini dapat memberikan pemahaman yang lebih mendalam mengenai logika perulangan dan penerapannya dalam pembuatan program yang efisien.

Penulis menyampaikan terima kasih kepada dosen pengampu dan asisten praktikum yang telah memberikan bimbingan selama kegiatan praktikum berlangsung. Penulis juga menyadari bahwa laporan ini masih jauh dari sempurna, sehingga kritik dan saran yang membangun sangat diharapkan untuk perbaikan di masa mendatang.

Padang, 7 November 2025

Nofri Ilham

## DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI .....	ii
BAB I PENDAHULUAN .....	1
1.1 Latar Belakang .....	1
1.2 Tujuan .....	1
1.3 Manfaat .....	1
BAB II PEMBAHASAN .....	3
2.1 Perulangan While .....	3
2.2 DO While .....	3
2.3 Aplikasi Perulangan While dan DO While .....	4
2.3.1 Kode Program Perulangan While .....	4
2.3.2 Kode Program Lempar Dadu .....	6
2.3.3 Kode Program Game Penjumlahan .....	8
2.3.4 Kode Program Sentinel Loop.....	10
2.3.5 Kode Program Do While .....	11
BAB III KESIMPULAN .....	13
3.1 Kesimpulan .....	13
3.2 Saran .....	13
DAFTAR PUSTAKA.....	15

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Dalam pemrograman, proses perulangan merupakan salah satu konsep dasar yang sangat penting karena memungkinkan program menjalankan suatu perintah secara berulang tanpa harus menuliskan kode yang sama berkali-kali. Bahasa pemrograman Java menyediakan beberapa jenis struktur perulangan, di antaranya adalah while dan do-while loop.

Kedua jenis perulangan ini sering digunakan untuk mengatasi permasalahan yang memerlukan pengulangan dengan syarat tertentu, seperti pengolahan data berulang atau pengulangan hingga kondisi tertentu terpenuhi. Oleh karena itu, pemahaman yang baik tentang cara kerja dan perbedaan antara while dan do-while sangat dibutuhkan agar programmer dapat menulis kode yang efisien, efektif, dan mudah dipahami.

### **1.2 Tujuan**

Tujuan dari praktikum Perulangan While dan Do-While ini adalah:

1. Memahami konsep dasar perulangan dalam pemrograman Java.
2. Mengetahui perbedaan antara struktur while dan do-while.
3. Menerapkan perulangan while dan do-while dalam menyelesaikan masalah pemrograman.
4. Melatih kemampuan logika dalam menyusun algoritma yang menggunakan struktur perulangan.

### **1.3 Manfaat**

Adapun manfaat yang dapat diperoleh dari praktikum ini antara lain:

1. Menambah pemahaman tentang konsep perulangan dan penerapannya dalam program.
2. Meningkatkan kemampuan analisis logika dan pemecahan masalah melalui implementasi perulangan.

3. Menjadi dasar dalam mempelajari struktur kontrol yang lebih kompleks pada pemrograman Java.
4. Membantu mahasiswa dalam membuat program yang lebih efisien dan mudah dikembangkan.

## BAB II PEMBAHASAN

### 2.1 Perulangan While

Perulangan while dalam Java adalah pernyataan alur kontrol yang memungkinkan kode dieksekusi berulang kali berdasarkan kondisi boolean yang diberikan. Perulangan akan terus dieksekusi selama kondisi yang ditentukan bernilai true. Hal ini sangat berguna ketika jumlah iterasi tidak diketahui sebelumnya.

Berikut format dasar struktur perulangan **WHILE** dalam bahasa Java:

1. start;
2. **while** (condition)
3. {
4. // kode program
5. // kode program
6. increment;
7. }

Di bagian **start** biasanya ditulis perintah inisialisasi variabel counter, misalnya **i = 0**. Di bagian **condition** terdapat kondisi yang harus dipenuhi agar perulangan berjalan, misalnya **i < 5**. Kemudian perintah **increment** harus berada di dalam block perulangan agar bisa menaikkan nilai variabel counter, misalnya dengan perintah **i++**.

### 2.2 DO While

Perulangan **DO WHILE** merupakan modifikasi dari perulangan **WHILE**, yakni dengan memindahkan posisi pemeriksaan kondisi ke akhir perulangan. Artinya, lakukan dahulu sebuah perulangan, baru periksa apakah kondisi *variabel counter* sudah terpenuhi atau belum di akhir perulangan.

Berikut format dasar struktur perulangan **DO WHILE** dalam bahasa Java:

1. start;
2. do
3. {
4. // kode program

5. // kode program
6. increment;
7. }
8. while (condition)

Sama seperti perulangan WHILE, di bagian start biasanya terdapat perintah inisialisasi variabel counter, misalnya  $i = 0$ . Kemudian di dalam block do ditulis kode program yang akan di ulang, tidak lupa sebuah perintah untuk menaikkan nilai variabel counter, misalnya dengan perintah  $i++$ .

Di bagian paling bawah, terdapat perintah while (condition). Di sinilah kondisi perulangan akan diperiksa. Selama kondisi ini menghasilkan nilai true, maka perulangan akan lanjut ke iterasi berikutnya.

## 2.3 Aplikasi Perulangan While dan DO While

Pada praktikum pekan ke-6, mahasiswa diberikan instruksi untuk menyalin dan menjalankan kode program perulangan while dan do-while yang telah disiapkan oleh dosen. Kegiatan ini bertujuan sebagai latihan sekaligus contoh penerapan struktur perulangan dalam bahasa pemrograman Java. Berikut merupakan kode program yang digunakan pada praktikum pekan ini.

### 2.3.1 Kode Program Perulangan While

```

1 package Pekan6_2511531013;
2
3 import java.util.Scanner;
4
5 public class PerulanganWhile1_2511531013 {
6     public static void main(String[] args) {
7
8         int counter=0;
9         String jawab;
10        boolean running=true;
11        //deklarasi scanner
12        Scanner scan= new Scanner(System.in);
13        while (running) {
14            counter++;
15            System.out.println("Jumlah = "+ counter);
16            System.out.println("Apakah Lanjut ( ya / tidak?)");
17            jawab=scan.nextLine();
18            //cek jawaban = tidak, perulangan berhenti
19            if (jawab.equalsIgnoreCase("Tidak")) {
20                running=false;
21            }
22        }
23        System.out.println("Anda sudah melakukan perulangan sebanyak "+ counter + " kali");
24    }
25 }
26
27 }

```

Kode Program 2.1

### Langkah-langkah Program

1. Deklarasikan variabel counter, jawab, dan running.
2. Buat objek Scanner untuk membaca input dari pengguna.
3. Jalankan perulangan while (running).
4. Setiap perulangan, tambahkan nilai counter dan tampilkan jumlahnya.
5. Tanyakan ke pengguna apakah ingin melanjutkan perulangan.
6. Jika pengguna menjawab "tidak", ubah running menjadi false agar perulangan berhenti.
7. Setelah keluar dari perulangan, tampilkan total jumlah perulangan yang telah dilakukan.

Program di atas merupakan contoh penerapan struktur perulangan while dalam bahasa pemrograman Java. Program ini bertujuan untuk menghitung jumlah perulangan berdasarkan input dari pengguna. Variabel counter digunakan untuk menghitung banyaknya perulangan, sedangkan variabel jawab menyimpan input berupa jawaban “ya” atau “tidak”. Program juga menggunakan variabel running bertipe boolean sebagai pengendali jalannya perulangan. Selama nilai running bernilai true, perulangan akan terus berjalan dan menampilkan jumlah perulangan ke layar.

Di dalam perulangan, program akan menanyakan kepada pengguna apakah ingin melanjutkan atau tidak. Jika pengguna mengetik “ya”, maka perulangan akan berlanjut. Namun, jika pengguna mengetik “tidak”, maka kondisi if (jawab.equalsIgnoreCase("tidak")) akan terpenuhi dan variabel running diubah menjadi false sehingga perulangan berhenti. Setelah keluar dari perulangan, program menampilkan pesan akhir berupa jumlah total pengulangan yang telah dilakukan.

### 2.3.2 Kode Program Lempar Dadu

```

1 package Pekan6_2511531013;
2 import java.util.Random;
3 public class LemparDadu_2511531013 {
4
5     public static void main(String[] args) {
6         Random rand = new Random();
7         int tries =0;
8         int sum =0;
9         while(sum!=7) {
10             //roll the dice one
11             int dadu1 = rand.nextInt(6)+1;
12             int dadu2 = rand.nextInt(6)+1;
13             sum= dadu1+dadu2;
14             System.out.println(dadu1 + " + " + dadu2 + " = "+ sum);
15             tries++;
16
17         }
18         System.out.println("You won after"+tries +" Tries!");
19
20     }
21
22 }

```

Kode Program 2.2

#### Langkah-langkah program

1. Buat objek Random untuk menghasilkan angka acak.
2. Deklarasikan variabel tries untuk menghitung jumlah percobaan, dan sum untuk menampung hasil penjumlahan dadu.
3. Jalankan perulangan while (sum != 7).
4. Di dalam perulangan, hasilkan dua angka acak antara 1–6 (dadu1 dan dadu2).
5. Jumlahkan kedua dadu dan simpan ke variabel sum.
6. Tampilkan hasil setiap percobaan ke layar, lalu tambahkan nilai tries sebanyak 1.
7. Jika jumlah dadu sama dengan 7, perulangan berhenti dan tampilkan pesan kemenangan dengan jumlah percobaan yang dilakukan.

Program di atas menggunakan struktur perulangan while untuk mensimulasikan permainan lempar dadu. Tujuan program ini adalah untuk melempar dua buah dadu secara acak hingga jumlah keduanya bernilai 7. Selama hasil penjumlahan belum sama dengan 7, program akan terus mengulang proses pelemparan. Setiap kali perulangan terjadi, nilai dari dua dadu dihasilkan secara acak menggunakan kelas Random dengan perintah `rand.nextInt(6) + 1`, yang berarti menghasilkan angka acak antara 1 sampai 6.

Di dalam perulangan, program menjumlahkan kedua nilai dadu (`dadu1 + dadu2`) dan menyimpannya ke variabel `sum`. Hasil lemparan setiap percobaan akan ditampilkan ke layar menggunakan `System.out.println()`, lalu variabel `tries` akan bertambah satu setiap kali dadu dilempar. Perulangan while (`sum != 7`) akan terus berjalan hingga total kedua dadu sama dengan 7, setelah itu perulangan berhenti dan program menampilkan pesan “You won after X tries!”, yang menunjukkan berapa kali percobaan dibutuhkan sampai mendapatkan jumlah 7.

### 2.3.3 Kode Program Game Penjumlahan

```

1 package Pekan6_2511531013;
2 import java.util.*;
3 public class GamePenjumlahan_2511531013 {
4
5     public static void main(String[] args) {
6         Scanner console = new Scanner(System.in);
7         Random rand = new Random();
8         //play until user get 3 wrong
9         int points=0;
10        int wrong=0;
11        while(wrong<3) {
12            int result= play(console,rand); // play one game
13            if (result >0) {
14                points++;
15            }else {
16                wrong++;
17            }
18        }
19        System.out.println("You earned"+ points + " Total points.");
20    }
21    // membuat soal penjumlahan dan ditampilkan ke user
22    public static int play(Scanner console,Random rand) {
23        //print the operands being added, and sum them
24        int operands = rand.nextInt(4)+2;
25        int sum = rand.nextInt(10)+1;
26        System.out.print(sum);
27        for (int i=2;i <=operands;i++) {
28            int n = rand.nextInt(10)+1;
29            sum += n;
30            System.out.print(" + " + n);
31        }
32        System.out.print(" = ");
33
34        //read user's guess and report whether it was correct
35        int guess=console.nextInt();
36        if (guess==sum) {
37            return 1;
38        }else {
39            System.out.println("Wrong! The Answer was " + sum);
40            return 0;
41        }
42    }
43
44 }

```

Kode Program 2.3

#### Langkah-langkah program

1. Buat objek Scanner dan Random.
2. Deklarasi variabel points = 0 dan wrong = 0.
3. Jalankan loop while (wrong < 3).
4. Panggil fungsi play(console, rand) untuk membuat satu soal penjumlahan acak.

5. Jika jawaban benar ( $\text{result} > 0$ ), tambah  $\text{points}++$ ; jika salah, tambah  $\text{wrong}++$ .
6. Ulangi hingga  $\text{wrong} == 3$ .
7. Setelah loop selesai, tampilkan total poin ke layar.
8. Di fungsi  $\text{play}()$ : buat beberapa angka acak, tampilkan soal, baca input, lalu cek apakah jawaban benar atau salah.

Program di atas merupakan permainan sederhana untuk menguji kemampuan penjumlahan pengguna. Program menggunakan perulangan `while` agar permainan terus berjalan selama jumlah jawaban salah belum mencapai tiga kali. Setiap kali perulangan berjalan, program akan memanggil metode  $\text{play}()$  untuk menghasilkan soal penjumlahan acak. Jika pengguna menjawab benar, poin (`points`) akan bertambah satu. Namun jika salah, variabel `wrong` akan bertambah satu. Ketika jumlah kesalahan sudah mencapai tiga kali, permainan otomatis berhenti dan program menampilkan total poin yang berhasil dikumpulkan.

Di dalam metode  $\text{play}()$ , program menggunakan kelas `Random` untuk membuat soal dengan beberapa angka acak yang harus dijumlahkan. Banyaknya angka yang dijumlahkan ditentukan oleh variabel `operands`, sedangkan nilai awal dan angka tambahan dihasilkan secara acak antara 1 sampai 10. Program menampilkan soal, lalu membaca jawaban pengguna menggunakan `Scanner`. Jika jawaban pengguna sesuai dengan hasil penjumlahan (`sum`), maka fungsi mengembalikan nilai 1 (benar), sedangkan jika salah akan menampilkan jawaban yang benar dan mengembalikan nilai 0.

### 2.3.4 Kode Program Sentinel Loop

```

1 package Pekan6_2511531013;
2 import java.util.Scanner;
3 public class SentinelLoop_2511531013 {
4
5     public static void main(String[] args) {
6         Scanner console= new Scanner(System.in);
7         int sum=0;
8         int number=12; // * dummy value*, anything but 0
9
10        while(number!=0) {
11            System.out.print("Masukkan angka (0 untuk keluar):");
12            number= console.nextInt();
13            sum=sum+number;
14        }
15        System.out.println("Totalnya adalah;" + sum);
16
17    }
18
19 }

```

Kode Program 2.4

#### Langkah-langkah program

1. Buat objek Scanner untuk input.
2. Inisialisasi sum = 0 dan number = 12 (dummy value).
3. Jalankan loop while (number != 0).
4. Tampilkan teks: "Masukkan angka (0 untuk keluar):".
5. Baca input user ke variabel number.
6. Tambahkan number ke sum.
7. Jika user memasukkan 0 ,maka loop berhenti.
8. Cetak hasil akhir: "Totalnya adalah: " + sum.

Program diatas menjumlahkan sekumpulan angka yang dimasukkan oleh pengguna sampai pengguna mengetik angka 0. Angka 0 di sini berfungsi sebagai nilai sentinel, yaitu tanda untuk menghentikan perulangan.

Program menggunakan perulangan while dengan kondisi number != 0, artinya selama nilai yang dimasukkan bukan 0, proses akan terus berulang. Setiap input dari pengguna ditambahkan ke variabel sum. Saat pengguna memasukkan angka 0, loop berhenti dan program

menampilkan hasil total penjumlahan seluruh angka yang sudah dimasukkan.

### 2.3.5 Kode Program Do While

```

1 package Pekan6_2511531013;
2 import java.util.Scanner;
3 public class doWhile1_2511531013 {
4
5     public static void main(String[] args) {
6         Scanner console= new Scanner(System.in);
7         String phrase;
8         do {
9             System.out.print("Input Password: ");
10            phrase = console.next();
11        } while(!phrase.equals("abcd"));
12
13
14
15
16    }
17
18 }

```

Kode Program 2.5

#### Langkah-langkah program

1. Import Scanner untuk mengambil input dari user.
2. Deklarasikan variabel phrase bertipe String.
3. Jalankan perulangan do { ... } while(...).
4. Di dalam loop, tampilkan teks "Input Password:".
5. Baca input user dan simpan ke variabel phrase.
6. Cek kondisi !phrase.equals("abcd").
7. Kalau input bukan "abcd", ulangi lagi.
8. Kalau input "abcd", loop berhenti dan program selesai.

Program doWhile1 diatas meminta pengguna memasukkan password hingga input yang diberikan benar, yaitu "abcd". Program menggunakan perulangan do...while, yang berarti isi loop akan

dijalankan setidaknya satu kali, baru kemudian kondisi dicek di bagian akhir.

Dalam program ini, pengguna akan terus diminta mengetik "Input Password:" sampai kata yang dimasukkan sama dengan string "abcd". Selama kondisi `!phrase.equals("abcd")` (tidak sama dengan "abcd") bernilai true, maka perulangan akan terus berjalan. Begitu pengguna mengetik password yang benar, perulangan berhenti dan program selesai.

## BAB III KESIMPULAN

### 3.1 Kesimpulan

Dari hasil praktikum mengenai **perulangan while dan do-while** dalam bahasa pemrograman Java, dapat disimpulkan bahwa:

1. **Perulangan while** akan menjalankan blok kode selama kondisi bernilai *true*, dan kondisi diperiksa **sebelum** perulangan dijalankan. Oleh karena itu, jika kondisi awal bernilai *false*, maka blok kode tidak akan dijalankan sama sekali.
2. **Perulangan do-while** melakukan hal serupa, namun dengan perbedaan utama bahwa kondisi diperiksa **setelah** perulangan dijalankan, sehingga blok kode akan **selalu dijalankan minimal satu kali**.
3. Melalui berbagai contoh program seperti simulasi lempar dadu, game penjumlahan, dan sentinel loop, mahasiswa dapat memahami bahwa struktur perulangan sangat berguna untuk menyelesaikan masalah yang membutuhkan proses berulang dengan kondisi tertentu.
4. Praktikum ini membantu mahasiswa meningkatkan kemampuan dalam **menganalisis logika, mengontrol alur program, serta menulis kode yang lebih efisien dan fleksibel** menggunakan struktur kontrol perulangan.

### 3.2 Saran

1. Mahasiswa sebaiknya terus berlatih membuat variasi program yang menggunakan struktur perulangan agar lebih memahami perbedaan dan keunggulan masing-masing jenis loop.
2. Disarankan untuk mengkombinasikan perulangan dengan **struktur keputusan (if-else)** dan **array** agar dapat membuat program yang lebih kompleks dan dinamis.
3. Dalam praktikum berikutnya, sebaiknya diberikan contoh penerapan perulangan dalam kasus nyata seperti pengolahan data atau validasi input untuk memperkuat pemahaman konsep.

4. Dokumentasi kode dan pemberian komentar pada setiap bagian program perlu ditingkatkan agar program lebih mudah dipahami dan dikembangkan oleh orang lain.

## DAFTAR PUSTAKA

- [1] DataCamp, “Java While Loop,” [Daring]. Tersedia pada: <https://www.datacamp.com/doc/java/java-while-loop> [Diakses: 8-Nov-2025].
- [2] Duniaikom, “Tutorial Belajar Java: Perulangan While (Bahasa Java),” [Daring]. Tersedia pada: <https://www.duniaikom.com/tutorial-belajar-java-perulangan-while-bahasa-java/> [Diakses: 8-Nov-2025].
- [3] Duniaikom, “Tutorial Belajar Java: Perulangan Do-While (Bahasa Java),” [Daring]. Tersedia pada: <https://www.duniaikom.com/tutorial-belajar-java-perulangan-do-while-bahasa-java/> [Diakses: 8-Nov-2025].