

LAPORAN PRATIKUM ALGORITMA PEMROGRAMAN PEKAN 6



Disusun oleh:

FAHRI SYAH PUTRA RAMADHAN

2511531015

Dosen pengampu

Wahyudi. Dr., S.T,M.T

Asisten Praktikum:

Aufan Taufiqurrahman

DEPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

PADANG

2025

KATA PENGANTAR

Laporan ini disusun sebagai salah satu bentuk pertanggungjawaban kegiatan praktikum algoritma pemrograman yang membahas mengenai Tipe Data dan Variabel pada Bahasa Pemrograman Java. Melalui laporan ini, penulis dapat memahami materi praktikum secara mendalam. Penulisan laporan ini juga dapat melatih ketelitian, keteraturan, serta kemampuan menulis sesuai kaidah akademik. Dengan demikian, laporan praktikum ini dapat berfungsi sebagai sarana belajar, dokumentasi kegiatan, dan referensi untuk praktikum atau pembelajaran jenjang berikutnya.

Penulis menyadari bahwa laporan ini masih memiliki kekurangan, baik dari isi maupun penyajiannya. Oleh sebab itu, saran dan kritik sangat penulis harapkan untuk laporan berikutnya.

Padang, 2025

FAHRI SYAH PUTRA RAMADHAN

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 Program Perulangan while 1	3
2.2 Program Lempar Dadu	5
2.3 Program Game Penjumlahan	6
2.4 Program Sentinel Loop.....	9
2.5 Program do while 1	10
BAB III KESIMPULAN	12
3.1 Kesimpulan.....	12
3.2 Saran	12
DAFTAR PUSTAKA	13

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Dalam dunia pemrograman, perulangan atau looping merupakan konsep dasar yang sangat penting untuk menjalankan suatu blok perintah secara berulang hingga kondisi tertentu terpenuhi. Bahasa pemrograman Java menyediakan beberapa jenis perulangan, di antaranya adalah while dan do while. Kedua struktur ini digunakan ketika jumlah pengulangan tidak diketahui secara pasti di awal program, melainkan bergantung pada suatu kondisi logis. Hal ini menjadikan perulangan while dan do while sangat berguna untuk berbagai proses yang memerlukan pemeriksaan kondisi secara terus-menerus, seperti validasi input, perhitungan dinamis, atau pengulangan sampai syarat tertentu tercapai. Perulangan while bekerja dengan cara memeriksa kondisi terlebih dahulu sebelum mengeksekusi blok perintah di dalamnya. Jika kondisi bernilai benar (true), maka perintah di dalam blok akan dijalankan berulang kali hingga kondisi menjadi salah (false). Sementara itu, perulangan do while memiliki keunikan karena mengeksekusi perintah minimal satu kali terlebih dahulu, baru kemudian melakukan pengecekan kondisi. Dengan memahami kedua struktur perulangan ini, seorang programmer dapat menulis kode yang lebih efisien, fleksibel, dan mudah disesuaikan dengan kebutuhan logika program yang kompleks.

1.2 Tujuan

1. Menjelaskan konsep dasar dan cara kerja perulangan while dan do while.
2. Menunjukkan perbedaan antara struktur while dan do while melalui contoh program.
3. Melatih kemampuan logika pemrograman dengan struktur perulangan while dan do while.

1.3 Manfaat

1. Menambah pemahaman tentang struktur perulangan while dan do while dalam bahasa java.
2. Meningkatkan kemampuan dalam menerapkan logika pemrograman untuk menyelesaikan masalah .
3. Menjadi dasar pembelajaran untuk memahami konsep control alur program (control flow) yang lebih kompleks.

BAB 2

PEMBAHASAN

2.1 Program Perulangan while 1

```
1 package pekan6_2511531015;
2 import java.util.Scanner;
3 public class perulanganWhile1_2511531015 {
4
5     public static void main(String[] args) {
6         int counter=0;
7         String jawab;
8         boolean running = true;
9         // deklarasi scanner
10        Scanner scan = new Scanner(System.in);
11        while (running) {
12            counter++;
13            System.out.println("Jumlah = "+counter);
14            System.out.print("Apakah lanjut (ya / tidak?)");
15            jawab=scan.nextLine();
16            //cek jawab = tidak, perulangan berhenti
17            if (jawab.equalsIgnoreCase("tidak")) {
18                running= false;
19            }
20        }
21        System.out.println("Anda sudah melakukan perulangan sebanyak "+counter+" kali");
22
23
24    }
25
26 }
27 }]
```

Kode 2.1

Nama file: perulanganWhile1_2511531015

1. int counter = 0; : inisialisasi penghitung = 0.
2. boolean running = true; : siapkan flag agar while bisa berjalan.
3. Scanner scan = new Scanner(System.in); : siap membaca input dari keyboard.
4. while (running) { : mulai loop selama running bernilai true.
 - counter++; : naikkan counter terlebih dahulu. Jadi pada iterasi pertama, counter berubah dari 0 → 1.
 - System.out.println("Jumlah = " + counter); : cetak nilai counter saat ini (di baris terpisah).
 - System.out.print("Apakah lanjut (ya/tidak)"); : tampilkan pertanyaan tanpa pindah baris (karena print, bukan println).
 - jawab = scan.nextLine(); : tunggu dan baca jawaban pengguna (seperti ya atau tidak).
 - if(jawab.equalsIgnoreCase("tidak")) { running = false; } : jika jawaban "tidak" (case-insensitive), ubah running jadi false sehingga loop tidak akan diulang lagi.
5. Setelah loop selesai, System.out.println("Anda sudah melakukan perulangan sebanyak " + counter + " kali"); : cetak jumlah total iterasi yang sudah dilakukan. Karena dimulai i = 1 dan diincrement 1 per iterasi, loop akan berjalan untuk i = 1,2,...,10 (total 10 iterasi). Setelah i menjadi 11, kondisi i <= 10 salah → loop berhenti.

Output:

```
Jumlah = 1
Apakah lanjut (ya / tidak?)ya
Jumlah = 2
Apakah lanjut (ya / tidak?)ya
Jumlah = 3
Apakah lanjut (ya / tidak?)ya
Jumlah = 4
Apakah lanjut (ya / tidak?)ya
Jumlah = 5
Apakah lanjut (ya / tidak?)ya
Jumlah = 6
Apakah lanjut (ya / tidak?)ya
Jumlah = 7
Apakah lanjut (ya / tidak?)tidak
Anda sudah melakukan perulangan sebanyak 7 kali
```

Output 2.1

Ketika program dijalankan, variabel counter mulai dari 0 dan running bernilai true, sehingga perulangan while langsung berjalan. Setiap kali loop dijalankan, nilai counter bertambah satu dan ditampilkan ke layar, lalu program menanyakan “Apakah lanjut (ya/tidak)” untuk menentukan apakah perulangan akan diteruskan. Jika pengguna mengetik “ya”, nilai running tetap true sehingga loop berlanjut, tetapi jika mengetik “tidak”, maka running diubah menjadi false dan perulangan berhenti. Setelah keluar dari loop, program menampilkan pesan akhir “Anda sudah melakukan perulangan sebanyak 4 kali”, yang menunjukkan jumlah total iterasi yang telah dijalankan sebelum pengguna memilih berhenti.

2.2 Program Lempar Dadu

```
1 package pekan6_2511531015;
2
3 import java.util.Random;
4
5 public class Lempardadu_2511531015 {
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 }
23 }
24 }
```

Kode 2.2

Nama file: Lempardadu_2511531015

1. Random rand = new Random(); membuat objek rand untuk menghasilkan angka acak dari 1 sampai 6.
2. int tries = 0; int sum = 0; menginisialisasi penghitung percobaan (tries) dan jumlah dadu (sum).
3. while (sum != 7) : memulai perulangan yang akan terus berjalan selama hasil penjumlahan dua dadu belum sama dengan 7.
4. Di dalam loop:
 - int dadu1 = rand.nextInt(6) + 1; menghasilkan angka acak antara 1–6 untuk dadu pertama.
 - int dadu2 = rand.nextInt(6) + 1; menghasilkan angka acak antara 1–6 untuk dadu kedua.
 - sum = dadu1 + dadu2; menjumlahkan kedua dadu.
 - System.out.println(dadu1 + "+" + dadu2 + "=" + sum); menampilkan hasil lemparan.
 - tries++; menambah jumlah percobaan setiap kali loop berjalan.
5. Setelah sum bernilai 7, kondisi while menjadi salah, loop berhenti.
6. System.out.println("You won after " + tries + " tries!"); menampilkan berapa kali percobaan dilakukan sampai jumlah dadu sama dengan 7. Output:

Output:

```
6 + 4 = 10
4 + 2 = 6
2 + 6 = 8
2 + 5 = 7
You won after 4 tries!
```

Output 2.2

Program melempar dua dadu secara acak berulang kali dan menampilkan hasil setiap lemparan hingga total kedua dadu sama dengan 7. Pada contoh di atas, percobaan pertama menghasilkan $1+1=2$, lalu $3+1=4$, kemudian $6+6=12$, dan akhirnya $4+3=7$. Karena pada percobaan keempat hasil penjumlahan kedua dadu bernilai 7, program keluar dari perulangan dan menampilkan pesan “You won after 4 tries!”, yang berarti pemain membutuhkan empat kali lemparan untuk mendapatkan total 7.

2.3 Program Game Penjumlahan

```
1 package pekan6_2511531015;
2 import java.util.Scanner;
3 import java.util.Random;
4 public class GamePenjumlahan_2511531015 {
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 3 wrong
10        int points = 0;
11        int wrong = 0;
12        while (wrong < 3) {
13            int result = play(console, rand);
14            //play one game
15            if (result > 0) {
16                points++;
17            } else {
18                wrong++;
19            }
20        }
21    }
22 System.out.println("You earned " + points + "total points.");
23 }
24 // membuat soal penjumlahan dan ditampilkan ke user
25 public static int play(Scanner console, Random rand) {
26     // print the operands being added, and sum them
27     int operands = rand.nextInt(4) + 2;
28     int sum = rand.nextInt(10) + 1;
29     System.out.print(sum);
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     // read user's guess and report whether it was correct
37     int guess = console.nextInt();
38     if (guess == sum) {
39         return 1;
40     } else {
41         System.out.println("Wrong! The answer was " + sum);
42         return 0;
43     }
44 }
```

Kode 2.3

Nama file: GamePenjumlahan_2511531015

1. Scanner console = new Scanner(System.in); dan Random rand = new Random(); membuat objek untuk membaca input pengguna dan menghasilkan angka acak.
2. int points = 0; int wrong = 0; menginisialisasi variabel untuk menghitung jumlah jawaban benar (points) dan jumlah jawaban salah (wrong).
3. while (wrong < 3) : Memulai perulangan utama, di mana permainan terus berlangsung selama jumlah jawaban salah masih kurang dari tiga.
4. Di dalam loop:
 - Program memanggil metode play(console, rand) untuk menampilkan soal dan membaca jawaban pengguna.
 - Jika hasil pengembalian (result) lebih besar dari 0 (berarti jawaban benar), poin bertambah satu.
 - Jika hasilnya 0 (jawaban salah), jumlah kesalahan bertambah satu.
5. Setelah pengguna salah tiga kali, perulangan berhenti.
6. System.out.println("You earned " + points + " total points."); menampilkan total poin yang berhasil diperoleh selama permainan.
7. Metode play() berfungsi untuk membuat soal penjumlahan acak:
 - Menentukan jumlah angka yang akan dijumlahkan antara 2 sampai 5.
 - Menghasilkan beberapa angka acak (1–10), menampilkannya, dan menghitung totalnya.
 - Pengguna diminta memasukkan hasil penjumlahan.
 - Jika benar, metode mengembalikan 1; jika salah, menampilkan jawaban yang benar dan mengembalikan 0.

Output:

```
5 + 10 + 2 + 4 + 4 =
20
Wrong! The answer was 25
4 + 7 + 6 + 1 + 1 = 19
3 + 1 + 4 + 4 = 8
Wrong! The answer was 12
4 + 9 + 10 + 3 = 25
Wrong! The answer was 26
You earned 1total points.
```

Output 2.3

Ketika program dijalankan, pengguna diberi beberapa soal penjumlahan acak yang terdiri dari dua hingga lima angka. Setiap kali soal muncul, pengguna memasukkan hasil penjumlahan ke dalam program. Jika jawaban benar, poin bertambah satu, sedangkan jika salah, program menampilkan jawaban yang benar dan menambah jumlah kesalahan. Pada contoh di atas, pengguna berhasil menjawab lima soal dengan benar sebelum melakukan tiga kesalahan berturut turut, yaitu pada soal “ $5 + 6 = 12$ ”, “ $1 + 8 + 10 + 4 + 7 = 0$ ”, dan “ $8 + 7 + 1 + 4 = 7$ ”, yang membuat perulangan berhenti. Setelah mencapai tiga kesalahan, program menampilkan pesan akhir “You earned 5 total points.” yang menunjukkan bahwa total skor pengguna selama permainan adalah lima poin.

2.4 Program Sentinel Loop

```
1 package pekan6_2511531015;
2 import java.util.Scanner;
3 public class SentinelLoop_2511531015 {
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
20}
21
```

Kode 2.4

Nama file: SentinelLoop_2511531015

1. Scanner console = new Scanner(System.in); membuat objek Scanner untuk membaca input dari pengguna melalui keyboard.
2. int sum = 0; int number = 12; menginisialisasi variabel sum untuk menyimpan total penjumlahan dan number sebagai nilai awal agar perulangan bisa dimulai.
3. while (number != 0) : memulai perulangan yang akan terus berjalan selama nilai number tidak sama dengan 0.
4. Di dalam perulangan:
 - Program menampilkan pesan “Masukan angka (0 untuk keluar): ” untuk meminta pengguna mengetik angka.
 - Angka yang dimasukkan pengguna dibaca dan disimpan ke variabel number.
 - Nilai number kemudian ditambahkan ke variabel sum.
5. Jika pengguna memasukkan angka 0, kondisi while (number != 0) menjadi salah, perulangan berhenti.
6. Setelah keluar dari loop, program menampilkan hasil akhir total penjumlahan semua angka yang dimasukkan pengguna melalui baris: System.out.println("totalnya adalah " + sum);

Output:

```
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 adalah15
```

Output 2.4

Ketika program dijalankan, pengguna diminta untuk memasukkan angka berulang kali melalui pesan “Masukan angka (0 untuk keluar)”. Setiap angka yang dimasukkan akan ditambahkan ke variabel sum, yang menyimpan total seluruh angka yang telah dimasukkan. Pada contoh di atas, pengguna memasukkan angka 3, 1, 2, 6, dan 7 secara berurutan, sehingga nilai totalnya menjadi $3 + 1 + 2 + 6 + 7 = 19$. Setelah itu, pengguna mengetik angka 0 sebagai tanda untuk berhenti, sehingga kondisi while (number != 0) menjadi salah dan perulangan berhenti. Program kemudian menampilkan hasil akhir dengan pesan “totalnya adalah 19”, yang merupakan jumlah keseluruhan dari semua angka sebelum 0 dimasukkan.

2.5 Program do while 1

```
1 package pekan6_2511531015;
2 import java.util.Scanner;
3 public class doWhile1_2511531015 {
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
```

Kode 2.5

Nama file: doWhile1_2511531015

1. Scanner console = new Scanner(System.in); membuat objek Scanner untuk membaca input dari keyboard.
2. String phrase; mendeklarasikan variabel phrase untuk menyimpan input dari pengguna.
3. do { ... } while (!phrase.equals("abcd")); memulai perulangan do-while, yaitu jenis perulangan yang selalu dijalankan minimal satu kali, meskipun kondisi akhirnya salah.
4. Di dalam blok do:
 - Program menampilkan pesan “Input Password: ” untuk meminta pengguna mengetik password.
 - Nilai yang dimasukkan pengguna dibaca menggunakan phrase = console.next();
5. Setelah pengguna mengetikkan password, program mengecek kondisi while (!phrase.equals("abcd")).
 - Jika password tidak sama dengan “abcd”, maka kondisi bernilai benar (true) dan perulangan diulang lagi.
 - Jika password sama dengan “abcd”, maka kondisi bernilai salah (false) dan perulangan berhenti.
6. Karena tidak ada perintah tambahan setelah loop, program langsung selesai ketika password benar dimasukkan.

Output:

```
Input Password: abcd
```

Output 2.5

Ketika program dijalankan, perulangan do while langsung menampilkan pesan “Input Password:” dan menunggu input dari pengguna. Pada contoh di atas, pengguna langsung mengetik “abcd” pada percobaan pertama. Karena kondisi !phrase.equals("abcd") bernilai salah (artinya input sudah benar), perulangan tidak dijalankan lagi, dan program berhenti setelah menerima password tersebut.

BAB 3

KESIMPULAN

3.1 Kesimpulan

Dari percobaan yang telah dilakukan, dapat disimpulkan bahwa perulangan while dan do while memiliki fungsi yang sama, yaitu menjalankan blok kode secara berulang selama kondisi tertentu masih bernilai benar. Namun, perbedaan utamanya terletak pada cara pengecekan kondisi. Pada perulangan while, kondisi diperiksa sebelum blok kode dijalankan, sehingga jika kondisi awal bernilai salah, perulangan tidak akan dijalankan sama sekali. Sebaliknya, pada do while, kondisi diperiksa setelah blok kode dijalankan, sehingga blok kode selalu dijalankan minimal satu kali meskipun kondisi awal bernilai salah. Dengan demikian, do while lebih cocok digunakan pada kasus di mana program harus mengeksekusi suatu perintah terlebih dahulu sebelum memeriksa kondisi, seperti pada proses input data atau validasi password.

3.2 Saran

Sebagai saran, sebaiknya penjelasan materi saat praktikum bisa lebih detail dan perlahan, supaya mahasiswa yang belum terlalu paham coding bisa mengikuti dengan baik. Selain itu, dosen atau asisten praktikum sebaiknya memberikan kriteria pelaksanaan tugas yang lebih jelas, agar mahasiswa mengetahui apa saja yang harus dikerjakan dan bagaimana penilaianya.

DAFTAR PUSTAKA

Oracle. (2025). *The while and do-while Statements (Java Tutorials)*. Oracle Documentation.
Diakses dari <https://docs.oracle.com/javase/tutorial/java/nutsandbolts/while.html>