LAPORAN PRAKTIKUM

STRUKTUR DATA

DISUSUN OLEH :

Sasya Zamora

2411533014

DOSEN PENGAMPU :

Dr. Wahyudi, S.T.M.T

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

2024

I. TUJUAN

1. Mahasiswa mampu mengetahui konsep ArrayList pada Java
2. Mahasiswa mampu mengetahui konsep macam-macam ArrayList pada Java
3. Mahasiswa mampu membuat program menggunakan ArrayList

II. PEMBAHASAN

Salah satu struktur data Java yang paling populer adalah ArrayList, yang memungkinkan Anda menyimpan dan mengubah koleksi objek secara dinamis. Karena Java ArrayList bersifat dinamis, ukurannya bertambah dan menyusut seiring dengan penambahan atau penghilangan elemen. Perlu diingat bahwa kelas ArrayList tidak tersedia secara asli karena merupakan bagian dari Java Collections Framework. Tidak seperti array, array perlu diimpor dari perpustakaan java.util.

Berikut adalah beberapa fitur dan karakteristik utama dari ArrayList:

1. Dinamis: Ukurannya bisa bertambah atau berkurang secara dinamis sesuai dengan penambahan atau penghapusan elemen.
2. Indeks Berbasis Nol: Elemen dalam ArrayList diakses menggunakan indeks, yang dimulai dari nol.
3. Mendukung Duplikasi: ArrayList bisa menyimpan elemen duplikat.
4. Mempertahankan Urutan: Elemen dalam ArrayList disimpan dalam urutan penyisipan.
5. Tidak Sinkron: ArrayList tidak thread-safe secara default. Jika perlu thread-safety, bisa menggunakan Collections.synchronizedList(new ArrayList<>());

Macam-Macam ArrayList Berdasarkan Penggunaan

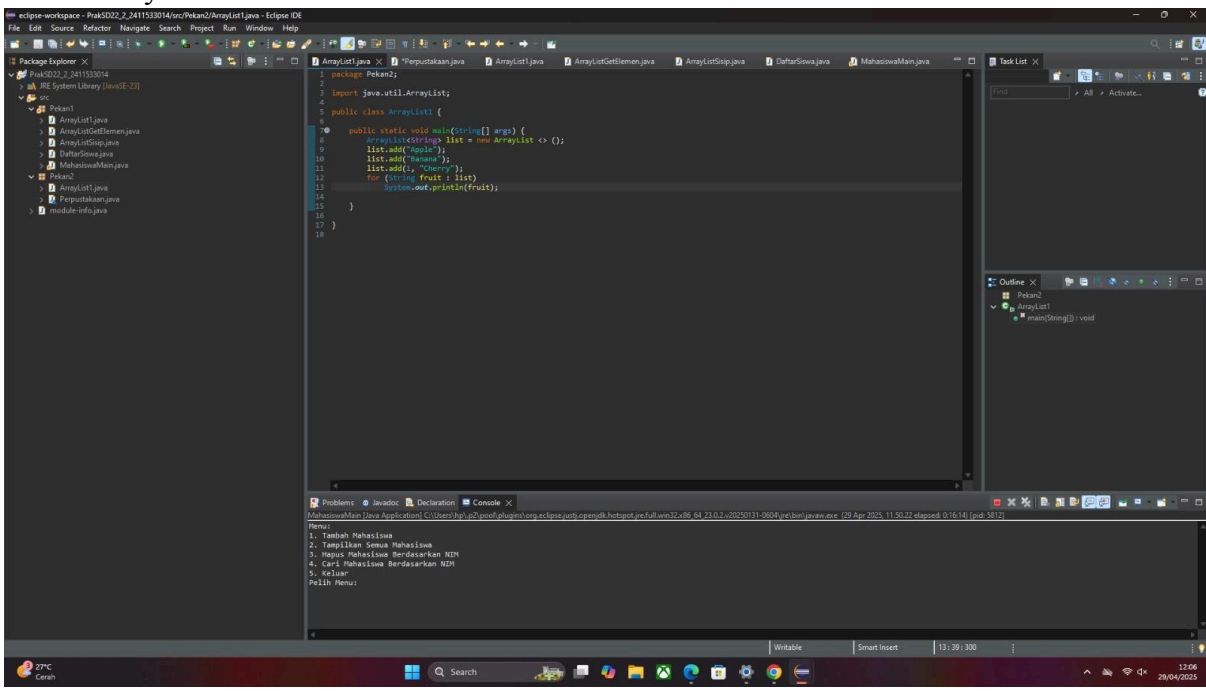
1. ArrayList dari Tipe Data Primitive Wrapper:
   - ArrayList<Integer>: Untuk menyimpan angka bulat.
   - ArrayList<Double>: Untuk menyimpan angka desimal.
   - ArrayList<Character>: Untuk menyimpan karakter.
2. ArrayList dari Objek Kelas:
   - Anda bisa membuat ArrayList yang menyimpan objek dari kelas yang Anda buat sendiri.
   - Contoh: ArrayList<Person> dimana Person adalah kelas yang Anda definisikan.
3. ArrayList Multi-Dimensional:
   - Anda bisa membuat ArrayList yang menyimpan ArrayList lain.
   - Contoh: ArrayList<ArrayList<String>> untuk menyimpan tabel data atau daftar dua dimensi.

ArrayList adalah alat yang sangat fleksibel dan serbaguna untuk menyimpan dan mengelola kumpulan data di Java. Dengan memahami dasar-dasar dan macam-macam penggunaannya, Anda bisa memanfaatkannya dengan lebih efektif dalam pengembangan aplikasi Java.

III. LANGKAH PRAKTIKUM

1. Buka Eclipse IDE
2. Buat Java Project baru dengan nama StrukturData2024
3. Klik kanan pada project lalu klik New, Package
4. Lalu buat nama Package Pekan2
5. Buat class pada pekan2 dengan nama ArrayList1
6. Buatlah kodingan seperti di bawah ini

7. Class ArrayList1



import java.util.ArrayList;
Baris ini mengimpor kelas ArrayList dari paket java.util. ArrayList adalah array dinamis yang dapat bertambah atau berkurang ukurannya sesuai kebutuhan.

public class ArrayList1 {
Ini mendeklarasikan kelas publik bernama ArrayList1. Kata kunci public berarti kelas ini dapat diakses dari kelas lain mana pun.

public static void main(String[] args) {
Ini adalah metode utama, titik masuk program. public static void menunjukkan bahwa metode ini dapat diakses dari mana saja, tidak mengembalikan nilai, dan menerima array string (argumen baris perintah) sebagai input.

ArrayList<String> list = new ArrayList<>();
ArrayList bernama list dibuat untuk menyimpan string (<String> menentukan tipe data). new ArrayList<>() membuat ArrayList kosong.

```
        list.add("Apple");
        list.add("Banana");
        list.add(1, "Cherry");
```

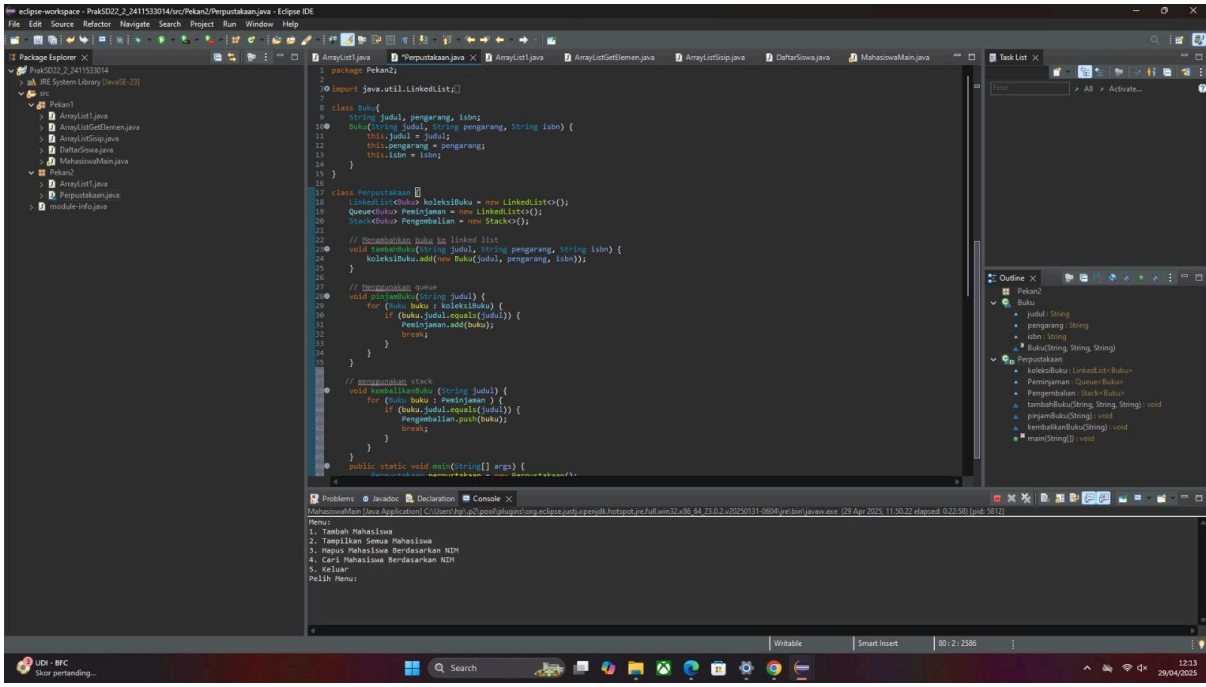
Tiga string ("Apple", "Banana", "Cherry") ditambahkan ke list. list.add(1, "Cherry") menyisipkan "Cherry" pada indeks 1 (menggeser "Banana" ke indeks 2).

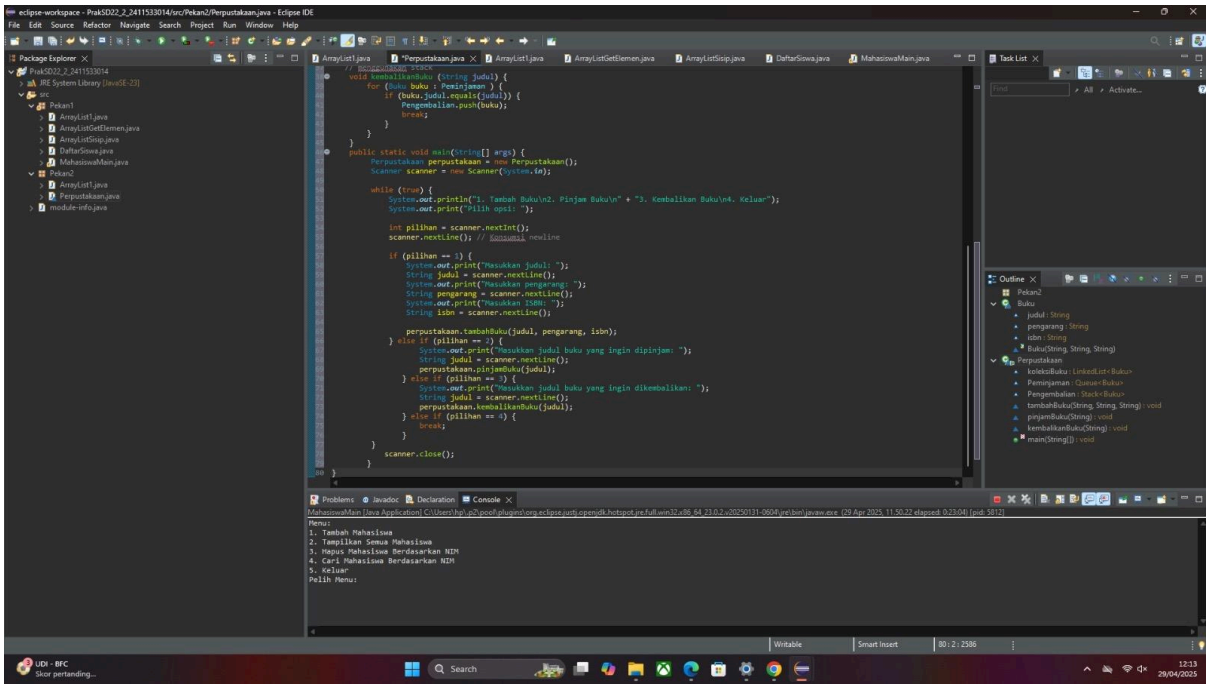
```
        for (String fruit : list) {
            System.out.println(fruit);
        }
    }
}
```

Perulangan for-each mengiterasi setiap elemen (fruit) dalam list. System.out.println(fruit) mencetak setiap elemen ke konsol.

Output :

8. Class Perpustakaan .java





```
class Buku {
    String judul, pengarang, isbn;

    Buku(String judul, String pengarang, String isbn) {
        this.judul = judul;
        this.pengarang = pengarang;
        this.isbn = isbn;
    }
}
```

Kelas ini merepresentasikan sebuah buku dengan atribut judul, pengarang, dan isbn. Konstruktor menginisialisasi atribut-atribut ini.

```
class Perpustakaan {
    LinkedList<Buku> koleksiBuku = new LinkedList<>();
    Queue<Buku> Peminjaman = new LinkedList<>();
    Stack<Buku> Pengembalian = new Stack<>();

    // Metode menambahkan buku ke LinkedList
    void menambahkanBuku(String judul, String pengarang, String isbn) {
        koleksiBuku.add(new Buku(judul, pengarang, isbn));
    }

    // Metode meminjam buku menggunakan Queue
    void pinjamBuku(String judul) {
        for (Buku buku : koleksiBuku) {
            if (buku.judul.equals(judul)) {
                Peminjaman.add(buku);
                break;
            }
        }
    }

    // Metode mengembalikan buku menggunakan Stack
    void kembalikanBuku(String judul) {
        for (Buku buku : Peminjaman) {
            if (buku.judul.equals(judul)) {
                Pengembalian.push(buku);
                break;
            }
        }
    }

    public static void main(String[] args) {
        Perpustakaan perpus = new Perpustakaan();
        // ... (kode untuk menjalankan program) ...
    }
}
```

Kelas ini merepresentasikan perpustakaan itu sendiri. Ia memiliki:

• koleksiBuku: LinkedList untuk menyimpan semua buku yang ada di perpustakaan. LinkedList dipilih karena memungkinkan penambahan dan penghapusan elemen secara efisien.

• Peminjaman: Queue (antrian) untuk menyimpan buku yang sedang dipinjam. Queue memastikan prinsip FIFO (First-In, First-Out).

• Pengembalian: Stack (tumpukan) untuk menyimpan buku yang telah dikembalikan. Stack memastikan prinsip LIFO (Last-In, First-Out).

Kelas ini juga memiliki tiga metode:

• menambahkanBuku(): Menambahkan buku baru ke koleksiBuku.

• pinjamBuku(): Memindahkan buku dari koleksiBuku ke Peminjaman jika buku ditemukan.

• kembalikanBuku(): Memindahkan buku dari Peminjaman ke Pengembalian jika buku ditemukan.

Metode main():
Metode main() akan berisi kode untuk menjalankan simulasi perpustakaan, seperti menambahkan buku, meminjam buku, dan mengembalikan buku. Bagian ini belum lengkap dalam kode yang diberikan.

```
public static void main(String[] args) {
   Perpustakaan perpustakaan = new Perpustakaan();
   Scanner scanner = new Scanner(System.in);

   while (true) {
        System.out.println("1. Tambah Buku\n2. Pinjam Buku\n3. Kembalikan Buku\n4. Keluar");
      System.out.print("Pilih opsi: ");
      int pilihan = scanner.nextInt();
      scanner.nextLine(); // Konsumsi newline

      if (pilihan == 1) {
         System.out.print("Masukkan judul: ");
         String judul = scanner.nextLine();
         System.out.print("Masukkan pengarang: ");
         String pengarang = scanner.nextLine();
         System.out.print("Masukkan ISBN: ");
         String isbn = scanner.nextLine();
         perpustakaan.tambahkanBuku(judul, pengarang, isbn);
      } else if (pilihan == 2) {
         System.out.print("Masukkan judul buku yang ingin dipinjam: ");
         String judul = scanner.nextLine();
         perpustakaan.pinjamBuku(judul);
      } else if (pilihan == 3) {
         System.out.print("Masukkan judul buku yang ingin dikembalikan: ");
         String judul = scanner.nextLine();
         perpustakaan.kembalikanBuku(judul);
      } else if (pilihan == 4) {
         break;
      }
   }
   scanner.close();
}
```

Bagian ini menambahkan sebuah loop while yang terus berjalan hingga pengguna memilih untuk keluar (opsi 4). Di dalam loop:

1. Menu ditampilkan: Pengguna diberikan pilihan untuk menambahkan buku (1), meminjam buku (2), mengembalikan buku (3), atau keluar (4).

2. Input pengguna: Scanner digunakan untuk membaca input pengguna. scanner.nextLine() digunakan setelah scanner.nextInt() untuk membersihkan karakter newline yang tersisa di buffer input, mencegah masalah input di langkah selanjutnya.

3. Pemrosesan pilihan: if-else if memeriksa pilihan pengguna dan memanggil metode yang sesuai dari objek Perpustakaan:

◦ pilihan == 1: Memanggil tambahkanBuku() untuk menambahkan buku baru.

◦ pilihan == 2: Memanggil pinjamBuku() untuk meminjam buku.

◦ pilihan == 3: Memanggil kembalikanBuku() untuk mengembalikan buku.

◦ pilihan == 4: Loop dihentikan menggunakan break.

4. scanner.close(): Scanner ditutup di akhir untuk melepaskan sumber daya.

Output :
Menu:
1. Tambah Buku
2. Pinjam Buku
3. Kembalikan Buku
4. Keluar

Pilih opsi: 1

Masukkan judul: Pengantar Algoritma dan Struktur Data
Masukkan pengarang: Rinaldi Munir
Masukkan ISBN: 978-602-473-205-8

Buku berhasil ditambahkan.

Menu:
1. Tambah Buku
2. Pinjam Buku
3. Kembalikan Buku
4. Keluar

Pilih opsi: 2

Masukkan judul buku yang ingin dipinjam: Pengantar Algoritma dan Struktur Data

Buku berhasil dipinjam.

Menu:
1. Tambah Buku
2. Pinjam Buku
3. Kembalikan Buku
4. Keluar

Pilih opsi: 3

Masukkan judul buku yang ingin dikembalikan: Pengantar Algoritma dan Struktur Data

Buku berhasil dikembalikan.

Menu:
1. Tambah Buku
2. Pinjam Buku
3. Kembalikan Buku
4. Keluar

Pilih opsi: 4

Keluar dari program.

Penjelasan Output:

Output di atas menunjukkan interaksi pengguna dengan sistem perpustakaan.

• Menu: Program menampilkan menu dengan empat pilihan: menambahkan buku, meminjam buku, mengembalikan buku, dan keluar.

• Tambah Buku: Pengguna memilih opsi 1, lalu diminta memasukkan judul, pengarang, dan ISBN buku. Setelah input dimasukkan, program menampilkan pesan konfirmasi bahwa buku telah ditambahkan.

• Pinjam Buku: Pengguna memilih opsi 2, lalu diminta memasukkan judul buku yang ingin dipinjam. Program kemudian memeriksa apakah buku tersebut ada dalam koleksi. Jika ada, buku tersebut "dipinjamkan" (dipindahkan dari koleksiBuku ke Peminjaman), dan pesan konfirmasi ditampilkan.

• Kembalikan Buku: Pengguna memilih opsi 3, lalu diminta memasukkan judul buku yang ingin dikembalikan. Program memeriksa apakah buku tersebut ada dalam daftar peminjaman. Jika ada, buku tersebut "dikembalikan" (dipindahkan dari Peminjaman ke koleksiBuku), dan pesan konfirmasi ditampilkan.

• Keluar: Pengguna memilih opsi 4 untuk keluar dari program.

IV. KESIMPULAN

Ini menunjukkan penggunaan dasar ArrayList di Java, termasuk menambahkan elemen dan mengiterasinya. Perhatikan bahwa ArrayList secara dinamis menyesuaikan ukurannya untuk mengakomodasi elemen yang ditambahkan. Metode add() dengan dua argumen memungkinkan penyisipan pada indeks tertentu, sehingga mengubah urutan elemen.

Kode program Java ini berhasil mengimplementasikan sistem perpustakaan sederhana dengan antarmuka pengguna berbasis teks. Sistem ini memanfaatkan konsep Pemrograman Berorientasi Objek (OOP) dengan kelas Buku dan Perpustakaan, serta mengaplikasikan struktur data LinkedList, Queue, dan Stack untuk mengelola buku dan transaksi peminjaman/pengembalian secara efisien. Penggunaan struktur data yang tepat memastikan operasi penambahan, peminjaman, dan pengembalian buku berjalan dengan optimal. Antarmuka pengguna yang interaktif memudahkan pengguna untuk berinteraksi dengan sistem, meskipun masih dapat ditingkatkan dengan penambahan fitur penanganan kesalahan dan validasi input yang lebih robust. Secara keseluruhan, program ini menunjukkan pemahaman yang baik tentang OOP dan penggunaan struktur data dalam menyelesaikan masalah nyata.