

LAPORAN PRAKTIKUM
AGORITMA PEMROGRAMAN
“PERULANGAN WHILE DAN DO WHILE”

DISUSUN OLEH:

ALIYATAR RAFI AHMAD

2511533031

DOSEN PENGAMPU:

Dr. WAHYUDI, S.T, M.T

ASISTEN PRAKTIKUM:

JOVANTRI IMMANUEL GULO



DEPARTEMEN INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Puji syukur kehadiran Tuhan Yang Maha Esa karena atas rahmat dan karunia-Nya, penulis dapat menyelesaikan laporan praktikum dengan judul “**Perulangan While dan Do While dalam Bahasa Java**” tepat pada waktunya. Laporan ini disusun sebagai salah satu tugas pada mata kuliah **Algoritma Pemrograman**, dengan tujuan untuk memahami serta mengaplikasikan konsep perulangan dalam bahasa pemrograman Java.

Melalui praktikum ini, penulis mempelajari cara kerja struktur perulangan **while** dan **do while** yang berfungsi untuk menjalankan suatu blok perintah secara berulang berdasarkan kondisi tertentu. Kedua jenis perulangan ini sangat penting dalam pembuatan program yang dinamis dan efisien, karena dapat mengurangi pengulangan kode serta mempermudah proses pengendalian alur program.

Penulis menyadari bahwa laporan ini masih memiliki kekurangan baik dalam penyusunan maupun isi. Oleh sebab itu, kritik dan saran yang bersifat membangun sangat diharapkan agar laporan dan pemahaman penulis dapat menjadi lebih baik ke depannya.

Akhir kata, semoga laporan ini dapat memberikan manfaat bagi pembaca dan menambah wawasan tentang penerapan perulangan **while** dan **do while** dalam bahasa java.

Padang, 5 November 2025

Aliyatar Rafi Ahmad

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI	ii
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Tujuan.....	1
1.3 Manfaat.....	2
BAB II PEMBAHASAN	3
2.1 Perulangan While	3
2.2 Do..While.....	4
2.3 Kode Program	5
2.3.1 Perulangan While 1	5
2.3.2 Sentinel Loop	6
2.3.3 Game Penjumlahan.....	7
2.3.4 Lempar Dadu.....	8
2.3.5 Do while.....	9
BAB III PENUTUP	11
3.1 Kesimpulan.....	11
3.2 Saran.....	11
DAFTAR PUSTAKA.....	12

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam dunia pemrograman, perulangan atau *looping* merupakan salah satu konsep dasar yang sangat penting untuk dikuasai. Perulangan digunakan untuk menjalankan suatu blok perintah secara berulang tanpa harus menulis kode yang sama berkali-kali. Dengan adanya perulangan, efisiensi dan fleksibilitas dalam penyusunan program dapat tercapai, sehingga programmer dapat menyelesaikan berbagai permasalahan logika dengan lebih efektif.

Dalam bahasa pemrograman Java, terdapat beberapa jenis perulangan yang sering digunakan, di antaranya *for*, *while*, dan *do while*. Pada praktikum ini, fokus pembahasan difokuskan pada *perulangan while* dan *do while*. Struktur *while* digunakan ketika jumlah pengulangan belum diketahui secara pasti, karena perulangan ini akan terus berjalan selama kondisi bernilai benar (*true*). Sedangkan *do while* memiliki perbedaan utama pada pengecekan kondisi yang dilakukan setelah perintah dijalankan, sehingga minimal satu kali perulangan akan selalu terjadi.

Melalui pemahaman terhadap perulangan *while* dan *do while*, mahasiswa diharapkan mampu memahami cara kerja logika pengulangan yang dikontrol oleh kondisi, serta mampu menerapkannya dalam pembuatan program sederhana hingga kompleks. Dengan demikian, penguasaan konsep ini menjadi pondasi penting dalam pembelajaran Algoritma Pemrograman, khususnya dalam membangun alur logika yang sistematis dan efisien.

1.2 Tujuan

1. Memahami konsep dasar perulangan dalam pemrograman, khususnya penggunaan struktur *while* dan *do while* pada bahasa Java.
2. Meningkatkan kemampuan logika algoritma, terutama dalam merancang program yang memerlukan proses pengulangan berdasarkan kondisi tertentu.

3. Mampu membedakan cara kerja antara perulangan *while* dan *do while*, baik dari segi alur logika maupun kondisi pengulangannya.

1.3 Manfaat

1. Menambah pemahaman dasar tentang logika pemrograman, khususnya dalam perulangan *while* dan *do while*
2. Memahami cara kerja perulangan dalam bahasa Java, khususnya struktur *while* dan *do while*.
3. Melatih kemampuan berpikir logis dan sistematis dalam menyusun algoritma yang menggunakan pengulangan.

BAB II

PEMBAHASAN

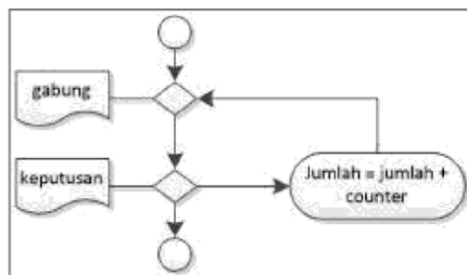
2.1 Perulangan While

Perulangan *while* adalah perulangan yang digunakan untuk mengulang suatu perintah perulangan yang belum diketahui jumlahnya. Perulangan **while** akan terus dijalankan selama kondisi yang dieksekusi bernilai true. Perulangan while juga digunakan untuk perulangan yang memiliki kondisi dan tidak tentu berapa banyak perulangannya. Perulangan while berguna untuk memproses suatu pernyataan atau beberapa pernyataan beberapa kali. Selama ungkapan bernilai benar, pernyataan akan selalu dikerjakan. Oleh karena itu, kita harus membuat kondisi suatu saat ungkapan bernilai salah agar perulangan berakhir.

```
while (kondisi) {  
    Statement;  
}
```

Ket: Jika kondisi bernilai true, maka Statement; akan terus dieksekusi dan proses akan berlanjut diulangi terus menerus. Tetapi, jika kondisi berada sebelum badan pernyataan. Sehingga ketika kondisi sejak awal bernilai false, maka Statement; tidak akan dieksekusi.

Pernyataan ;



Gambar 1.1

Keterangan :

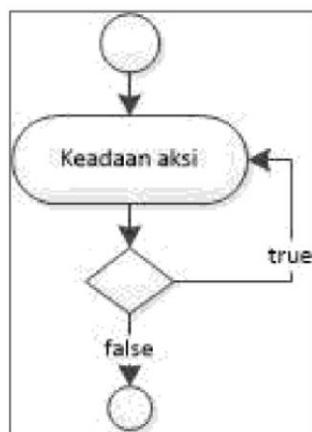
1. Bagian pernyataan akan dieksekusi selama ungkapan dalam while bernilai benar.

2. Pengujian terhadap ungkapan pada `while` dilakukan sebelum bagian pernyataan.
3. Kemungkinan pernyataan pada `while` tidak dijalankan sama sekali, jika ketemu kondisi yang pertama kali bernilai salah.

2.2 Do..While

Pengulangan dengan ***do...while*** ini juga digunakan untuk mengerjakan sebuah atau sekelompok pernyataan berulang-ulang. Bedanya dengan ***while*** adalah pernyataan ***do...while*** akan mengecek kondisi di belakang, sementara ***while*** cek kondisi ada di depan. Bentuk umumnya adalah :

```
do {  
    Statement;  
} while (kondisi);
```



Gambar 1.2

Keterangan :

1. Bagian pernyataan 1 hingga pernyataan N dijalankan secara berulang sampai pernyataan bernilai salah.
2. Pengujian ungkapan dilakukan setelah bagian pernyataan, maka pada pernyataan `do while` minimal akan dijalankan sekali, karena begitu masuk ke blok perulangan, tidak ada cek kondisi tetapi langsung mengerjakan pernyataan.

2.3 Kode Program

Pada praktikum pekan 6, mahasiswa telah membuat program mengenai penulisan *while* dan *do while* dalam bahasa java. Berikut kode program yang telah dibuat dalam praktikum pekan 6.

2.3.1 Perulangan While 1

```

1 package pekan6_2511533031;
2
3 import java.util.Scanner;
4
5 public class perulanganWhile1_2511533031 {
6
7     public static void main(String[] args) {
8
9         int counter=0;
10        String jawab;
11        boolean running = true;
12        //deklarasi scanner
13        Scanner scan = new Scanner(System.in);
14        while (running) {
15            counter++;
16            System.out.println("Jumlah = "+counter);
17            System.out.print("Apakah lanjut (ya / tidak?)");
18            jawab= scan.nextLine();
19            //cek jawab = tidak, perulangan berhenti
20            if(jawab.equalsIgnoreCase("tidak")) {
21                running= false;
22            }
23        }
24        System.out.println("Anda sudah melakukan perulangan sebanyak "+counter+" kali");
25    }
26 }
27 }

```

Gambar 2.1

Program ini menggunakan perulangan *while* dalam bahasa Java untuk menjalankan proses berulang berdasarkan input pengguna. Variabel counter digunakan untuk menghitung jumlah perulangan, sedangkan running berfungsi sebagai penentu apakah perulangan masih berjalan. Di dalam loop, program menampilkan jumlah perulangan dan meminta pengguna menjawab “ya” atau “tidak”. Jika pengguna menjawab “ya”, perulangan akan dilanjutkan; tetapi jika menjawab “tidak”, nilai running diubah menjadi false sehingga perulangan berhenti. Setelah keluar dari loop, program menampilkan total berapa kali perulangan dilakukan.

Langkah Pembuatan :

1. Deklarasikan variabel counter, jawab, dan running.
2. Buat objek Scanner untuk membaca input dari pengguna.
3. Gunakan **perulangan while** dengan kondisi running == true.

4. Di dalam perulangan, tampilkan nilai counter dan minta input “ya” atau “tidak”.
5. Jika pengguna menjawab “tidak”, ubah nilai running menjadi false agar perulangan berhenti.
6. Setelah keluar dari perulangan, tampilkan jumlah total perulangan yang telah dilakukan.

2.3.2 Sentinel Loop

```

1 package pekan6_2511533031;
2
3 import java.util.Scanner;
4
5 public class SentinelLoop_2511533031 {
6
7     public static void main(String[] args) {
8         Scanner console = new Scanner(System.in);
9         int sum = 0;
10        int number = 12;
11
12        while (number != 0) {
13            System.out.print("Masukkan angka (0 untuk keluar): ");
14            number = console.nextInt();
15            sum = sum + number;
16        }
17        System.out.println("Totalnya adalah " + sum);
18    }
19 }
20 }
21

```

Gambar 2.2

Program ini mengimplementasikan *sentinel loop* memakai while di Java untuk menjumlahkan angka yang dimasukkan pengguna sampai pengguna memasukkan angka 0 (sebagai sentinel/penanda berhenti). Program mendeklarasikan `int sum = 0`; sebagai penampung total dan `int number = 12`; sebagai variabel input awal (diinisialisasi bukan karena diperlukan, hanya untuk membuat kondisi awal berbeda dari 0). Di dalam `while (number != 0)` program meminta input angka, membaca dengan `console.nextInt()`, lalu menambahkan nilai tersebut ke `sum`. Jika pengguna memasukkan 0, kondisi `number != 0` menjadi false dan loop berhenti, kemudian program mencetak total penjumlahan.

Langkah Pembuatan :

1. Deklarasikan `int sum = 0`; untuk menyimpan total.
2. Siapkan variabel `int number` (bisa diinisialisasi non-nol) untuk kontrol loop.

3. Buat Scanner console = new Scanner(System.in); untuk baca input.
4. Gunakan while (number != 0) sebagai loop sentinel.
5. Di dalam loop: tampilkan prompt, baca number = console.nextInt();, lalu sum += number;.
6. Setelah loop selesai (user mengetik 0), cetak sum dengan System.out.println("Totalnya adalah " + sum);.

2.3.3 Game Penjumlahan

```

6 public class Gamepenjumlahan_2511533031 {
7     public static void main(String[] args) {
8         Scanner console = new Scanner (System.in);
9         Random rand = new Random();
10        //play until user gets 3 wrong
11        int points = 0;
12        int wrong = 0;
13        while (wrong < 3) {
14            int result = play (console, rand);
15            if (result > 0) {
16                points++;
17            } else {
18                wrong++;
19            }
20        }
21        System.out.println("You earned " + points + " total points.");
22    }
23    //membuat soal penjumlahan dan tampilkan ke user
24    public static int play(Scanner console, Random rand) {
25        //print the operand being added, and sum them
26        int operands = rand.nextInt(4) + 2;
27        int sum = rand.nextInt(10) + 1;
28        System.out.print(sum);
29
30        for (int i = 2; i <= operands; i++) {
31            int n = rand.nextInt (10) + 1;
32            sum += n;
33            System.out.print(" + " + n);
34        }
35        System.out.print(" = ");
36
37        // read user's guess and report wheather it was correct
38        int guess = console.nextInt();
39        if (guess == sum) {
40            System.out.println("Correct!");
41            return 1;
42        } else {
43            System.out.println("Wrong! The answer was " + sum);
44            return 0;
45        }
46    }
47 }

```

Gambar 2.3

Program ini adalah permainan sederhana menggunakan perulangan while di Java. Tujuannya untuk menguji kemampuan pengguna dalam menjawab soal penjumlahan acak. Program menggunakan dua objek penting: Scanner untuk input dari pengguna, dan Random untuk menghasilkan angka acak. Pada awalnya, program menetapkan nilai awal points = 0 dan wrong = 0. Selama pengguna belum salah tiga kali (while

(wrong < 3)), program akan memanggil metode play(console, rand) untuk menampilkan soal baru. Jika jawaban pengguna benar, poin bertambah (points++); kalau salah, jumlah kesalahan naik (wrong++). Setelah pengguna salah tiga kali, program berhenti dan menampilkan total poin yang didapat.

Langkah Pembuatan :

1. Deklarasikan Scanner dan Random untuk input dan angka acak.
2. Buat variabel points dan wrong untuk menghitung skor dan kesalahan.
3. Gunakan perulangan while (wrong < 3) agar permainan berlanjut selama salah kurang dari 3 kali.
4. Di dalam loop, panggil metode play() untuk menampilkan soal dan memeriksa jawaban.
5. Tambah poin jika benar, tambah kesalahan jika salah.
6. Setelah keluar dari loop, tampilkan total poin yang diperoleh
7. Dalam metode play(), buat angka acak, tampilkan soal, baca jawaban pengguna, lalu kembalikan 1 (benar) atau 0 (salah).

2.3.4 Lempar Dadu

```

1 package pekan6_2511533031;
2
3 import java.util.Random;
4
5 public class Lempardadu_2511533031 {
6
7     public static void main(String[] args) {
8         Random rand = new Random();
9         int tries = 0;
10        int sum = 0;
11        while (sum != 7) {
12            // roll the dice once
13            int dadu1 = rand.nextInt(6) + 1;
14            int dadu2 = rand.nextInt(6) + 1;
15            sum = dadu1 + dadu2;
16            System.out.println(dadu1 + " + " + dadu2 + " = " + sum);
17            tries++;
18        }
19        System.out.println("You won after " + tries + " tries!");
20    }
21
22 }

```

Gambar 2.4

Program ini mensimulasikan permainan lempar dadu menggunakan perulangan *while*. Tujuannya adalah melempar dua dadu hingga jumlah keduanya sama dengan 7. Program memakai Random untuk menghasilkan angka acak antara 1 sampai 6 pada dua variabel, yaitu

dadu1 dan dadu2. Setiap kali dadu dilempar, hasilnya dijumlahkan dan dicetak ke layar. Nilai tries digunakan untuk menghitung berapa kali percobaan dilakukan. Perulangan while ($\text{sum} \neq 7$) akan terus berjalan sampai total dua dadu sama dengan 7. Setelah itu, program berhenti dan menampilkan berapa kali percobaan yang dibutuhkan untuk mendapatkan jumlah 7.

Langkah Pembuatan :

1. Import library `java.util.Random` untuk menghasilkan angka acak.
2. Buat objek `Random rand = new Random();` agar bisa digunakan untuk lempar dadu.
3. Inisialisasi variabel `tries` untuk menghitung jumlah percobaan dan `sum` untuk menyimpan hasil penjumlahan dadu.
4. Gunakan perulangan `while (sum != 7)` agar program terus berjalan sampai hasil dadu berjumlah 7.
5. Di dalam loop, buat dua angka acak `dadu1` dan `dadu2` dengan `rand.nextInt(6) + 1`.
6. Jumlahkan kedua dadu, tampilkan hasilnya, lalu tambahkan nilai `tries`
7. Setelah keluar dari loop, cetak pesan kemenangan dengan jumlah percobaan yang dilakukan.

2.3.5 Do while

```

1 package pekan6_2511533031;
2
3 import java.util.Scanner;
4
5 public class doWhile1_2511533031 {
6
7     public static void main(String[] args) {
8         Scanner console = new Scanner(System.in);
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

```

Gambar 2.5

Program ini menggunakan **perulangan do-while** untuk memverifikasi password yang dimasukkan oleh pengguna. Pada setiap perulangan, program meminta input password lewat `System.out.print("Input Password: ")`. Nilai input disimpan ke dalam variabel `phrase`. Perulangan akan terus dijalankan selama input tidak sama dengan string `"abcd"`. Kalau pengguna mengetik `"abcd"`, kondisi `!phrase.equals("abcd")` akan bernilai **false**, sehingga loop berhenti. Program ini cocok buat simulasi sederhana login/password menggunakan konsep **do-while**, karena minimal satu kali input pasti dijalankan.

Langkah Pembuatan :

1. Import library `java.util.Scanner` untuk membaca input dari pengguna.
2. Buat objek `Scanner console = new Scanner(System.in);`.
3. Deklarasikan variabel `String phrase;` untuk menampung input.
4. Gunakan struktur **do-while** agar perintah dijalankan minimal satu kali.
5. Di dalam blok `do`, tampilkan pesan `"Input Password:"` dan baca input dari user.
6. Kondisikan **while** `(!phrase.equals("abcd"))`; agar loop berhenti hanya jika password yang dimasukkan benar.

BAB III

PENUTUP

3.1 Kesimpulan

Dari seluruh percobaan yang dilakukan, dapat disimpulkan bahwa struktur perulangan *while* dan *do while* berfungsi untuk menjalankan suatu blok kode secara berulang selama kondisi tertentu terpenuhi. Perulangan *while* akan memeriksa kondisi terlebih dahulu sebelum menjalankan blok perintah, sedangkan *do while* menjalankan blok perintah minimal satu kali sebelum melakukan pengecekan kondisi. Melalui berbagai contoh program seperti perhitungan jumlah, sentinel loop, permainan penjumlahan, lempar dadu, hingga validasi password, dapat dipahami bahwa penggunaan perulangan ini mempermudah pengulangan proses tanpa harus menulis kode berulang.

Secara keseluruhan, konsep perulangan sangat penting dalam pemrograman karena membantu efisiensi, mengurangi kesalahan penulisan kode, dan meningkatkan fleksibilitas program.

3.2 Saran

1. Praktikum sebaiknya dilakukan dengan penjelasan yang lebih interaktif agar peserta lebih mudah memahami konsep perulangan.
2. Dosen maupun asisten dapat memberikan contoh program yang lebih beragam agar peserta bisa melihat penerapan dalam berbagai situasi.
3. Materi perulangan *while* dan *do while* sebaiknya dilengkapi dengan penjelasan logika kondisi agar kesalahan saat membuat program bisa diminimalkan.
4. Setiap percobaan perlu diberikan pembahasan hasil secara singkat supaya peserta memahami alur kerja program.

DAFTAR PUSTAKA

- [1] Bellshade, “Java – Perulangan while”, [Daring]. Tersedia pada: [Bellshade | Java - Perulangan While](#) [Diakses 06-November-2025]
- [2] Evi Pratiwi, “Konsep Dasar Algoritma dan Pemrograman dengan Bahasa Java”, 2020.