Laporan Praktikum
Struktur Data Pekan 2

Disusun Oleh :

SASKIA ALIFAH

2411531002

Dosen Pengampu : Dr. Wahyudi, S.T, M.T

Departemen Informatika

Fakultas Teknologi Informasi

Universitas Andalas

Tahun 2025

Laporan Pratikum Pekan 2

A. Tujuan Praktikum

1. Memahami cara kerja dan penggunaan struktur data dinamis seperti ArrayList, LinkedList, Queue, dan Stack dalam bahasa pemrograman Java.Memahami
2. Mampu membuat program berbasis objek yang memanfaatkan struktur data tersebut dalam studi kasus sistem perpustakaan.
3. Mengetahui perbedaan karakteristik setiap struktur data dan penerapannya sesuai kebutuhan program.

B. Pendahuluan

Struktur data adalah cara untuk menyimpan dan mengatur data agar operasi tertentu dapat dilakukan secara efisien. Dalam Java, tersedia berbagai struktur data yang dapat digunakan sesuai dengan kebutuhan, antara lain:

1. ArrayList

ArrayList merupakan salah satu implementasi dari struktur data list di Java yang berada dalam paket java.util. Berbeda dengan array biasa yang bersifat statis, ArrayList bersifat dinamis, artinya ukuran dari list ini dapat berubah-ubah sesuai dengan jumlah elemen yang disimpan. Saat elemen baru ditambahkan melebihi kapasitas awal, ArrayList akan secara otomatis membuat array baru yang lebih besar dan menyalin elemen lama ke dalamnya. Hal ini memudahkan pengelolaan data karena tidak perlu menentukan jumlah elemen secara pasti di awal.Selain fleksibilitas ukuran, ArrayList juga menyediakan berbagai metode bawaan yang sangat berguna, seperti add(), remove(), get(), set(), dan contains(). Elemen-elemen dalam ArrayList disimpan secara berurutan dan dapat diakses melalui indeks. Hal ini membuat ArrayList sangat cocok digunakan ketika operasi pencarian dan pengambilan elemen secara acak (random access) sering dilakukan, karena ArrayList mendukung operasi ini dengan kecepatan konstan.

Namun, ArrayList memiliki kelemahan dalam hal penambahan dan penghapusan elemen di posisi awal atau tengah list. Karena elemen disimpan dalam array, setiap kali ada elemen yang ditambahkan atau dihapus di tengah, maka seluruh elemen setelahnya perlu digeser satu posisi. Oleh karena itu, meskipun ArrayList sangat efisien untuk akses data, penggunaannya kurang optimal jika program sering melakukan penyisipan atau penghapusan elemen di posisi acak.

2. LinkedList

LinkedList adalah struktur data linier yang terdiri dari serangkaian node yang saling terhubung satu sama lain. Setiap node pada LinkedList berisi data dan referensi (atau pointer) ke node berikutnya. Java menyediakan kelas LinkedList dalam paket java.util yang dapat digunakan untuk membuat list dinamis yang efisien dalam operasi penambahan dan penghapusan data, terutama di bagian awal dan akhir list.Kelebihan utama dari LinkedList terletak pada kemampuannya untuk menambah atau menghapus elemen secara efisien tanpa perlu menggeser elemen lainnya seperti pada ArrayList. Ini membuat LinkedList menjadi pilihan yang ideal untuk implementasi struktur data seperti stack dan queue, di mana operasi keluar-masuk data dilakukan secara intensif di ujung-ujung list. Selain itu, LinkedList juga mendukung operasi traversal yang memungkinkan iterasi dari elemen pertama hingga terakhir dengan lancar.

Namun demikian, LinkedList memiliki kekurangan dalam hal akses data secara acak. Tidak seperti ArrayList yang menyediakan akses langsung ke indeks tertentu, LinkedList harus menelusuri list dari awal setiap kali ingin mengakses elemen tertentu. Ini menyebabkan waktu akses yang lebih lambat jika dibandingkan dengan ArrayList. Oleh karena itu, pemilihan antara ArrayList dan LinkedList sebaiknya disesuaikan dengan karakteristik operasi yang paling dominan dalam program.

3. Queue

Queue adalah struktur data antrian yang mengikuti prinsip **FIFO** (First In, First Out), yang berarti elemen yang pertama kali dimasukkan akan menjadi elemen pertama yang dikeluarkan. Dalam Java, struktur ini dapat diimplementasikan dengan menggunakan kelas LinkedList karena mendukung operasi add() untuk memasukkan data dan poll() atau remove() untuk mengeluarkan data. Konsep ini sangat berguna dalam banyak aplikasi dunia nyata seperti antrean pelanggan, sistem pencetakan dokumen, atau manajemen proses dalam sistem operasi.Penggunaan Queue dalam program sangat membantu dalam mengelola urutan proses secara efisien dan teratur. Misalnya, dalam kasus perpustakaan, ketika seseorang meminjam buku, maka informasi buku tersebut bisa disimpan dalam antrian Queue agar sistem dapat mengetahui buku mana yang sedang dipinjam terlebih dahulu. Ini penting agar data dapat diproses sesuai urutan dan tidak terjadi konflik atau data yang tertinggal.

Walaupun Queue sangat cocok untuk menangani proses yang terjadi secara berurutan, struktur ini memiliki keterbatasan dalam akses data. Tidak seperti List, kita tidak dapat langsung mengakses elemen di tengah antrian. Oleh karena itu, Queue lebih cocok digunakan untuk proses yang bersifat

sementara dan teratur, bukan untuk penyimpanan jangka panjang yang membutuhkan akses acak.

4. Stack

Stack adalah struktur data yang menggunakan prinsip **LIFO** (Last In, First Out), yang artinya elemen terakhir yang dimasukkan adalah elemen pertama yang akan dikeluarkan. Struktur ini dapat dianalogikan seperti tumpukan buku, di mana buku yang paling atas akan diambil terlebih dahulu. Java menyediakan kelas Stack dalam paket java.util yang memungkinkan penambahan elemen melalui metode push() dan penghapusan melalui metode pop().Struktur data Stack sangat berguna dalam situasi di mana urutan pembalikan atau pelacakan status dibutuhkan, misalnya saat proses undo dalam aplikasi, pemrosesan ekspresi matematika, atau saat mengevaluasi urutan fungsi yang dipanggil (call stack). Dalam konteks program perpustakaan ini, Stack digunakan untuk menyimpan buku yang dikembalikan, sehingga bisa diketahui buku mana yang terakhir dikembalikan.

Walaupun Stack sangat berguna dalam skenario tertentu, penggunaannya harus diperhatikan karena tidak cocok untuk semua jenis data. Akses acak terhadap elemen di tengah stack tidak dimungkinkan tanpa memodifikasi struktur, dan data hanya dapat diakses dari bagian atas. Oleh karena itu, Stack lebih cocok digunakan dalam konteks yang benar-benar membutuhkan pengelolaan data satu arah secara terbalik.

C. Metode Praktikum

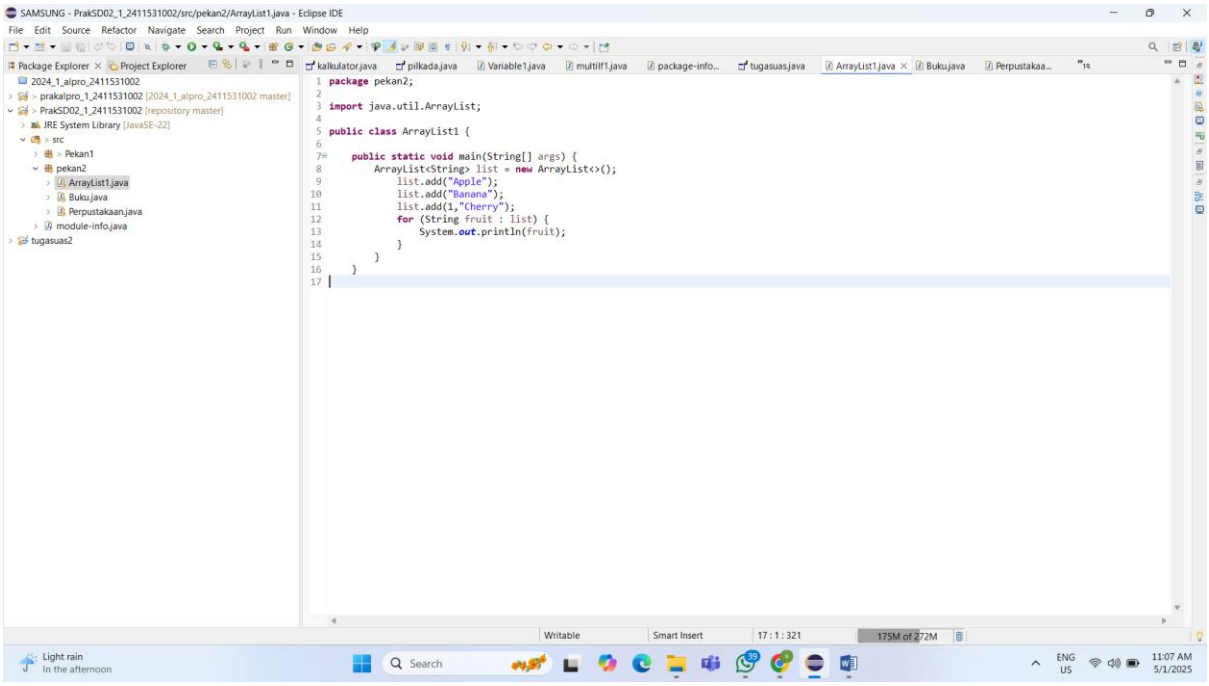
1. Kelas ArrayList1

Kelas ArrayList1 merupakan contoh penggunaan ArrayList dalam Java. Di dalam metode main(), dibuat sebuah list bernama list dengan tipe data String. Tiga buah data ditambahkan ke dalam list: "Apple", "Banana", dan "Cherry". Menariknya, "Cherry" dimasukkan pada indeks ke-1, sehingga data sebelumnya bergeser otomatis oleh ArrayList.

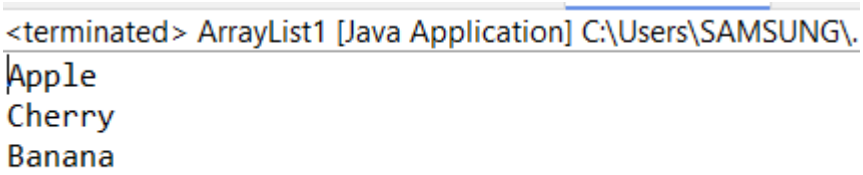
Kode ini menunjukkan bagaimana ArrayList dapat menambahkan data di posisi tertentu tanpa perlu membuat array baru secara manual. Fungsi add(index, element) memungkinkan pengembang menyisipkan data di posisi tertentu, dan list akan menyesuaikan dirinya. Selanjutnya, perulangan for-each digunakan untuk mencetak seluruh elemen dalam list secara berurutan.

Dengan kode program :

```
package pekan2;

import java.util.ArrayList;

public class ArrayList1 {

	public static void main(String[] args) {
		ArrayList<String> list = new ArrayList<>();
			list.add("Apple");
			list.add("Banana");
			list.add(1,"Cherry");
			for (String fruit : list) {
				System.out.println(fruit);
			}
		}
	}
```



Sehingga, menghasilkan output berikut :



2. Kelas Buku

Kelas Buku merupakan representasi objek buku dalam sistem perpustakaan. Kelas ini terdiri dari tiga atribut: judul, pengarang, dan isbn, yang semuanya bertipe String. Konstruktor digunakan untuk menginisialisasi ketiga atribut tersebut ketika objek Buku dibuat.

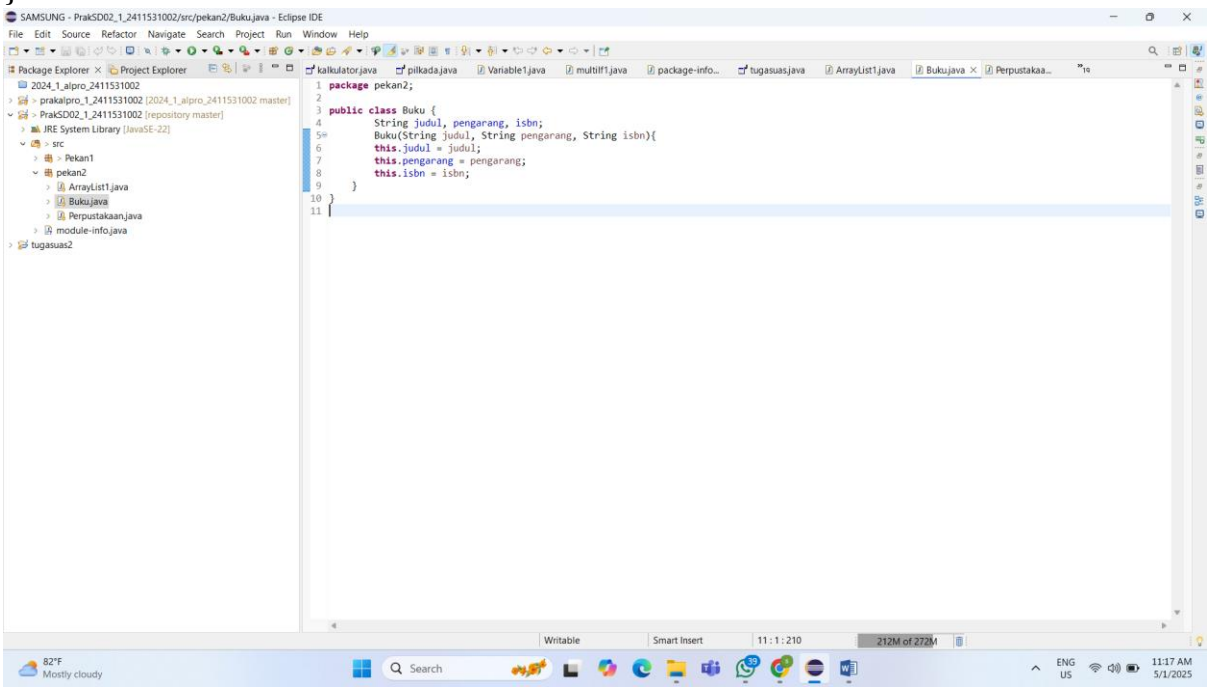
Dengan pendekatan ini, setiap buku dalam sistem dapat disimpan dalam bentuk objek terpisah yang menyimpan informasi lengkap. Pemisahan data ke dalam objek Buku membuat pengelolaan data menjadi lebih terstruktur dan sesuai dengan prinsip pemrograman berorientasi objek (OOP). Objek Buku nantinya akan digunakan sebagai elemen dalam struktur data seperti LinkedList, Queue, dan Stack.

this digunakan untuk membedakan antara variabel lokal dan variabel instance (atribut kelas). Saat objek Buku dibuat, nilai-nilai ini langsung diinisialisasi berdasarkan input pengguna. Objek-objek ini kemudian digunakan dalam struktur data lain seperti LinkedList, Queue, dan Stack.

Meskipun kelas ini tidak memiliki metode lain seperti getter atau setter, strukturnya sudah cukup untuk digunakan dalam kebutuhan dasar sistem perpustakaan sederhana. Informasi dari objek Buku bisa diakses langsung oleh kelas lain dalam paket yang sama, seperti Perpustakaan.

Dengan kode program :

```
package pekan2;

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



3. Kelas Perpustakaan

Kelas Perpustakaan merupakan inti dari program karena mengatur manajemen koleksi, peminjaman, dan pengembalian buku. Tiga struktur data digunakan: LinkedList untuk menyimpan semua koleksi buku (koleksiBuku), Queue untuk antrian peminjaman (Peminjaman), dan Stack untuk pencatatan pengembalian buku (Pengembalian).

Metode tambahBuku() digunakan untuk menambahkan buku ke dalam koleksi perpustakaan. Saat pengguna memilih opsi ini, program akan meminta

input judul, pengarang, dan ISBN, lalu menyimpannya ke dalam koleksiBuku. Metode pinjamBuku() akan mencari buku berdasarkan judul di koleksiBuku, lalu menambahkan objek buku ke dalam antrian Peminjaman. Sedangkan kembalikanBuku() akan mencari judul di antrian pinjaman, lalu menambahkan ke Pengembalian dengan sistem tumpukan.

Pada metode main(), antarmuka berbasis teks dibuat untuk memungkinkan pengguna memilih menu: tambah, pinjam, kembalikan, atau keluar. Input dari pengguna akan diterima melalui objek Scanner. Program berjalan terus hingga opsi keluar (angka 4) dipilih. Output dari setiap aksi tidak eksplisit dicetak (misalnya, tidak ada pesan "Buku berhasil dipinjam"), namun struktur data secara internal akan berubah sesuai aksi.

Dengan kode program :

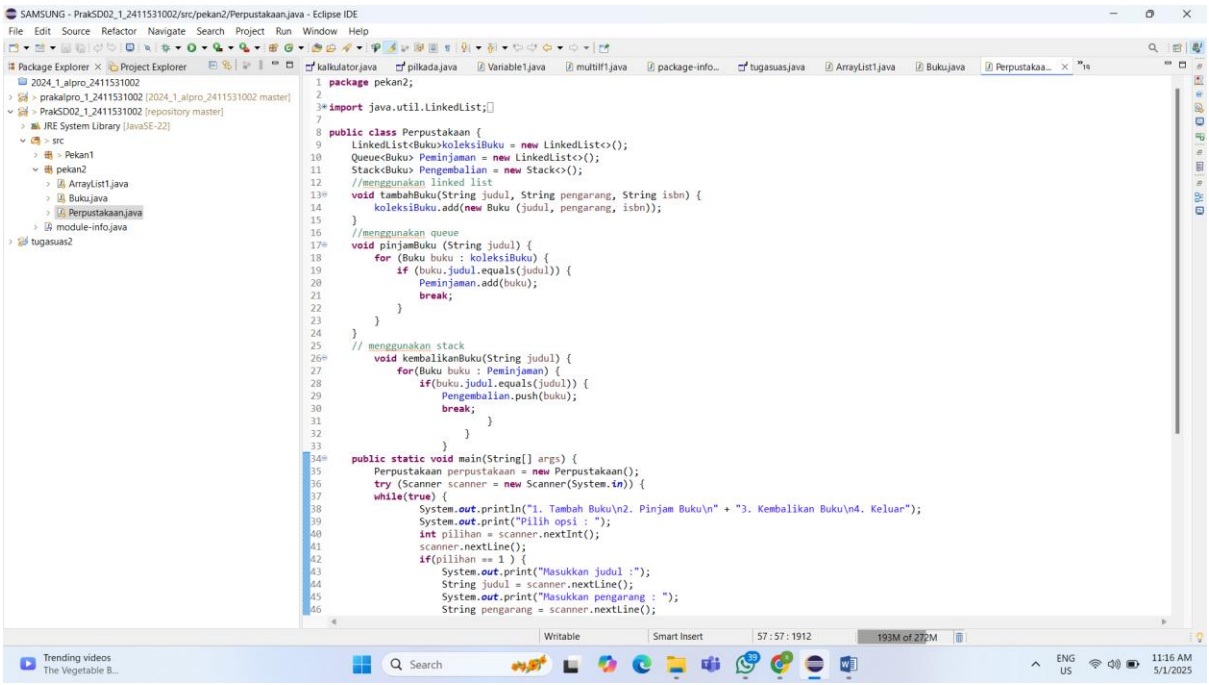
```
package pekan2;



import java.util.LinkedList;

import java.util.Queue;

import java.util.Stack;

import java.util.Scanner;



public class Perpustakaan {

	LinkedList<Buku>koleksiBuku = new LinkedList<>();

	Queue<Buku> Peminjaman = new LinkedList<>();

	Stack<Buku> Pengembalian = new Stack<>();

	//menggunakan linked list

	void tambahBuku(String judul, String pengarang, String isbn) {

		koleksiBuku.add(new Buku (judul, pengarang, isbn));

	}

	//menggunakan queue
```

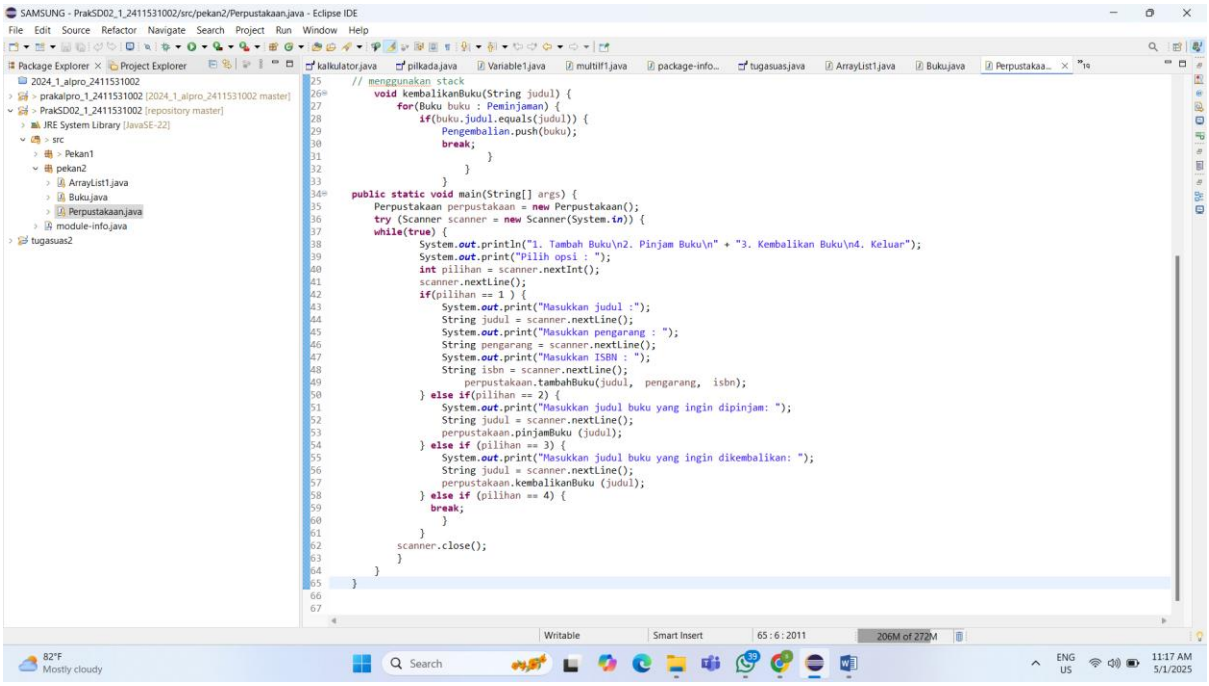
```
    void pinjamBuku (String judul) {
        for (Buku buku : koleksiBuku) {
            if (buku.judul.equals(judul)) {
                Peminjaman.add(buku);
                break;
            }
        }
    }
    // menggunakan stack
        void kembalikanBuku(String judul) {
            for(Buku buku : Peminjaman) {
                if(buku.judul.equals(judul)) {
                    Pengembalian.push(buku);
                    break;
                        }
                    }
                    }
    public static void main(String[] args) {
        Perpustakaan perpustakaan = new Perpustakaan();
        try (Scanner scanner = new Scanner(System.in)) {
        while(true) {
                    System.out.println("1. Tambah Buku\n2. Pinjam
Buku\n" + "3. Kembalikan Buku\n4. Keluar");
                    System.out.print("Pilih opsi : ");
                    int pilihan = scanner.nextInt();
```

```
                    scanner.nextLine();
                    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.tambahBuku(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();
                }
```

}

}


SAMSUNG - PrakSD02_1_2411531002/src/pekan2/Perpustakaan.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Project Explorer
2024_1_alpro_2411531002
prakalpro_1_2411531002 [2024_1_alpro_2411531002 master]
PrakSD02_1_2411531002 [repository master]
JRE System Library [JavaSE-22]
src
Pekan1
pekan2
ArrayList1.java
Buku.java
Perpustakaan.java
module-info.java
tugasuas2
kalkulator.java
pilkada.java
Variable1.java
multilf1.java
package-info...
tugasuas.java
ArrayList1.java
Buku.java
Perpustakaa...
1 package pekan2;
2
3 import java.util.LinkedList;
7
8 public class Perpustakaan {
9 LinkedList<Buku>koleksiBuku = new LinkedList<>();
10 Queue<Buku> Peminjaman = new LinkedList<>();
11 Stack<Buku> Pengembalian = new Stack<>();
12 //menggunakan linked list
13 void tambahBuku(String judul, String pengarang, String isbn) {
14 koleksiBuku.add(new Buku (judul, pengarang, isbn));
15 }
16 //menggunakan queue
17 void pinjamBuku (String judul) {
18 for (Buku buku : koleksiBuku) {
19 if (buku.judul.equals(judul)) {
20 Peminjaman.add(buku);
21 break;
22 }
23 }
24 }
25 // menggunakan stack
26 void kembalikanBuku(String judul) {
27 for(Buku buku : Peminjaman) {
28 if(buku.judul.equals(judul)) {
29 Pengembalian.push(buku);
30 break;
31 }
32 }
33 }
34 public static void main(String[] args) {
35 Perpustakaan perpustakaan = new Perpustakaan();
36 try (Scanner scanner = new Scanner(System.in)) {
37 while(true) {
38 System.out.println("1. Tambah Buku\n2. Pinjam Buku\n" + "3. Kembalikan Buku\n4. Keluar");
39 System.out.print("Pilih opsi : ");
40 int pilihan = scanner.nextInt();
41 scanner.nextLine();
42 if(pilihan == 1) {
43 System.out.print("Masukkan judul :");
44 String judul = scanner.nextLine();
45 System.out.print("Masukkan pengarang : ");
46 String pengarang = scanner.nextLine();
Writable
Smart Insert
57 : 57 : 1912
193M of 272M
Trending videos
The Vegetable B...
Search
ENG
US
11:16 AM
5/1/2025


SAMSUNG - PrakSD02_1_2411531002/src/pekan2/Perpustakaan.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Project Explorer
2024_1_alpro_2411531002
prakalpro_1_2411531002 [2024_1_alpro_2411531002 master]
PrakSD02_1_2411531002 [repository master]
JRE System Library [JavaSE-22]
src
Pekan1
pekan2
ArrayList1.java
Buku.java
Perpustakaan.java
module-info.java
tugasuas2
kalkulator.java
pilkada.java
Variable1.java
multilf1.java
package-info...
tugasuas.java
ArrayList1.java
Buku.java
Perpustakaa...
25 // menggunakan stack
26 void kembalikanBuku(String judul) {
27 for(Buku buku : Peminjaman) {
28 if(buku.judul.equals(judul)) {
29 Pengembalian.push(buku);
30 break;
31 }
32 }
33 }
34 public static void main(String[] args) {
35 Perpustakaan perpustakaan = new Perpustakaan();
36 try (Scanner scanner = new Scanner(System.in)) {
37 while(true) {
38 System.out.println("1. Tambah Buku\n2. Pinjam Buku\n" + "3. Kembalikan Buku\n4. Keluar");
39 System.out.print("Pilih opsi : ");
40 int pilihan = scanner.nextInt();
41 scanner.nextLine();
42 if(pilihan == 1) {
43 System.out.print("Masukkan judul :");
44 String judul = scanner.nextLine();
45 System.out.print("Masukkan pengarang : ");
46 String pengarang = scanner.nextLine();
47 System.out.print("Masukkan ISBN : ");
48 String isbn = scanner.nextLine();
49 perpustakaan.tambahBuku(judul, pengarang, isbn);
50 } else if(pilihan == 2) {
51 System.out.print("Masukkan judul buku yang ingin dipinjam: ");
52 String judul = scanner.nextLine();
53 perpustakaan.pinjamBuku (judul);
54 } else if (pilihan == 3) {
55 System.out.print("Masukkan judul buku yang ingin dikembalikan: ");
56 String judul = scanner.nextLine();
57 perpustakaan.kembalikanBuku (judul);
58 } else if (pilihan == 4) {
59 break;
60 }
61 }
62 scanner.close();
63 }
64 }
65 }
66
67
Writable
Smart Insert
65 : 6 : 2011
206M of 272M
82°F
Mostly cloudy
Search
ENG
US
11:17 AM
5/1/2025

Sehingga, ouputnya adalah :

```
Perpustakaan [Java Application] C:\Users\SAMSUNG\.p2\pool\pl
1. Tambah Buku
2. Pinjam Buku
3. Kembalikan Buku
4. Keluar
Pilih opsi : 1
Masukkan judul :Langit Tak Pernah Ingkar Janji
Masukkan pengarang : Arsyila Putri
Masukkan ISBN : 978-623-91234-01-7

1. Tambah Buku
2. Pinjam Buku
3. Kembalikan Buku
4. Keluar
Pilih opsi : 2
Masukkan judul buku yang ingin dipinjam: Langit Tak Pernah Ingkar Janji
Masukkan judul buku
1. Tambah Buku
2. Pinjam Buku
3. Kembalikan Buku
4. Keluar
Pilih opsi : 4

1. Tambah Buku
2. Pinjam Buku
3. Kembalikan Buku
4. Keluar
Pilih opsi : 3
Masukkan judul buku yang ingin dikembalikan: Langit Tak Pernah Ingkar Janji
```

D. Kesimpulan Praktikum

Dari praktikum ini dapat disimpulkan bahwa struktur data dinamis seperti ArrayList, LinkedList, Queue, dan Stack sangat membantu dalam pengelolaan data yang fleksibel dan efisien. Masing-masing struktur memiliki karakteristik dan keunggulan tersendiri tergantung dari kebutuhan pemrosesan data yang dihadapi. Misalnya, ArrayList unggul dalam akses indeks langsung, sementara LinkedList lebih unggul untuk operasi penambahan dan penghapusan elemen di tengah.

Dalam implementasi sistem perpustakaan, pemanfaatan tiga struktur data (LinkedList untuk menyimpan koleksi buku, Queue untuk peminjaman, dan Stack untuk pengembalian) menggambarkan bagaimana struktur-struktur tersebut bekerja di skenario nyata. Hal ini menunjukkan bahwa pemilihan struktur data yang tepat sangat mempengaruhi efisiensi dan alur program. Selain itu, pendekatan pemrograman berorientasi objek juga membantu menyusun program menjadi lebih terstruktur dan modular.

Secara keseluruhan, praktikum ini memberikan pemahaman yang kuat tentang cara kerja dan penerapan berbagai jenis struktur data dalam membangun aplikasi sederhana. Dengan mengenal karakteristik masing-masing struktur, proses analisis dan pemrograman menjadi lebih terarah dan logis dalam menyelesaikan permasalahan di dunia nyata.