**Exercise 6: Library Management System**

**Steps:**

1. **Understand Search Algorithms:**

* Explain linear search and binary search algorithms.

Ans. **Binary Search Algorithm:**

Below is the step-by-step algorithm for Binary Search:

* Divide the search space into two halves by [**finding the middle index “mid”**](https://www.geeksforgeeks.org/problem-binary-search-implementations/).
* Compare the middle element of the search space with the **key**.
* If the **key**is found at middle element, the process is terminated.
* If the **key**is not found at middle element, choose which half will be used as the next search space.
  + If the **key**is smaller than the middle element, then the **left**side is used for next search.
  + If the **key**is larger than the middle element, then the **right**side is used for next search.
* This process is continued until the **key**is found or the total search space is exhausted.

**Algorithm for Linear Search Algorithm:**

The algorithm for linear search can be broken down into the following steps:

* **Start:** Begin at the first element of the collection of elements.
* **Compare:** Compare the current element with the desired element.
* **Found:** If the current element is equal to the desired element, return true or index to the current element.
* **Move:**Otherwise, move to the next element in the collection.
* **Repeat:** Repeat steps 2-4 until we have reached the end of collection.
* **Not found:** If the end of the collection is reached without finding the desired element, return that the desired element is not in the array.

1. **Setup:**
   * Create a class **Book** with attributes like **bookId**, **title**, and **author**.
2. **Implementation:**
   * Implement linear search to find books by title.
   * Implement binary search to find books by title (assuming the list is sorted).

Code:

package Algorithm\_DataStructures.Exercise6.Library\_Management\_System;

class Book {

    int bookId;

    String title;

    String author;

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

        this.bookId = bookId;

        this.title = title;

        this.author = author;

    }

}

class BookSearch {

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

        for (Book book : books) {

            if (book.title.equals(title)) {

                return book;

            }

        }

        return null;

    }

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

        int low = 0;

        int high = books.length - 1;

        while (low <= high) {

            int mid = low + (high - low) / 2;

            int compare = title.compareTo(books[mid].title);

            if (compare == 0) {

                return books[mid];

            } else if (compare < 0) {

                high = mid - 1;

            } else {

                low = mid + 1;

            }

        }

        return null;

    }

}

public class LMS {

    public static void main(String[] args) {

        Book[] books = {

            new Book(1, "Java Programming", "John Smith"),

            new Book(2, "Algorithms Unlocked", "Thomas Cormen"),

            new Book(3, "Clean Code", "Robert C. Martin")

        };

        String searchTitle = "Clean Code";

        // Linear Search

        Book linearResult = BookSearch.linearSearch(books, searchTitle);

        if (linearResult != null) {

            System.out.println("Book found using linear search: " + linearResult.title + " by " + linearResult.author);

        } else {

            System.out.println("Book not found using linear search.");

        }

        // Binary Search (Assuming books array is sorted by title)

        // Sorting the array for binary search

        java.util.Arrays.sort(books, (book1, book2) -> book1.title.compareTo(book2.title));

        Book binaryResult = BookSearch.binarySearch(books, searchTitle);

        if (binaryResult != null) {

            System.out.println("Book found using binary search: " + binaryResult.title + " by " + binaryResult.author);

        } else {

            System.out.println("Book not found using binary search.");

        }

    }

}

1. **Analysis:**

* *Compare the time complexity of linear and binary search*.

Ans. The time complexity of binary search is O(log n), while the time complexity of linear search is O(n)

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

Ans. Linear Search can be used when –

* **Data Set Size**: Linear search is suitable for small data sets or unsorted data.
* **Order**: It works on both sorted and unsorted arrays.

Binary Search can be used when –

* **Data Set Size**: Binary search is efficient for large, sorted data sets.
* **Order**: Requires the data to be sorted in ascending or descending order.