

LAPORAN PRAKTIKUM PEKAN 6



MATA KULIAH ALGORITMA DAN PEMROGRAMAN

DOSEN PENGAMPU :

WAHYUDI, S.T M.T

OLEH :

MUHAMMAD HANS NAFIS

NIM : 2511532027

FAKULTAS TEKNOLOGI INFORMASI

DEPARTEMEN INFORMATIKA

UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Puji syukur kepada Allah SWT atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Laporan Praktikum Pekan 6 ini tepat pada waktunya. Laporan ini disusun untuk memenuhi salah satu tugas praktikum mata kuliah Algoritma dan Pemograman, dengan topik pembahasan mengenai *while loop* dan *do while loop* pada Bahasa Pemrograman Java.

Melalui praktikum ini, penulis mempelajari bagaimana *while loop* dan *do while loop* digunakan untuk menjalankan suatu blok perintah secara berulang hingga kondisi tertentu terpenuhi. Pemahaman terhadap jenis-jenis perulangan ini sangat penting karena menjadi dasar dalam pembuatan logika program yang efisien dan terstruktur.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun demi perbaikan di masa mendatang. Akhir kata, penulis mengucapkan terima kasih kepada Bapak Wahyudi S.T. M.T. selaku dosen pengampu mata kuliah Algoritma dan Pemrograman, asisten laboratorium, serta teman-teman praktikum dan pihak lain yang turut mendukung penulisan laporan ini.

Rabu, 05 November 2025

Muhammad Hans Nafis

DAFTAR ISI

KATA PENGANTAR	i
DAFTAR ISI	ii
BAB 1	1
PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Tujuan Praktikum	1
1.3 Manfaat Praktikum	1
BAB 2	2
PEMBAHASAN	2
2.1 While	2
2.2 Do While	6
BAB 3	7
KESIMPULAN	7
DAFTAR PUSTAKA	8
LAMPIRAN	9

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Dalam dunia pemrograman, proses pengulangan atau *looping* merupakan salah satu konsep dasar yang sangat penting. *Looping* memungkinkan sebuah program untuk mengeksekusi perintah tertentu secara berulang tanpa harus menulis kode yang sama berkali-kali. Dalam bahasa pemrograman Java, terdapat beberapa jenis perulangan seperti *while loop* dan *do while loop*.

While loop digunakan ketika jumlah pengulangan belum diketahui dan bergantung pada kondisi yang dievaluasi di awal. *Do while loop* serupa, namun memastikan bahwa blok perintah dijalankan setidaknya satu kali sebelum pengecekan kondisi.

1.2 Tujuan Praktikum

Tujuan dari pelaksanaan praktikum antara lain sebagai berikut :

1. Memahami konsep dasar perulangan *while* dan *do while* dalam bahasa Java.
2. Mampu mengimplementasikan perulangan tersebut dalam berbagai kasus pemrograman.
3. Mengetahui perbedaan dan penggunaan yang tepat dari masing-masing jenis perulangan.

1.3 Manfaat Praktikum

Manfaat dari pelaksanaan praktikum antara lain sebagai berikut :

1. Meningkatkan pemahaman mahasiswa terhadap struktur kontrol perulangan dalam pemrograman Java.
2. Membantu mahasiswa menulis kode program yang efisien, sistematis, dan mudah dipahami.
3. Menjadi dasar dalam pengembangan logika pemrograman untuk menyelesaikan permasalahan yang membutuhkan proses berulang.

BAB 2

PEMBAHASAN

2.1 While

While Loop dapat ditemukan hampir di semua bahasa pemrograman. *While Loop* berfungsi untuk mengulangi blok pernyataan secara terus menerus, selama kondisinya bernilai *true*. Jika kondisinya bernilai *false*, kontrol dipindahkan ke blok pernyataan atau baris kode yang muncul setelah kode perulangan. Sintaks *while loop* ini hampir sama dengan sintaks *for loop*, beda nya ketiga kondisinya saling terpisah.

Sintaks *while* :

```
inisialisasi;  
while (test) {  
    statement;  
    update;  
}
```

Bagian *inisialisasi* digunakan untuk menginisialisasi variabel penghitung yang akan digunakan dalam perulangan. Inisialisasi hanya dilakukan sekali. Bagian *test* merupakan ekspresi logika yang akan diperiksa sebelum setiap iterasi. Jika bernilai *true*, blok pernyataan di dalam *loop* akan dijalankan. Jika bernilai *false*, perulangan akan berhenti. Bagian *update* digunakan untuk mengubah nilai variabel penghitung setelah setiap kali iterasi selesai, biasanya dengan menambah nilai/*increment* (variabel++) atau mengurangi nilai/*decrement* (variabel--). Perlu diketahui bahwa pada *while loop*, kita bisa meletakkan kondisi *update* di bagian atas *statement*, maupun di bagian bawah, tergantung pada kode program yang akan dibuat dan dijalankan.

Beberapa contoh kode program *while* :

```
public static void main(String[] args) {  
    Random rand = new Random();  
    int tries = 0;  
    int sum = 6;  
    while (sum != 7) {  
        // roll the dice once  
        int dadu1 = rand.nextInt(6) + 1;  
        int dadu2 = rand.nextInt(6) + 1;  
        sum = dadu1 + dadu2;  
        System.out.println(dadu1 + " + " + dadu2 + " = " +  
    sum);
```

```

        tries++;
    }
    System.out.println("You won after " + tries + " tries!");
}
}

```

Program di atas merupakan simulasi permainan dadu menggunakan perulangan *while*. Program akan terus melempar dua dadu secara acak hingga jumlah keduanya menghasilkan angka 7. Setiap kali perulangan dijalankan, dua angka acak antara 1 sampai 6 dihasilkan menggunakan `rand.nextInt(6) + 1`, kemudian dijumlahkan dan ditampilkan ke layar dalam format “dadu1 + dadu2 = sum”. Variabel *tries* digunakan untuk menghitung berapa kali percobaan dilakukan. Perulangan akan berhenti ketika jumlah kedua dadu (*sum*) sama dengan 7, dan program menampilkan pesan kemenangan berupa “You won after X tries!” yang menunjukkan jumlah percobaan yang dibutuhkan untuk mendapatkan angka 7.

Output :

```

1 + 2 = 3
1 + 1 = 2
6 + 4 = 10
4 + 5 = 9
4 + 6 = 10
1 + 3 = 4
3 + 1 = 4
5 + 1 = 6
3 + 5 = 8
6 + 6 = 12
2 + 4 = 6
2 + 5 = 7
You won after 12 tries!

```

Gambar 2.1

```

public static void main(String[] args) {
    int counter=0;
    String jawab;
    boolean running = true;
    // deklarasi scanner
    Scanner scan = new Scanner (System.in);
    while (running) {
        counter++;
        System.out.println("Jumlah = "+counter);
        System.out.println("Apakah lanjut (ya / tidak?)");
        jawab= scan.nextLine();
        // cek jawab = tidak, perulangan berhenti
        if (jawab.equalsIgnoreCase("tidak")) {
            running= false;
        }
    }
}

```

```

        System.out.println("Anda sudah melakukan perulangan
sebanyak "+counter+" kali");
        scan.close();
    }
}

```

Program akan terus mengulangi proses selama variabel *running* bernilai *true*. Setiap kali perulangan dilakukan, nilai *counter* bertambah satu dan program menampilkan jumlah perulangan saat ini. Kemudian, pengguna diminta untuk menjawab apakah ingin melanjutkan perulangan atau tidak. Jika pengguna mengetik “tidak”, maka kondisi *running* diubah menjadi *false*, sehingga perulangan berhenti. Setelah keluar dari *loop*, program menampilkan total berapa kali perulangan telah dilakukan sebelum program selesai.

Output :

```

Jumlah = 1
Apakah lanjut (ya / tidak?)
ya
Jumlah = 2
Apakah lanjut (ya / tidak?)
ya
Jumlah = 3
Apakah lanjut (ya / tidak?)
tidak
Anda sudah melakukan perulangan sebanyak 3 kali

```

Gambar 2.2

```

public static void main(String[] args) {
    Scanner console = new Scanner(System.in);
    Random rand = new Random();
    // play until gets 3 wrong
    int points = 0;
    int wrong = 0;
    while (wrong < 3) {
        int result = play(console, rand); // play one game
        if (result > 0) {
            points++;
        } else {
            wrong++;
        }
    }
    System.out.println("You earned " + points + " total
points.");
}
// membuat soal penjumlahan dan ditampilkan ke user
public static int play(Scanner console, Random rand) {
    // print the operands being added, and sum them
    int operands = rand.nextInt(4) + 2;
    int sum = rand.nextInt(10) + 1;
}

```

```

        System.out.print(sum);
        for (int i = 2; i <= operands; i++) {
            int n = rand.nextInt(10) + 1;
            sum += n;
            System.out.print(" + " + n);
        }
        System.out.print(" = ");

        // read user's guess and report whether it was correct
        int guess = console.nextInt();
        if (guess == sum) {
            return 1;
        }else {
            System.out.println("Wrong! The answer was " +
sum);
            return 0;
        }
    }
}

```

Program ini menggunakan *loop while* untuk terus berjalan sampai pengguna menjawab salah sebanyak tiga kali. Dalam setiap putaran, metode *play()* akan membuat soal penjumlahan acak dengan 2 hingga 5 bilangan antara 1 sampai 10, lalu menampilkan soal tersebut ke layar dan meminta pengguna memasukkan jawabannya. Jika jawaban benar, poin pengguna bertambah satu; jika salah, jumlah kesalahan meningkat satu dan program menampilkan jawaban yang benar. Setelah pengguna melakukan tiga kesalahan, permainan berakhir dan program menampilkan total poin yang berhasil diperoleh.

Output :

```

2 + 1 = 3
5 + 8 + 10 + 3 = 26
7 + 1 = 8
7 + 2 + 5 = 1
Wrong! The answer was 14
2 + 6 + 7 + 4 + 10 = 1
Wrong! The answer was 29
2 + 8 = 1
Wrong! The answer was 10
You earned 3 total points.

```

Gambar 2.3

2.2 Do While

Do while loop merupakan modifikasi dari *while loop*, yakni dengan memindahkan posisi *test* ke bagian akhir perulangan. Artinya, suatu perulangan dilakukan terlebih dahulu, baru diperiksa apakah kondisinya terpenuhi (bernilai *true*) atau belum (bernilai *false*).

Sintaks *Do While* :

```
inisialisasi;  
do{  
    statement;  
    update;  
}  
while (test)
```

Contoh kode program *Do While* :

```
public static void main(String[] args) {  
    Scanner console = new Scanner(System.in);  
    String phrase;  
    do {  
        System.out.print("Input Password: ");  
        phrase = console.next();  
    }while (!phrase.equals("abcd"));  
    console.close();  
}
```

Program di atas merupakan contoh penggunaan struktur perulangan *do-while* pada bahasa Java untuk memvalidasi input pengguna. Program akan terus meminta pengguna memasukkan kata sandi dengan menampilkan pesan “Input Password:” sampai pengguna mengetikkan kata sandi yang benar, yaitu “abcd”. Perulangan *do-while* digunakan karena menjamin bahwa blok perintah di dalamnya akan dijalankan setidaknya satu kali, sebelum kondisi diperiksa. Setelah input yang dimasukkan sesuai dengan string “abcd”, kondisi pada while menjadi *false*, sehingga perulangan berhenti dan program menutup objek *Scanner*.

Output :

```
Input Password: 1234  
Input Password: qwerty  
Input Password: abcd  
|
```

Gambar 2.4

BAB 3

KESIMPULAN

Berdasarkan hasil praktikum yang telah dilakukan, dapat disimpulkan bahwa perulangan (*looping*) merupakan struktur penting dalam pemrograman yang berfungsi untuk mengeksekusi suatu blok perintah secara berulang hingga kondisi tertentu terpenuhi. Pada bahasa Java, *while loop* digunakan ketika jumlah pengulangan belum diketahui dan bergantung pada kondisi yang diuji di awal, sedangkan *do while loop* memastikan bahwa perintah dijalankan minimal satu kali sebelum kondisi dievaluasi. Melalui praktikum ini, mahasiswa dapat memahami perbedaan cara kerja, kelebihan, serta penerapan masing-masing jenis perulangan dalam berbagai situasi pemrograman. Pemahaman ini menjadi dasar penting untuk mengembangkan logika algoritmik dan menulis program yang lebih efisien, terstruktur, serta mudah dikembangkan di masa mendatang.

Sebagai saran, disarankan mahasiswa lebih aktif berlatih menulis dan menjalankan program secara mandiri di luar jam praktikum. Hal ini penting karena keterampilan pemrograman hanya dapat dikuasai dengan sering berlatih. Dengan demikian, diharapkan pada praktikum berikutnya mahasiswa dapat lebih cepat memahami materi, meminimalisir kesalahan sintaks, serta mampu mengembangkan program yang lebih kompleks dan bermanfaat.

DAFTAR PUSTAKA

- [1] <https://www.duniaIlkom.com/tutorial-belajar-java-perulangan-while-bahasa-java/> [Diakses 05 November 2025]
- [2] https://www.w3schools.com/java/java_while_loop.asp [Diakses 05 November 2025]
- [3] <https://codegym.cc/id/groups/posts/id.691.java-while-loop> [Diakses 05 November 2025]
- [4] <https://www.duniaIlkom.com/tutorial-belajar-java-perulangan-do-while-bahasa-java/> [Diakses 05 November 2025]

LAMPIRAN

Gambar 2.1	3
Gambar 2.2	4
Gambar 2.3	5
Gambar 2.4	6