LAPORAN PRAKTIKUM

PEMROGRAMAN ALGORITMA DAN PEMROGRAMAN

“PERULANGAN WHILE DAN DO WHILE”

disusun Oleh:

NAIRA RAMADHANI HALIL

2511533027

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 penulis panjatkan ke hadirat Tuhan Yang Maha Esa karena berkat rahmat dan karunia-Nya, laporan praktikum dengan judul "PERULANGAN WHILE DAN DO WHILE" dapat diselesaikan tepat waktu.

Laporan ini disusun untuk memenuhi salah satu tugas mata kuliah Praktikum Algoritma dan Pemrograman, sekaligus sebagai sarana pembelajaran dalam memahami konsep dasar pemrograman khususnya perulangan while dan do while pada Bahasa Java.

Pada kesempatan ini, penulis menyampaikan terimakasih kepada:

1. Bapak Dr. Wahyudi, S.T., M.T. selaku dosen pengampu mata kuliah Algoritma dan pemrograman yang telah memberikan bimbingan dan arahan.
2. Bang Jovantri Immanuel Gulo selaku asisten praktikum kelas A yang telah membantu pelaksanaan praktikum.
3. Teman-teman mahasiswa yang telah membantu dan memberi dukungan serta berdiskusi bersama dalam menyelesaikan laporan ini.

Penulis menyadari bahwa laporan ini masih jauh dari kata sempurna. Oleh karena itu, krirtik dan saran yang membangun sangat diharapkan demi penyempurnaan laporan di masa mendatang.

Akhir kata, semoga laporan ini dapat memberikan manfaat bagi pembaca dan menambah wawasan mengenai pemrograman Java.

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Di dunia pemrograman, perulangan (looping) merupakan salah satu konsep dasar yang sangat penting untuk mahasiswa informatika. Perulangan memungkinkan program mengeksekusi suatu blok perintah secara berulang tanpa repot-repot menuliskannya berkali-kali. Dengan menggunakan struktur perulangan, proses yang bersifat repetitive dapat dilakukan secara ototmatis dan efisien, sehingga memudahkan dalam pengembangan program yang dinamis.

Pada bahasa Java, terdapat tiga jenis perulangan utama, yaitu for, while, dan do while. Pada pertemuan kemarin (pekan ke-5) sudah membahasan tentang perulangan for yaitu perulangan yang digunakan ketika jumlah pengulangan sudah diketahui, kalau perulangan while dan do while digunakan ketika jumlah pengulangan belum diketahui secara pasti, dan bergantung pada suatu kondisi logika yang akan dihentikan apabila kondisi tersebut tidak lagi terpenuhi.

Perulangan while melakukan pemeriksaan kondisi terlebih dahulu sebelum menjalankan blok kode di dalamnya. Artinya, jika kondisi bernilai salah sejak awal, maka perulangan tidak akan dijalankan sama sekali. Sedangkan perulangan do while mengeksekusi blok kode terlebih dahulu baru kemudian memeriksa kondisinya. Dengan demikian, perulangan do while selalu dijalankan minimal satu kali meskipun kodisi bernilai salah.

1.2 Tujuan Praktikum

1. Memahami konsep dasar perulangan (looping) dalam bahasa Java.
2. Mengetahui struktur penulisan (syntax) dari perulangan while dan do while.
3. Mampu membedakan cara kerja antara while dan do while.
4. Dapat menerapkan kedua jenis perulangan dalam berbagai contoh kasus pemrograman.
5. Melatih kemampuan logika dan analisis dalam memecahkan masalah yang melibatkan proses berulang.

1.3 Manfaat Praktikum

1. Menambah pemahaman mengenai struktur control dalam Java, khususnya while dan do while.
2. Melatih kemampuan mahasiswa dalam membuat program yang dinamis dan interaktif.
3. Membiasakan mahasiswa menggunakan logika kondisi (boolean) untuk mengatur alur program.
4. Menjadi dasar untuk memahami konsep perulangan bersarang, validasi input, dan struktur data.
5. Meningkatkan keterampilan pemrograman melalui penerapan pengulangan pada berbagai studi kasus.

BAB II
PEMBAHASAN

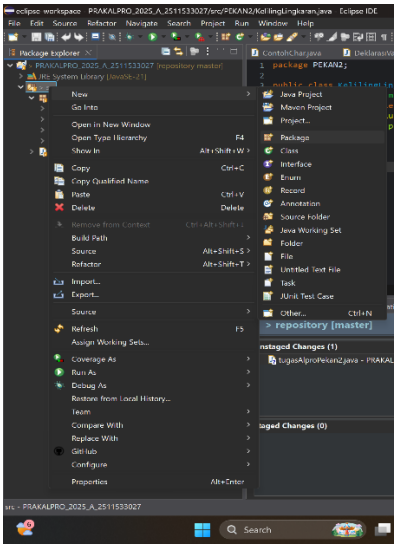
2.1 Langkah Kerja Praktikum

1. Persiapan Awal

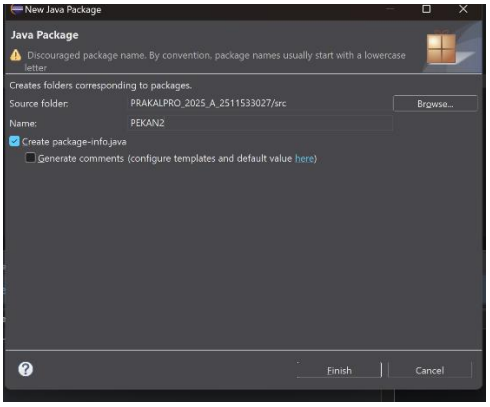
   a) Buka aplikasi Eclipse IDE di laptop atau di komputer.

2. Membuat Package

   a) Klik kanan pada folder **src**, lalu pilih **New**, terakhir klik **Package**.

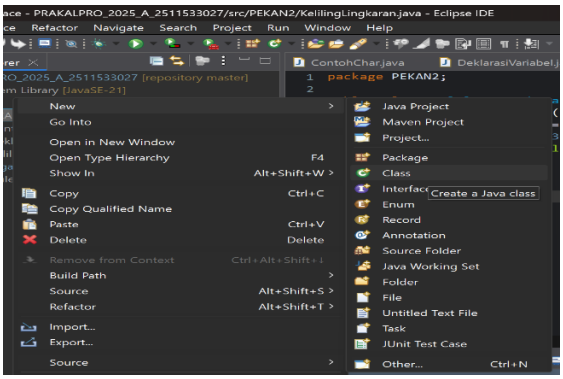


   b) Beri nama package **PEKAN6_2511533027**.



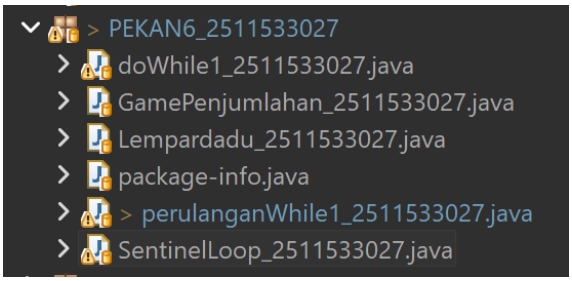
3. Membuat Class Baru untuk Setiap Percobaan

a) Klik kanan pada package **PEKAN6**, lalu pilih **New**, terakhir klik **Class**.



b) Buat beberapa class sesuai percobaan.

- doWhile1_2511533027.java
- GamePenjumlahan_2511533027.java
- Lempardadu_2511533027.java
- perulanganWhile1_2511533027.java
- SentinelLoop_2511533027.java



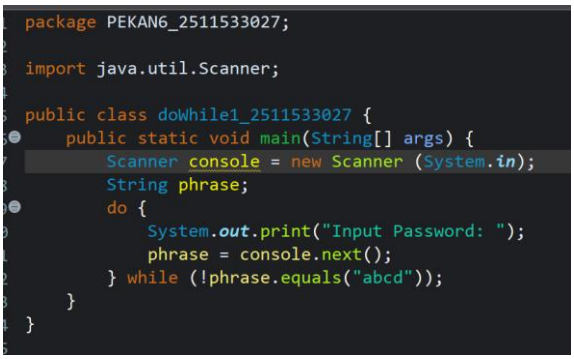
c) Saat membuat class, centang opsi **public static void main(String[] args)** agar otomatis ada fungsi **main**.

4. Menulis Kode Program

a) doWhile1_2511533027.java

- Menuliskan program menggunakan **perulangan do while** untuk meminta pengguna memasukkan password.
- Menggunakan **kelas Scanner** untuk membaca input dari keyboard.
- Variabel phrase bertipe String digunakan untuk menyimpan input dari pengguna.
- Program akan meminta input berulang kali sampai pengguna mengetik **"abcd"**.
- Kondisi pada bagian while menggunakan **operator logika NOT (!)** agar perulangan terus berjalan selama input belum sama dengan kata sandi yang benar.

- Setelah input sesuai, program berhenti dan tidak meminta input lagi.



```java
package PEKAN6_2511533027;

import java.util.Scanner;

public class doWhile1_2511533027 {
    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"));
    }
}
```

b) GamePenjumlahan_2511533027.java

- Menuliskan program menggunakan **perulangan while** untuk membuat permainan penjumlahan sederhana.
- Program akan terus berjalan sampai pengguna menjawab **salah sebanyak 3 kali**.
- Menggunakan **kelas Scanner** untuk membaca input dari pengguna.
- Menggunakan **kelas Random** untuk menghasilkan bilangan acak pada setiap soal penjumlahan.
- Variabel **points** digunakan untuk menyimpan jumlah jawaban benar, dan variabel **wrong** untuk menghitung jumlah kesalahan.
- Pemanggilan method `play(console, rand)` dilakukan di dalam perulangan untuk menjalankan satu soal penjumlahan setiap iterasi.
- Jika pengguna menjawab benar (`result > 0`), maka nilai `points` akan bertambah satu, jika salah maka `wrong` bertambah satu.
- Perulangan akan berhenti ketika nilai `wrong` sudah mencapai 3.
- Setelah permainan selesai, program menampilkan total poin yang diperoleh pemain.

```java
package PEKAN6_2511533027;

import java.util.Random;

public class GamePenjumlahan_2511533027 {
    public static void main(String[] args) {
        Scanner console = new Scanner(System.in);
        Random rand = new Random();
        //play until user 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);
```

c) Lempardadu_2511533027.java

- Menuliskan program menggunakan **perulangan while** untuk melakukan simulasi permainan lempar dadu.
- Menggunakan **kelas Random** untuk menghasilkan dua angka acak antara 1 sampai 6 yang mewakili hasil lemparan dua buah dadu.
- Variabel **tries** digunakan untuk menghitung jumlah percobaan yang dilakukan.
- Variabel **sum** digunakan untuk menyimpan hasil penjumlahan kedua dadu.
- Program akan terus melempar dadu selama hasil penjumlahan kedua dadu **tidak sama dengan 7**.
- Pada setiap iterasi, nilai kedua dadu (**dadu1** dan **dadu2**) dicetak ke layar bersama hasil jumlahnya menggunakan **System.out.println()**.
- Setelah total dadu bernilai **7**, perulangan berhenti, dan program menampilkan jumlah percobaan yang dibutuhkan untuk mendapatkan angka 7.

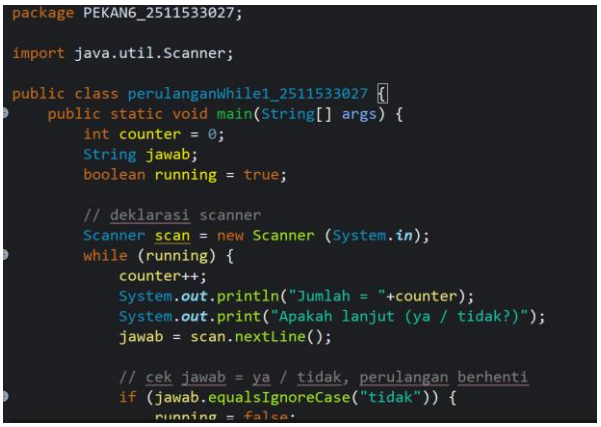
```
public class Lempardadu_2511533027 {
    public static void main(String[] args) {
        Random rand = new Random();
        int tries = 0;
        int sum = 0;
        while (sum != 7) {

            // rol 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! ");
    }
}
```

d) perulanganWhile1_2511533027.java

- Menuliskan program menggunakan **perulangan while** untuk melakukan penghitungan secara berulang berdasarkan input dari pengguna.
- Menggunakan kelas Scanner untuk membaca input teks dari keyboard.

- Variabel **counter** digunakan untuk menghitung jumlah pengulangan yang telah dilakukan.
- Variabel **jawab** bertipe **String** berfungsi menyimpan jawaban pengguna (“ya” atau “tidak”).
- Variabel **running** bertipe **boolean** digunakan sebagai kondisi kontrol dalam perulangan.
- Di dalam blok **while**, setiap kali perulangan berjalan, nilai counter akan bertambah satu (**counter++**).
- Program menampilkan nilai **counter** menggunakan perintah:
  System.out.println("Jumlah = " + counter);
- Setelah itu, program menanyakan kepada pengguna apakah ingin melanjutkan perulangan dengan pesan:
  System.out.print("Apakah lanjut (ya / tidak?)");
- Pengguna mengetikkan jawaban melalui **scan.nextLine()**.
- Jika pengguna mengetik “tidak”, maka kondisi running diubah menjadi false sehingga perulangan berhenti.
- Jika pengguna menjawab “ya”, maka perulangan akan terus berlanjut dan counter akan bertambah lagi.



```
package PEKAN6_2511533027;

import java.util.Scanner;

public class perulanganWhile1_2511533027 {
    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.print("Apakah lanjut (ya / tidak?)");
            jawab = scan.nextLine();

            // cek jawab = ya / tidak, perulangan berhenti
            if (jawab.equalsIgnoreCase("tidak")) {
                running = false;
```

e) SentinelLoop_2511533027.java

- Menuliskan program menggunakan **perulangan while** dengan konsep **sentinel (penanda berhenti)** untuk menjumlahkan serangkaian angka yang dimasukkan oleh pengguna.
- Menggunakan kelas Scanner untuk membaca input bilangan dari keyboard.

- Variabel sum digunakan untuk menyimpan hasil penjumlahan dari seluruh angka yang dimasukkan pengguna.
- Variabel number diberi nilai awal (dummy value) agar perulangan dapat dijalankan pertama kali.
- Kondisi perulangan while (number != 0) memastikan program akan terus meminta input selama pengguna tidak mengetik angka 0.
- Di dalam perulangan, program menampilkan pesan: System.out.print("Masukkan angka (0 untuk keluar): ");
- Setelah pengguna memasukkan angka, nilainya dibaca dengan console.nextInt() dan langsung ditambahkan ke variabel sum.
- Jika pengguna memasukkan angka 0, kondisi while menjadi salah (false) dan perulangan berhenti.
- Setelah keluar dari perulangan, program menampilkan hasil total penjumlahan semua angka yang telah dimasukkan.

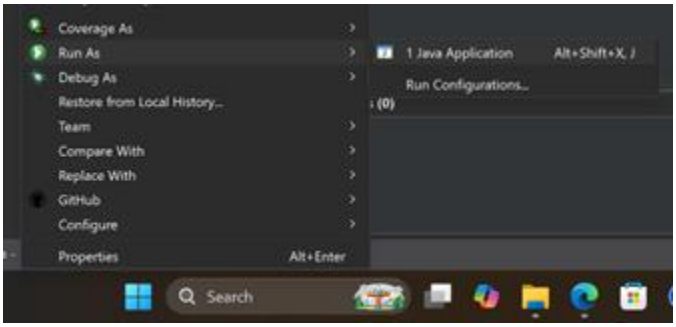
```java
package PEKAN6_2511533027;

import java.util.Scanner;

public class SentinelLoop_2511533027 {
    public static void main(String[] args) {
        Scanner console = new Scanner (System.in);
        int sum = 0;
        int number = 12; // "dummy value", anything but 0

        while (number != 0) {
            System.out.print("Masukkan angka (0 untuk keluar): ");
            number = console.nextInt();
            sum = sum + number;
        }
        System.out.println("totalnya adalah " + sum);
    }
}
```

5. Kompilasi dan Menjalankan Program.
   a) Klik kanan pada file program, pilih **Run As**, terakhir klik **Java Application**.
   b) Lihat hasil output pada tab Console.



2.2 Analisis Hasil dan Pembahasan

1. doWhile1_2511533027.java

a) Hasil: Program meminta pengguna untuk memasukkan password berulang kali hingga input yang dimasukkan sama dengan “abcd”. Program akan berhenti setelah pengguna mengetikkan password yang benar.
b) Analisis: Program menggunakan struktur **perulangan do while**, yang mengeksekusi blok perintah terlebih dahulu sebelum melakukan pengecekan kondisi. Hal ini menyebabkan program selalu meminta input minimal satu kali. Kondisi !phrase.equals("abcd") memastikan bahwa perulangan akan terus berulang selama password yang dimasukkan belum benar.
c) Teori Pendukung: Menurut **Oracle Java SE Documentation (2023)**, perulangan do while digunakan ketika perintah harus dijalankan setidaknya satu kali sebelum kondisi diperiksa. Struktur ini cocok untuk validasi input pengguna seperti konfirmasi password.

2. GamePenjumlahan_2511533027.java
   a) Hasil: Program memberikan beberapa soal penjumlahan secara acak kepada pengguna. Jika pengguna salah menjawab tiga kali, permainan berakhir dan jumlah skor ditampilkan di layar.
   b) Analisis: Program menggunakan **perulangan while** untuk menjalankan permainan selama jumlah kesalahan (wrong) belum mencapai tiga. Pada setiap iterasi, method play() dipanggil untuk menghasilkan soal penjumlahan acak dan mengecek jawaban pengguna. Jika jawaban benar, nilai points bertambah satu; jika salah, variabel wrong bertambah. Program berhenti ketika wrong == 3.
   c) Teori Pendukung: Menurut **Schildt (2021)** dalam *Java: The Complete Reference*, perulangan while sangat efektif digunakan dalam kasus ketika jumlah iterasi tidak diketahui sebelumnya dan berhenti berdasarkan kondisi tertentu. Penggunaan method terpisah seperti play() juga membantu menjaga modularitas program.
3. Lempardadu_2511533027.java
   a) Hasil: Program melempar dua buah dadu acak secara berulang hingga jumlah kedua dadu sama dengan angka 7. Setelah itu, program menampilkan berapa kali percobaan yang dibutuhkan untuk mendapatkan hasil 7.
   b) Analisis: Program memanfaatkan **perulangan while** dengan kondisi sum != 7, yang berarti program terus berjalan sampai hasil penjumlahan dua dadu sama dengan 7. Pada setiap iterasi, dua angka acak antara 1–6 dihasilkan menggunakan rand.nextInt(6) + 1. Nilai jumlah kedua dadu ditampilkan, dan variabel tries

bertambah setiap kali perulangan dilakukan. Program berhenti ketika sum bernilai 7.

c) Teori Pendukung: Menurut **Java Platform SE Documentation (Oracle, 2023)**, kelas Random digunakan untuk menghasilkan angka acak dalam berbagai keperluan seperti simulasi atau permainan. Struktur while cocok digunakan ketika kondisi berhenti baru diketahui setelah proses dijalankan beberapa kali.

4. perulanganWhile1_2511533027.java
   a) Hasil: Program menampilkan jumlah pengulangan (counter) dan menanyakan kepada pengguna apakah ingin melanjutkan proses atau tidak. Jika pengguna mengetikkan “tidak”, maka program berhenti.
   b) Analisis: Program menggunakan **perulangan while** yang dikontrol oleh variabel boolean running. Selama nilai running bernilai true, perulangan akan terus berjalan. Pada setiap iterasi, program menambah nilai counter dan menanyakan kepada pengguna apakah ingin melanjutkan. Jika pengguna mengetikkan “tidak”, maka nilai running diubah menjadi false dan perulangan berhenti.
   c) Teori Pendukung: Menurut **GeeksforGeeks (2024)**, perulangan while sangat sesuai untuk situasi di mana jumlah iterasi tidak diketahui dan dikontrol oleh kondisi logika yang bergantung pada input pengguna.
5. SentinelLoop_2511533027.java
   a) Hasil: Program meminta pengguna untuk memasukkan sejumlah angka, kemudian menjumlahkan semua angka tersebut hingga pengguna memasukkan angka 0 sebagai penanda berhenti (sentinel value).
   b) Analisis: Program menggunakan **perulangan while** dengan konsep sentinel loop. Variabel number diisi nilai awal non-nol agar perulangan dapat berjalan pertama kali. Setiap angka yang dimasukkan akan ditambahkan ke dalam variabel sum. Ketika pengguna memasukkan angka 0, kondisi while (number != 0) menjadi salah, dan perulangan berhenti. Program kemudian menampilkan total hasil penjumlahan seluruh angka.
   c) Teori Pendukung: Menurut **Oracle (2023)** dan **PetaniKode (2024)**, sentinel loop digunakan untuk mengendalikan perulangan berdasarkan nilai penanda (sentinel) tertentu. Struktur ini umum digunakan pada proses input data yang tidak diketahui jumlahnya di awal.

BAB III
PENUTUP

3.1 Kesimpulan

Berdasarkan praktikum yang telah dilakukan pada materi **Perulangan While dan Do While**, dapat disimpulkan bahwa perulangan merupakan salah satu struktur kontrol penting dalam pemrograman yang berfungsi untuk mengeksekusi perintah secara berulang hingga kondisi tertentu terpenuhi.

Perulangan **while** digunakan ketika jumlah pengulangan **belum diketahui sebelumnya**, dan program akan terus berjalan selama kondisi bernilai benar. Kondisi diperiksa terlebih dahulu sebelum blok perintah dijalankan, sehingga apabila kondisi awal bernilai salah, maka perulangan tidak akan dijalankan sama sekali.

Sedangkan perulangan **do while** hampir sama dengan while, namun memiliki perbedaan pada urutan pengecekan kondisi. Pada do while, blok perintah dijalankan terlebih dahulu, baru setelah itu kondisi diperiksa. Hal ini menyebabkan do while **selalu dijalankan minimal satu kali** walaupun kondisi awal bernilai salah.

Dari beberapa percobaan yang telah dilakukan (dowhile1_2511533027.java, GamePenjumlahan_2511533027.java, LemparDadu_2511533027.java, perulanganWhile1_2511533027.java, dan SentinelLoop_2511533027.java), diperoleh pemahaman bahwa:

1. Struktur perulangan **while** cocok digunakan untuk proses yang bergantung pada kondisi logika dan belum diketahui jumlah pengulangannya.
2. Struktur perulangan **do while** digunakan ketika program harus dijalankan setidaknya sekali sebelum kondisi diperiksa.
3. Perulangan dapat dikombinasikan dengan input pengguna untuk membuat program yang lebih interaktif.
4. Penggunaan **Scanner** dan **Random** memperkaya variasi percobaan dengan input acak dan dinamis.
5. Penerapan perulangan while dan do while memungkinkan programmer membuat program yang efisien, adaptif, serta mudah dikembangkan untuk berbagai kasus seperti validasi data, permainan, maupun perhitungan matematis.

Melalui praktikum ini, mahasiswa diharapkan dapat memahami dan menerapkan konsep **perulangan while dan do while** secara efektif dalam menyelesaikan permasalahan pemrograman yang bersifat dinamis dan berulang.

DAFTAR PUSTAKA

[1] Wahyudi, *Perulangan While dan Do While di Java*, presentasi kuliah, Program Studi Informatika, Universitas Andalas, 2025.

[2] Oracle Corporation. *Java Platform, Standard Edition 21 Documentation*, 2023. [Online]. Tersedia di: https://docs.oracle.com/javase/21/docs/api/

[3] H. Schildt, *Java: The Complete Reference*, 12th ed., New York, NY, USA: McGraw-Hill, 2021.

[4] GeeksforGeeks. "Loops in Java," 2024. [Online]. Tersedia di: https://www.geeksforgeeks.org/java/loops-in-java/

[5] W3Schools. "Java While and Do While Loop," 2024. [Online]. Tersedia di: https://www.w3schools.com/java/java_while_loop.asp

[6] PetaniKode. "Belajar Java: Perulangan While dan Do While," 2024. [Online]. Tersedia di: https://www.petanikode.com/java-while-dowhile/

[7] MalasNgoding. "Perulangan While dan Do While pada Java," 2024. [Online]. Tersedia di: https://www.malasngoding.com/perulangan-while-dan-do-while-pada-java/