

LAPORAN PRAKTIKUM
PRAKTIKUM ALGORITMA DAN PEMROGRAMAN

“PERULANGAN WHILE DAN DO=WHILE”

disusun oleh:

Dennis Shauqi Akbar

2511533020

Dosen pengampu: Dr. Wahyudi. S.T.M.T

Asisten Pratikum: Rahmad DRO



DEPARTEMEN INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ANDALAS
TAHUN 2025

KATA PENGANTAR

Puji dan syukur kami panjatkan ke hadirat Allah SWT atas segala rahmat dan karunia-Nya, sehingga kami dapat menyelesaikan tugas kelompok ini dengan baik. Shalawat serta salam semoga senantiasa tercurah kepada junjungan kita Nabi Muhammad SAW, keluarga, sahabat, serta para pengikutnya hingga akhir zaman.

Tugas Individu ini kami susun dengan tujuan untuk memenuhi salah satu tugas mata kuliah Praktikum Algoritma dan Pemrograman, dengan Judul “ Perulangan while dan do=while” yang mencakup Lempar dadu, Game Penjumlahan, Sentinel Loop, doWhile

Kami menyadari bahwa penyusunan tugas ini masih jauh dari sempurna, baik dari segi materi maupun penyajiannya. Oleh karena itu, kami sangat mengharapkan kritik dan saran yang membangun dari semua pihak demi kesempurnaan tugas kami di masa mendatang.

Akhir kata, kami mengucapkan terima kasih kepada dosen pembimbing yang telah memberikan arahan sehingga tugas ini dapat terselesaikan tepat waktu. Semoga tugas ini dapat bermanfaat bagi kami khususnya, dan bagi pembaca pada umumnya.

Padang, 9 November 2025

Dennis Shauqi Akbar

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI	ii
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Tujuan.....	1
1.3 Manfaat Praktikum.....	2
BAB II PEMBAHASAN	3
2.1 Program PerulanganFor1	3
2.2 Program PerulanganFor2	4
2.3 Program PerulanganFor3	6
2.4 Program PerulanganFor4	8
2.5 Program NestedFor0	Error! Bookmark not defined.
2.6 Program NestedFor1	Error! Bookmark not defined.
2.6 Program NestedFor2	Error! Bookmark not defined.
BAB III KESIMPULAN.....	14
3.1 Kesimpulan	14
3.2 Saran Pengembangan	14
DAFTAR PUSTAKA	15

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam dunia pemrograman komputer, konsep perulangan memiliki peranan yang penting karena digunakan untuk menjalankan perintah secara berulang tanpa perlu menulis kode secara berulang-ulang. Java sebagai salah satu bahasa pemrograman populer menyediakan berbagai bentuk perulangan, di antaranya `while` dan `do-while`. Kedua jenis perulangan ini memungkinkan sebuah program menjalankan blok perintah selama kondisi yang ditentukan masih terpenuhi.

Perbedaan utama antara keduanya terletak pada urutan pengecekan kondisi. Struktur `while` akan melakukan pemeriksaan kondisi terlebih dahulu sebelum mengeksekusi blok kode, sedangkan `do-while` mengeksekusi blok kode minimal satu kali terlebih dahulu, baru kemudian memeriksa kondisi. Pemahaman mengenai kedua perulangan ini sangat penting agar programmer dapat memilih bentuk perulangan yang tepat untuk setiap situasi pemrograman.

Melalui kegiatan praktikum ini, mahasiswa diharapkan dapat memahami logika kerja perulangan `while` dan `do-while`, serta mampu menerapkannya dalam pembuatan program Java sederhana yang memerlukan proses pengulangan secara efektif.

1.2 Tujuan

1. Menjelaskan konsep dan cara kerja perulangan `while` dan `do-while` dalam bahasa Java.
2. Membedakan karakteristik serta penggunaan dari masing-masing struktur perulangan.
3. Mengimplementasikan perulangan `while` dan `do-while` dalam program Java sederhana.
4. Melatih kemampuan logika dan analisis dalam menyusun algoritma yang melibatkan perulangan.

1.3 Manfaat Praktikum

1. Meningkatkan pemahaman mahasiswa mengenai penerapan struktur perulangan pada bahasa Java.
2. Membantu mahasiswa menentukan jenis perulangan yang sesuai dengan kebutuhan program.
3. Membiasakan mahasiswa berpikir efisien dalam menyelesaikan masalah menggunakan logika pengulangan.
4. Menjadi bekal awal untuk memahami konsep pemrograman lanjutan seperti rekursi dan pengulangan bersarang.

BAB II

PEMBAHASAN

2.1 Program Lempar Dadu

```
1 package pekan6_2511533020;  
2  
3 import java.util.Random;  
4  
5 public class Lempardadu_2511533020 {  
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            int dadu1 = rand.nextInt(6) + 1;  
13            int dadu2 = rand.nextInt(6) + 1;  
14            sum = dadu1 + dadu2;  
15            System.out.println(dadu1 + " + " + dadu2 + " = " + sum);  
16            tries++;  
17        }  
18        System.out.println("You won after " + tries + "tries!");  
19    }  
20 }  
21  
22  
23 }  
24
```

Kode Program 2.1

Program Lempar Dadu ini dibuat untuk mensimulasikan proses pelemparan dua buah dadu sampai total angkanya berjumlah tujuh. Pada bagian awal, program mengimpor library `java.util.Random` yang digunakan untuk menghasilkan angka secara acak. Selanjutnya, beberapa variabel disiapkan, seperti `tries` yang berfungsi menghitung berapa kali percobaan dilakukan, serta `sum` yang digunakan untuk menyimpan hasil penjumlahan dari kedua dadu. Struktur perulangan yang dipakai adalah `while`, di mana proses akan terus berulang selama nilai `sum` belum mencapai tujuh.

Di dalam perulangan, dua variabel yaitu `dadu1` dan `dadu2` diisi dengan angka acak antara satu hingga enam menggunakan perintah `rand.nextInt(6) + 1`. Nilai dari kedua dadu tersebut kemudian dijumlahkan dan disimpan dalam variabel `sum`. Setiap kali program menjalankan satu putaran perulangan, hasil dari kedua dadu akan ditampilkan ke layar dan variabel `tries` akan bertambah satu sebagai penanda jumlah percobaan. Jika hasil penjumlahan kedua dadu akhirnya bernilai tujuh,

kondisi while menjadi salah dan perulangan otomatis berhenti. Setelah keluar dari perulangan, program menampilkan pesan kemenangan yang menyatakan berapa kali percobaan dilakukan hingga mendapatkan total tujuh, contohnya “You won after 4 tries!”.

Secara keseluruhan, program ini menunjukkan cara kerja perulangan while di Java, di mana kondisi akan diperiksa terlebih dahulu sebelum perintah di dalamnya dijalankan. Melalui program ini, pengguna dapat melihat bagaimana konsep logika pengulangan diterapkan untuk menjalankan proses secara berulang hingga mencapai kondisi tertentu. Selain itu, program juga membantu mahasiswa memahami penerapan dasar logika, operasi aritmatika, serta penggunaan variabel dan kontrol alur dalam pembuatan program.

Output dari program diatas jika dijalankan adalah

```
<terminated> Lempardadu_251
5 + 6 = 11
3 + 2 = 5
5 + 2 = 7
You won after 3tries!
```

Gambar 2.1

2.2 Program Game Penjumlahan

```
1 package pekan0_2511533020;
2 import java.util.*;
3
4 public class GamePenjumlahan_2511533020 {
5
6     public static void main(String[] args) {
7         Scanner console = new Scanner(System.in);
8         Random rand = new Random();
9         // play until user gets wrong
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            System.out.println("You earned" + points + " total points.");
20        }
21    }
22
23    // membuat soal penjumlahan dan ditamalkan ke user
24    public static int play(Scanner console, Random rand) {
25        // print the operands being added, and sum them
26        int operands = rand.nextInt(4) + 2;
27        int sum = rand.nextInt(10) + 1;
28        System.out.println(sum);
29        for (int i = 2; i <= operands; i++) {
30            int n = rand.nextInt(10) + 1;
31            sum += n;
32            System.out.print(" + " + n);
33        }
34        System.out.print(" = ?");
35    }
36
37    // read user's guess and report whether it was correct
38    int guess = console.nextInt();
39    if (guess == sum) {
40        return 1;
41    } else {
42        System.out.println("Tebakan Anda Salah ");
43        return 0;
44    }
45 }
46 }
```

Kode Program 2.2

Program Game Penjumlahan ini dibuat untuk memberikan latihan sederhana kepada pengguna dalam menjawab soal-soal penjumlahan acak. Program

memanfaatkan kelas `Scanner` untuk menerima input dari pengguna dan kelas `Random` untuk menghasilkan angka secara acak. Di dalam metode `main`, terdapat dua variabel penting, yaitu `points` untuk mencatat skor pengguna setiap kali menjawab dengan benar, dan `wrong` untuk menghitung jumlah kesalahan. Program dijalankan di dalam perulangan `while` dengan syarat selama nilai `wrong` masih kurang dari tiga, artinya pengguna diperbolehkan melakukan kesalahan hingga tiga kali sebelum permainan berakhir.

Di setiap iterasi perulangan, program memanggil metode `play()` yang bertugas membuat soal penjumlahan acak. Di dalam metode tersebut, jumlah bilangan yang akan dijumlahkan ditentukan secara acak, lalu setiap bilangan dihasilkan menggunakan `rand.nextInt(10) + 1`, sehingga nilainya berkisar antara satu sampai sepuluh. Angka-angka tersebut kemudian ditampilkan ke layar sebagai soal yang harus dijawab pengguna. Setelah itu, pengguna diminta memasukkan jawabannya melalui `console.nextInt()`. Program akan membandingkan hasil input dengan nilai sebenarnya dari penjumlahan tersebut.

Jika jawaban pengguna benar, metode `play()` akan mengembalikan nilai 1 dan menampilkan pesan keberhasilan, sehingga skor (`points`) bertambah satu. Namun, apabila jawaban salah, metode akan mengembalikan nilai 0 dan menampilkan pesan “Tebakan Anda Salah”. Saat pengguna melakukan tiga kali kesalahan, perulangan berhenti dan program menampilkan total poin yang berhasil diperoleh dengan pesan “You earned X total points.”

Secara keseluruhan, program ini memperlihatkan penerapan perulangan `while`, penggunaan metode dengan nilai balik (`return`), serta interaksi antara input dan output dalam bahasa pemrograman Java. Melalui latihan ini, mahasiswa dapat memahami cara memanfaatkan logika kondisi dan struktur pengulangan untuk membuat program interaktif yang memberikan umpan balik langsung berdasarkan jawaban pengguna.

Output dari program diatas jika kita input jawaban yang benar dan jawaban yang salah maka outputnya adalah sebagai berikut:

```

2 + 7 + 7 + 3 = 19
8 + 9 + 4 + 3 = 24
10 + 8 + 9 + 9 + 8 = 36
Tebakan Anda Salah
1 + 10 + 2 + 3 + 9 = 25
3 + 8 + 8 + 5 = 24

```

Gambar 2.2

2.3 Program Sentinel Loop

```

1 package pekan6_2511533020;
2 import java.util.*;
3 public class SentinelLoop_2511533020 {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         Scanner console = new Scanner (System.in);
8         int sum = 0;
9         int number = 12;          // "dummy value", anything but 0
10
11         while (number != 0) {
12             System.out.println("Masukkan angka (0) untuk keluar:");
13             number = console.nextInt();
14             sum = sum + number;
15
16         }
17         System.out.println( "totalnya adalah " + sum );
18     }
19 }
20 }
21

```

Kode Program 2.3

Program Sentinel Loop ini dirancang untuk menjumlahkan beberapa angka yang dimasukkan oleh pengguna hingga pengguna memutuskan untuk berhenti dengan memasukkan nilai tertentu, yaitu angka nol (0). Program ini memanfaatkan kelas Scanner untuk membaca input dari pengguna melalui keyboard. Pada bagian awal, dua variabel utama dideklarasikan, yaitu sum dengan nilai awal nol untuk menyimpan hasil penjumlahan, dan number yang diisi dengan nilai sembarang, misalnya 12, agar perulangan dapat berjalan pada awal program.

Struktur utama program menggunakan perulangan while dengan kondisi number != 0, yang berarti perulangan akan terus dijalankan selama nilai yang dimasukkan pengguna bukan nol. Di dalam perulangan, program akan menampilkan pesan “Masukkan angka (0) untuk keluar:” sebagai petunjuk kepada pengguna. Kemudian, pengguna diminta memasukkan sebuah bilangan, yang dibaca dan disimpan dalam variabel number. Setiap kali pengguna memasukkan angka selain nol, nilai tersebut akan dijumlahkan dengan sum. Proses ini terus berulang hingga pengguna mengetikkan angka nol.

Begitu angka nol dimasukkan, kondisi perulangan menjadi salah (false), sehingga program keluar dari loop dan menampilkan hasil akhir berupa total dari seluruh angka yang telah dijumlahkan, misalnya “totalnya adalah 45”. Dengan demikian, program ini menunjukkan penerapan konsep sentinel-controlled loop, yaitu perulangan yang berhenti berdasarkan nilai “penanda” tertentu yang dimasukkan oleh pengguna.

Melalui program ini, mahasiswa dapat memahami bagaimana perulangan dikendalikan oleh nilai masukan, serta bagaimana konsep variabel akumulator digunakan untuk menghitung total dari sejumlah data yang dimasukkan secara berulang. Program ini juga memperkuat pemahaman mengenai interaksi antara pengguna dan program melalui input serta penggunaan kondisi dalam perulangan.

Jika Programnya dijalankan dengan input 1 - 5 maka outputnya adalah sebagai berikut:

```
Masukkan angka (0) untuk keluar:
1
Masukkan angka (0) untuk keluar:
2
Masukkan angka (0) untuk keluar:
3
Masukkan angka (0) untuk keluar:
4
Masukkan angka (0) untuk keluar:
5
Masukkan angka (0) untuk keluar:
0
totalnya adalah 15
```

Gambar 2.3

2.4 Program doWhile

```

1 package pekan6_2511533020;
2
3 import java.util.*;
4
5 public class doWhile_2511533020 {
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("abcde"));
14        }
15
16    }

```

Kode Program 2.4

Program di atas merupakan implementasi sederhana dari struktur perulangan do-while dalam bahasa Java yang digunakan untuk melakukan validasi input berupa kata sandi. Pada bagian awal, program mengimpor pustaka `java.util.*` agar dapat menggunakan kelas `Scanner`, yang berfungsi untuk membaca data yang dimasukkan oleh pengguna melalui keyboard. Selanjutnya, dibuat sebuah objek `Scanner` bernama `console` yang akan digunakan untuk mengambil input tersebut.

Di dalam blok `do`, program menampilkan pesan “Input Password” dan menunggu pengguna mengetikkan sebuah kata atau frasa yang kemudian disimpan dalam variabel `phrase`. Setelah itu, kondisi pada bagian `while` akan memeriksa apakah nilai dari `phrase` sudah sama dengan string `"abcde"`.

Jika belum sesuai, program akan mengulangi proses input sampai pengguna memasukkan kata sandi yang benar. Ketika input yang dimasukkan sudah cocok dengan kata kunci tersebut, perulangan berhenti dan program pun selesai dijalankan.

Jika Programnya dijalankan dengan input sampai “abcde” maka outputnya adalah sebagai berikut

```
Input Password 12345  
Input Password 54321  
Input Password 11111  
Input Password 88888  
Input Password abcde
```

Gambar 2.4

BAB III

KESIMPULAN

3.1 Kesimpulan

Dari pembelajaran mengenai struktur perulangan `while` dan `do-while`, dapat disimpulkan bahwa keduanya memiliki fungsi utama untuk menjalankan suatu blok perintah secara berulang selama kondisi yang ditentukan masih bernilai benar. Perbedaan mendasarnya terletak pada proses pengecekan kondisi. Pada `while`, kondisi diperiksa terlebih dahulu sebelum perintah dijalankan, sedangkan pada `do-while`, perintah akan dijalankan minimal satu kali terlebih dahulu sebelum kondisi dievaluasi. Dengan memahami konsep ini, programmer dapat menentukan jenis perulangan yang paling sesuai dengan kebutuhan logika program yang sedang dibuat.

3.2 Saran Pengembangan

Untuk pengembangan selanjutnya, penggunaan perulangan `while` dan `do-while` dapat dikombinasikan dengan struktur kontrol lain seperti `if-else` atau `switch-case` agar program menjadi lebih interaktif dan dinamis. Selain itu, mahasiswa juga disarankan untuk mencoba mengimplementasikan perulangan ini pada kasus nyata, seperti pembuatan sistem login, pengulangan input data, atau simulasi sederhana. Dengan begitu, pemahaman terhadap logika perulangan tidak hanya bersifat teoritis, tetapi juga terasah melalui penerapan langsung dalam pemrograman praktis. Pemrograman yang lebih nyata dan bermanfaat di dunia pengembangan perangkat lunak.

DAFTAR PUSTAKA

- [5] P. Shouthiri and N. Thushika, “A Comparative Analysis of Looping Structures: Comparison of ‘While’ Loop and ‘Do-While’ Loop in the C++ Language,” *Asian Journal of Research in Computer Science*, vol. 2, no. 3, pp. 1–5, Jan. 2019, doi: 10.9734/ajrcos/2018/v2i328752.
- [3] J. Makhijani, M. Niranjani, and Y. Sharma, “A Comparison of while, do-while and for loop in C programming language based on Assembly Code Generation,” *Int. J. Comput. Sci. Eng.*, vol. 7, no. 6, pp. –, Jun. 2019, doi: 10.26438/ijcse/v7i6.12041211.
- [4] M. T. Morazán, “How to Design While Loops,” in *Proc. 9th Computer Science Education Research Conference*, 2020, pp. –, doi: 10.1145/3442481.3442504.