**Exercise 6: Library Management System**

**Scenario:**

You are developing a library management system where users can search for books by title or author.

**Steps:**

1. **Understand Search Algorithms:**
   * **Explain linear search and binary search algorithms.**

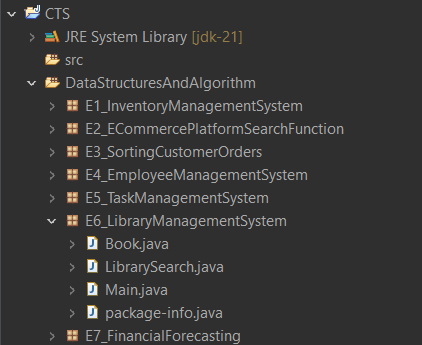
**Linear Search**

* Checks each element one by one.
* Works on unsorted data.
* Time Complexity: O(n)

**Binary Search**

* Works on sorted data.
* Repeatedly splits the list in half.
* Time Complexity: O(log n)

1. **Setup:**
   * **Create a class Book with attributes like bookId, title, and author.**

****

1. **Implementation:**
   * **Implement linear search to find books by title.**
   * **Implement binary search to find books by title (assuming the list is sorted).**

**Books.java**

package E6\_LibraryManagementSystem;

public class Book {

int bookId;

String title;

String author;

public Book(int id, String title, String author) {

this.bookId = id;

this.title = title;

this.author = author;

}

public String toString() {

return bookId + " - " + title + " by " + author;

}

}

**LibrarySearch.java**

package E6\_LibraryManagementSystem;

public class LibrarySearch {

// Linear Search by title

public static Book linearSearch(Book[] books, String title) {

for (Book b : books) {

if (b.title.equalsIgnoreCase(title)) {

return b;

}

}

return null;

}

// Binary Search by title (assumes sorted)

public static Book binarySearch(Book[] books, String title) {

int low = 0, high = books.length - 1;

while (low <= high) {

int mid = (low + high) / 2;

int cmp = title.compareToIgnoreCase(books[mid].title);

if (cmp == 0)

return books[mid];

else if (cmp > 0)

low = mid + 1;

else

high = mid - 1;

}

return null;

}

}

**Main.java**

package E6\_LibraryManagementSystem;

import java.util.\*;

public class Main {

public static void main(String[] args) {

Book[] books = {

new Book(101, "Java Programming", "James Gosling"),

new Book(102, "Python Basics", "Guido van Rossum"),

new Book(103, "Data Structures", "Robert Lafore"),

new Book(104, "Algorithms", "CLRS")

};

// Sort the array for binary search

Arrays.*sort*(books, Comparator.*comparing*(b -> b.title));

// Test linear search

Book found1 = LibrarySearch.*linearSearch*(books, "Algorithms");

System.*out*.println("Linear Search: " + (found1 != null ? found1 : "Not found"));

// Test binary search

Book found2 = LibrarySearch.*binarySearch*(books, "Algorithms");

System.*out*.println("Binary Search: " + (found2 != null ? found2 : "Not found"));

// Try not found case

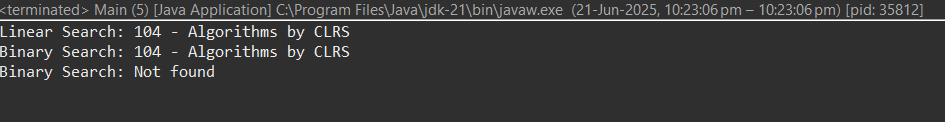
Book notFound = LibrarySearch.*binarySearch*(books, "Machine Learning");

System.*out*.println("Binary Search: " + (notFound != null ? notFound : "Not found"));

}

}

**Output:**

****

1. **Analysis:**
   * **Compare the time complexity of linear and binary search.**

|  |  |  |
| --- | --- | --- |
| **Algorithm** | **Time Complexity** | **Suitable For** |
| **Linear Search** | O(n) | Unsorted, small data |
| **Binary Search** | O(log n) | Sorted, large data |

* + **Discuss when to use each algorithm based on the data set size and order.**

Use **linear search** when:

* The data is **unsorted**
* You’re working with **few records**

Use **binary search** when:

* The data is **already sorted**
* You need **fast search** on **large data**