

**LAPORAN PRATIKUM
PEMROGRAMAN ALGORITMA PEMROGRAMAN**

**LAPORAN PRAKTIKUM PEMROGRAMAN DASAR JAVA: PENERAPAN
PERULANGAN WHILE DAN DO-WHILE**

disusun oleh:

Khairunnisa M.

NIM 2511532005

Dosen Pengampu:

DR. Wahyudi, S. T., M. T.

Asisten Pratikum:

Aufan Taufiqurrahman



DEPARTEMEN INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Puji syukur penulis penjatkan ke hadirat Tuhan Yang Maha Esa, karena atas rahmat dan karunia-Nya, penulis dapat menyelesaikan laporan praktikum ini dengan baik. Laporan ini disusun untuk memenuhi salah satu tugas pada mata kuliah Pemrograman Dasar Java dengan topik pembahasan Perulangan While dan Do-While. Melalui praktikum ini, penulis belajar memahami bagaimana struktur perulangan bekerja dalam Bahasa pemrograman Java, baik perulangan tunggal maupun perulangan bersarang. Selain itu, praktikum ini juga membantu penulis dalam melatih logika berpikir secara terstruktur, efisien, dan sistematis dalam menyelesaikan permasalahan pemrograman.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, kritik dan saran yang membangun sangat diharapkan demi perbaikan di masa mendatang. Akhir kata, penulis mengucapkan terima kasih kepada dosen pengampu, asistensi praktikum, serta rekan-rekan yang telah membantu dalam proses pelaksanaan praktikum ini. Semoga laporan ini dapat memberikan manfaat bagi pembaca dan menjadi referensi untuk memahami konsep perulangan dalam Bahasa Java.

Padang, 3 November 2025

Khairunnisa M.

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI	ii
BAB I	1
PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Tujuan Praktikum	1
1.3 Manfaat Praktikum	1
BAB II.....	3
PEMBAHASAN.....	3
2.1. Do While.....	3
2.2. Game Penjumlahan	5
2.3. Lempar Dadu	6
2.4. Perulangan While 1	7
2.5. Sentinel Loop	9
BAB III.....	11
KESIMPULAN	11
DAFTAR PUSTAKA.....	12

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam dunia pemrograman, perulangan merupakan salah satu konsep dasar yang sangat penting karena memungkinkan program untuk menjalankan suatu perintah secara berulang tanpa harus menulis kode yang sama berkali-kali. Struktur perulangan membantu efisiensi dalam menulis program, menghemat waktu, serta mempermudah pengolahan data yang bersifat berulang. Di antara berbagai jenis perulangan yang ada, *while* dan *do while* menjadi dua struktur utama yang sering karena keduanya memberikan fleksibilitas dalam mengatur kondisi perulangan.

Perulangan *while* digunakan ketika suatu proses perlu dijalankan selama kondisi tertentu terpenuhi, di mana kondisi akan diperiksa sebelum blok perintah dijalankan. Sebaliknya, *do while* mengeksekusi blok perintah terlebih dahulu sebelum memeriksa kondisi, sehingga perulangan ini akan selalu dijalankan minimal satu kali. Melalui praktikum ini, mahasiswa diharapkan dapat memahami perbedaan mendasar antara *while* dan *do while*, serta mampu mengimplementasikannya dalam menyusuk logika program yang efisien dan sesuai kebutuhan.

1.2 Tujuan Praktikum

1. Memahami konsep dasar struktur perulangan (looping) dalam pemrograman.
2. Mempelajari cara kerja dan perbedaan antara perulangan *while* dan *do while*
3. Mampu menuliskan dan mengimplementasikan program menggunakan struktur *while* dan *do while* dengan benar.
4. Mengetahui kondisi dan situasi yang tepat untuk menggunakan *while* maupun *do while* dalam penyelesaian masalah pemrograman.
5. Melatih kemampuan logika dan berpikir sistematis dalam menyusun alur program yang efisien.

1.3 Manfaat Praktikum

1. Mahasiswa dapat memahami dan menerapkan konsep perulangan dalam pembuatan program.
2. Meningkatkan kemampuan analisis terhadap penggunaan struktur control dalam pemrograman.
3. Membantu mahasiswa dalam menulis kode program yang lebih efisien dan mudah dipelihara.

4. Memberikan pengalaman langsung dalam menguji perbedaan hasil antara penggunaan *while* dan *do while*.
5. Menjadi dasar dalam mempelajari konsep perulangan yang lebih kompleks pada pemrograman lanjutan.

BAB II

PEMBAHASAN

Perulangan *while* dan *do while* merupakan struktur control dalam pemrograman yang digunakan untuk menjalankan satu atau beberapa perintah secara berulang seama kondisi tertentu terpenuhi. Pada perulangan *while*, kondisi diperiksa terlebih dahulu sebelum blok perintah dijalankan, sehingga jika kondisi awal bernilai salah, maka perulangan tidak akan dijalankan sama sekali. Sebaliknya, pada perulangan *do while*, blok perintah dieksekusi terlebih dahulu baru kemudian kondisi diperiksa, sehingga perulangan ini akan selalu dijalankan minimal satu kali meskipun kondisi bernilai salah. Kedua jenis perulangan ini sangat berguna untuk membuat program yang dinamis dan efisien, terutama ketika jumlah pengulangan tidak diketahui secara pasti sejak awal. Untuk memahami perbedaan dan penerapan keduanya secara lebih jelas, dapat dilihat pada praktikum yang dilakukan saat kelas pemrograman algoritma yang akan dijelaskan di bawah ini.

2.1. Do While

Program ini dirancang untuk meminta pengguna memasukkan kata sandi hingga input yang diberikan sesuai dengan nilai yang ditentukan, yaitu “abcd”. Melalui contoh ini, dapat dilihat bagaimana perulangan *do while* memastikan bahwa blok perintah dijalankan minimal satu kali sebelum kondisi diperiksa, sehingga pengguna akan selalu diminta untuk memasukkan data terlebih dahulu sebelum program memutuskan untuk mengulangi atau menghentikan proses.

```

1 package Pekan6_2511532005;
2
3 import java.util.Scanner;
4
5 public class doWhile_2511532005 {
6     public static void main(String[] args) {
7         Scanner console = new Scanner(System.in);
8
9         String phrase;
10        do {
11            System.out.print("Input Password: ");
12            phrase = console.next();
13        } while (!phrase.equals("abcd"));
14
15    }
16
17
18
19 }

```

Gambar 2.1. Program Java dengan do While.

Penjelasan tiap bagian :

1. *Scanner console = new Scanner(System.in);*

Ini digunakan untuk membuat objek console yang digunakan untuk mengambil input dari pengguna.

2. *String phrase;*

Mendeklarasikan variabel phrase bertipe *String* untuk menyimpan kata sandii yang diinput pengguna.

3. *do {...} while (!phrase.equals("abcs"));*

Struktur perulangan *do while* akan mengeksekusi perintah di dalam blok {} terlebih dahulu, kemudian memeriksa kondisi di bagian *while*.

Langkah kerja :

1. Program dijalankan.
2. Sistem menampilkan pesan: "Input Password: ".
3. Pengguna mengetikkan kata apa pun, misalnya xyz.
4. Program memeriksa apakah phrase sama dengan "abcd".

5. Jika tidak sama, maka perulangan akan berjalan lagi, dan pesan “Input Password:” muncul kembali.
6. Jika pengguna akhirnya mengetik “abcd”, kondisi menjadi salah (false) dan perulangan berhenti.
7. Program selesai di eksekusi.

Program ini menunjukkan perbedaan mendasar antara struktur *while* dan *do while*. Pada *do while*, blok perintah selalu dijalankan minimal satu kali, bahkan jika kondisi awal sudah salah. Secara keseluruhan, program ini sederhana namun efektif untuk memahami cara kerja perulangan *do while* dan penerapannya dalam proses validasi input.

2.2. Game Penjumlahan

Program ini merupakan permainan sederhana yang menggunakan perulangan *while* untuk memberikan soal penjumlahan acak kepada pengguna. Setiap jawaban benar akan menambah poin, sedangkan jawaban salah akan menambah jumlah kesalahan hingga mencapai tiga kali. Program berakhir setelah pengguna melakukan tiga kesalahan, lalu menampilkan total poin yang diperoleh.

```

1 package Pekan6_2511532005;
2 import java.util.Random;
3 import java.util.Scanner;
4
5 public class GamePenjumlahan_2511532005 {
6     public static void main(String[] args) {
7         Scanner console = new Scanner(System.in);
8         Random rand = new Random();
9
10        int points = 0;
11        int wrong = 0;
12        while (wrong < 3) {
13            int result = play(console, rand);
14            if (result > 0) {
15                points++;
16            } else {
17                wrong++;
18            }
19        }
20        System.out.println("You earned " + points + " total points.");
21    }
22
23    public static int play(Scanner console, Random rand) {
24        int operands = rand.nextInt(4) + 2;
25        int sum = rand.nextInt(10) + 1;
26        System.out.print(sum);
27
28        for (int i = 2; i <= operands; i++) {
29            int n = rand.nextInt(10) + 1;
30            sum += n;
31            System.out.print(" + " + n);
32        }
33        System.out.print(" = ");
34
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 }

```


Gambar 2.2. Program Java dengan Game Penjumlahan.

Langkah kerja:

1. Membuat file baru bernama `PerulanganFor2.java`
2. Menuliskan kode program untuk menampilkan deret angka dalam satu baris menggunakan `system.out.print`.
3. Menjalankan program dan mengamati hasil keluaran di layar.
4. Menyimpan hasil untuk dibahas pada laporan.

Program ini menunjukkan penerapan konsep *while loop*, fungsi, dan penggunaan objek *random* dalam Java untuk membuat permainan edukatif interaktif. Fungsi *play()* menghasilkan sejumlah bilangan acak untuk dijumlahkan dan meminta pengguna menembak hasilnya. Struktur *control if-else* digunakan untuk menentukan apakah jawaban pengguna benar atau salah. Logika perulangan *while* memastikan permainan berjalan terus hingga pengguna melakukan tiga kesalahan, sehingga konsep pengulangan bersyarat dapat dipahami dengan jelas melalui simulasi permainan ini.

2.3. Lempar Dadu

Program ini merupakan simulasi permainan melempar dua buah dadu secara acak menggunakan perulangan *while* pada bahasa Java. Program ini akan terus melempar kedua dadu hingga jumlah angka yang muncul sama dengan 7. Setiap kali dadu dilempar, hasilnya ditampilkan di layar, dan Ketika kombinasi yang menghasilkan angka 7 tercapai, program menampilkan jumlah percobaan yang dibutuhkan untuk mencapainya.

```

1 package Pekan6_2511532005;
2 import java.util.Random;
3
4 public class LemparDadu_2511532005 {
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
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         System.out.println("You won after " + tries + " tries! ");
18     }
19 }
20 }
21

```

Gambar 2.3. Program Java dengan Lempar Dadu.

Langkah kerja:

1. Membuat file baru dengan nama LemparDadu.Java
2. Impor library `java.util.Random` untuk menghasilkan angka acak.
3. Deklarasi variabel `tries` untuk menghitung jumlah percobaan dan `sum` untuk menyimpan jumlah dua dadu.
4. Gunakan perulangan *while* (`sum != 7`) agar program terus berjalan sampai hasil penjumlahan kedua dadu bernilai 7.
5. Di dalam perulangan, hasilkan dua angka acak dari 1 sampai 6 untuk dadu1 dadu2.
6. Hitung jumlahnya (`sum = dadu1 + dadu2`) lalu tampilkan hasilnya di layar menggunakan `System.out.println`.
7. Tambahkan nilai `tries` setiap kali dadu dilempar.
8. Setelah jumlah dua dadu sama dengan 7, keluar dari perulangan dan tampilkan pesan kemenangan beserta jumlah percobaannya.

Program ini menggunakan konsep dasar perulangan *while* untuk melakukan proses yang tidak diketahui jumlah pengulangannya secara pasti, sampai jumlah kedua dadu sama dengan 7. Penggunaan `Random` memberikan efek acak yang membuat setiap hasil lemparan berbeda. Logika program sederhana namun efektif untuk menunjukkan bagaimana *looping* bekerja dalam situasi yang bergantung pada kondisi tertentu. Selain itu, program ini juga menggambarkan penerapan praktis konsep probabilitas dan logika pemrograman secara menarik dan mudah dipahami.

2.4. Perulangan While 1

Program ini menggunakan struktur *while loop* untuk mengulangi proses berdasarkan input pengguna. Setiap kali perulangan berjalan, program menampilkan jumlah hitungan (*counter*) dan menanyakan apakah pengguna ingin melanjutkan atau berhenti. Jika pengguna mengetik “tidak”, maka kondisi perulangan berubah menjadi `false` dan program berhenti. Lalu menampilkan total jumlah perulangan yang telah dilakukan.

```

1 package Pekan6_2511532005;
2
3 import java.util.Scanner;
4
5 public class perulanganWhile1_2511532005 {
6     public static void main (String[] args) {
7
8         int counter=0;
9         String jawab;
10        boolean running = true;
11
12        Scanner scan = new Scanner (System.in);
13        while (running) {
14            counter++;
15            System.out.println("Jumlah = "+counter);
16            System.out.print("Apakah lanjut (ya / tidak)");
17            jawab = scan.nextLine();
18
19            if (jawab.equalsIgnoreCase("tidak")) {
20                running = false;
21            }
22
23        }
24        System.out.println("Anda sudah melakukan perulangan sebanyak "+counter+" kali");
25    }
26
27 }
28

```

Gambar 2.4. Program Java dengan Perulangan While 1.

Langkah kerja:

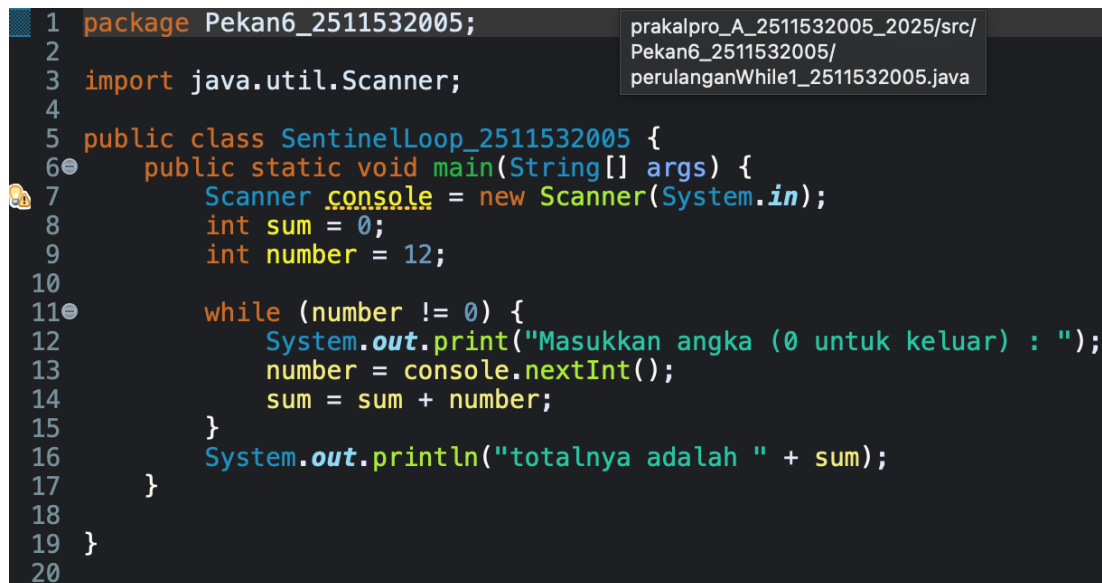
1. Membuat class baru dengan nama perulangan while 1.
2. Impor library java.util.Scanner untuk membaca input dari pengguna.
3. Deklarasi variable counter untuk menghitung jumlah perulangan, jawab untuk menampung jawaban pengguna, dan *running* sebagai penentu kondisi perulangan
4. Inisialisasi *running = true* agar perulangan *while* bisa berjalan.
5. Didalam perulangan, tambahkan nilai counter setiap kali iterasi berjalan.
6. Tampilkan jumlah perulangan ke layar dan minta pengguna menjawab “ya” atau “tidak”.
7. Jika pengguna menjawab “tidak”, ubah nilai *running* menjadi *false* agar perulangan berhenti.
8. Setelah keluar dari perulangan, tampilkan total jumlah perulangan yang telah dilakukan.

Program ini menunjukkan penggunaan *while loop* dengan kondisi logis yang dikendalikan oleh input pengguna. Dengan memanfaatkan variabel *boolean running*, program dapat berjalan secara interaktif sesuai keinginan pengguna-apakah ingin melanjutkan atau menghentikan proses. Pendekatan ini menggambarkan konsep dasar

perulangan bersyarat dengan cara yang sederhana dan mudah dipahami, sekaligus melatih logika control alur program menggunakan tipe data dan percabangan di Java.

2.5. Sentinel Loop

Program ini adalah contoh penerapan perulangan dengan *sentinel value*, yaitu nilai khusus yang digunakan untuk menghentikan *loop*. Program meminta pengguna untuk memasukkan beberapa angka, lalu menjumlahkannya satu per satu selama nilai yang dimasukkan bukan 0. Ketika pengguna mengetik 0, program berhenti dan menampilkan total hasil penjumlahan semua angka yang telah dimasukkan.



```

1 package Pekan6_2511532005;
2
3 import java.util.Scanner;
4
5 public class SentinelLoop_2511532005 {
6     public static void main(String[] args) {
7         Scanner console = new Scanner(System.in);
8         int sum = 0;
9         int number = 12;
10
11         while (number != 0) {
12             System.out.print("Masukkan angka (0 untuk keluar) : ");
13             number = console.nextInt();
14             sum = sum + number;
15         }
16         System.out.println("totalnya adalah " + sum);
17     }
18 }
19 }
20

```

Gambar 2.5. Program Java dengan *Sentinel Loop*.

Langkah kerja:

1. Buat file baru dan impor *library java.util.Scanner*.
2. Di dalam *main()*, buat objek *Scanner* untuk menerima *input* dari pengguna.
3. Deklarasikan variabel *sum* sebagai penampung total nilai dan *number* dengan nilai awal selain 0 agar perulangan bisa dimulai.
4. Gunakan perulangan *while (number !=0)* supaya program terus berjalan selama nilai yang dimasukkan bukan 0.
5. Di dalam *loop*, minta pengguna memasukkan angka, lalu tambahkan nilai tersebut ke variabel *sum*.

6. Jika pengguna memasukkan 0, kondisi while menjadi salah satu dan program keluar dari perulangan.
7. Tampilkan hasil akhir berupa total penjumlahan angka yang dimasukkan keluar dari perulangan.
8. Tampilkan hasil akhir berupa total penjumlahan angka yang dimasukkan pengguna.

Program ini menggambarkan konsep kontrol perulangan menggunakan *sentinel value* dengan sederhana dan efektif. Penggunaan angka 0 sebagai penanda akhir memberikan fleksibilitas bagi pengguna untuk menentukan kapan program berhenti. Dari sisi logika, program ini mengajarkan bagaimana mengelola kondisi penghentian loop serta akumulasi nilai menggunakan variabel penjumlahan (*sum*), yang sangat penting dalam pembuatan program interaktif berbasis input pengguna.

BAB III

KESIMPULAN

Dari hasil praktikum mengenai perulangan *while* dan *do while*, bisa disimpulkan bahwa kedua struktur ini sangat berguna untuk menjalankan perintah secara berulang berdasarkan kondisi tertentu. Perulangan *while* cocok digunakan Ketika kita belum tahu pasti berapa kali proses harus dijalankan, sedangkan *do while* menjamin minimal satu kali eksekusi sebelum memeriksa kondisi. Melalui percobaan seperti lempar dadu, perulangan input pengguna, dan penjumlahan angka sentinel, kita bisa lebih paham bagaimana logika perulangan bekerja dalam program secara nyata dan interaktif.

Agar hasil praktikum semakin maksimal, disarankan untuk sering mencoba berbagai variasi kondisi dan menambahkan fitur tambahan seperti validasi input atau pesan interaktif agar program lebih dinamis. Selain itu, penting juga untuk memahami alur logika sebelum mengetik kode, supaya tidak hanya bisa menjalankan program, tapi juga benar-benar mengerti cara kerjanya. Dengan begitu, kemampuan logika dan pemahan konsep pemrograman akan semakin matang sering latihan yang konsisten.

DAFTAR PUSTAKA

- [1] andre, "Tutorial Belajar Java Part 35: Perulangan DO WHILE Bahasa Java," Duniaikom, 26 September 2020. [Online]. Available: <https://www.duniaikom.com/tutorial-belajar-java-perulangan-do-while-bahasa-java/>.
- [2] J. Squirrels, "Java While Loop," CodeGym, 21 Juli 2023. [Online]. Available: <https://codegym.cc/id/groups/posts/id.691.java-while-loop>.
- [3] B. Hartono, "Pemrograman Java Untuk Pemula," Yayasan Prima Agus Teknik , 27 juli 2020. [Online]. Available: <https://penerbit.stekom.ac.id/index.php/yayasanpat/article/view/351/370>.