**Exercise 6: Library Management System**

**Library Management System.java**

**package** mypackage;

**import** java.util.\*;

**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;

}

}

**public** **class** LibraryManagementSystem {

**static** Scanner *sc* = **new** Scanner(System.***in***);

**static** ArrayList<Book> *books* = **new** ArrayList<>();

**public** **static** **void** main(String[] args) {

**while** (**true**) {

System.***out***.println("\nLibrary Management System");

System.***out***.println("1. Add Book");

System.***out***.println("2. Linear Search by Title");

System.***out***.println("3. Binary Search by Title");

System.***out***.println("4. Display All Books");

System.***out***.println("5. Exit");

System.***out***.print("Enter your choice: ");

**int** choice = *sc*.nextInt();

*sc*.nextLine(); // consume newline

**switch** (choice) {

**case** 1:

*addBook*();

**break**;

**case** 2:

*linearSearch*();

**break**;

**case** 3:

*binarySearch*();

**break**;

**case** 4:

*displayBooks*();

**break**;

**case** 5:

**return**;

**default**:

System.***out***.println("Invalid choice.");

}

}

}

**public** **static** **void** addBook() {

System.***out***.print("Enter Book ID: ");

**int** id = *sc*.nextInt();

*sc*.nextLine();

System.***out***.print("Enter Book Title: ");

String title = *sc*.nextLine();

System.***out***.print("Enter Author Name: ");

String author = *sc*.nextLine();

*books*.add(**new** Book(id, title, author));

System.***out***.println("Book added successfully.");

}

**public** **static** **void** linearSearch() {

**if** (*books*.isEmpty()) {

System.***out***.println("No books available.");

**return**;

}

System.***out***.print("Enter title to search: ");

String searchTitle = *sc*.nextLine();

**boolean** found = **false**;

**for** (Book b : *books*) {

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

System.***out***.println("Book found: " + b.bookId + ", " + b.title + ", " + b.author);

found = **true**;

}

}

**if** (!found) {

System.***out***.println("Book not found.");

}

}

**public** **static** **void** binarySearch() {

**if** (*books*.isEmpty()) {

System.***out***.println("No books available.");

**return**;

}

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

System.***out***.print("Enter title to search: ");

String searchTitle = *sc*.nextLine();

**int** left = 0;

**int** right = *books*.size() - 1;

**boolean** found = **false**;

**while** (left <= right) {

**int** mid = (left + right) / 2;

String midTitle = *books*.get(mid).title.toLowerCase();

**int** cmp = searchTitle.toLowerCase().compareTo(midTitle);

**if** (cmp == 0) {

System.***out***.println("Book found: " + *books*.get(mid).bookId + ", " + *books*.get(mid).title + ", " + *books*.get(mid).author);

found = **true**;

**break**;

} **else** **if** (cmp < 0) {

right = mid - 1;

} **else** {

left = mid + 1;

}

}

**if** (!found) {

System.***out***.println("Book not found.");

}

}

**public** **static** **void** displayBooks() {

**if** (*books*.isEmpty()) {

System.***out***.println("No books available.");

**return**;

}

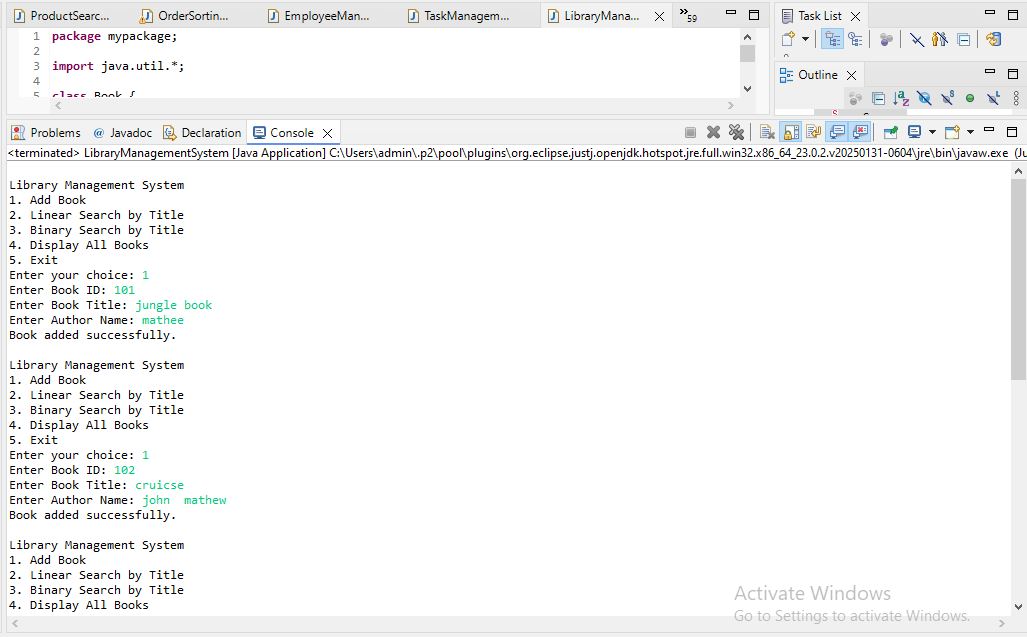
**for** (Book b : *books*) {

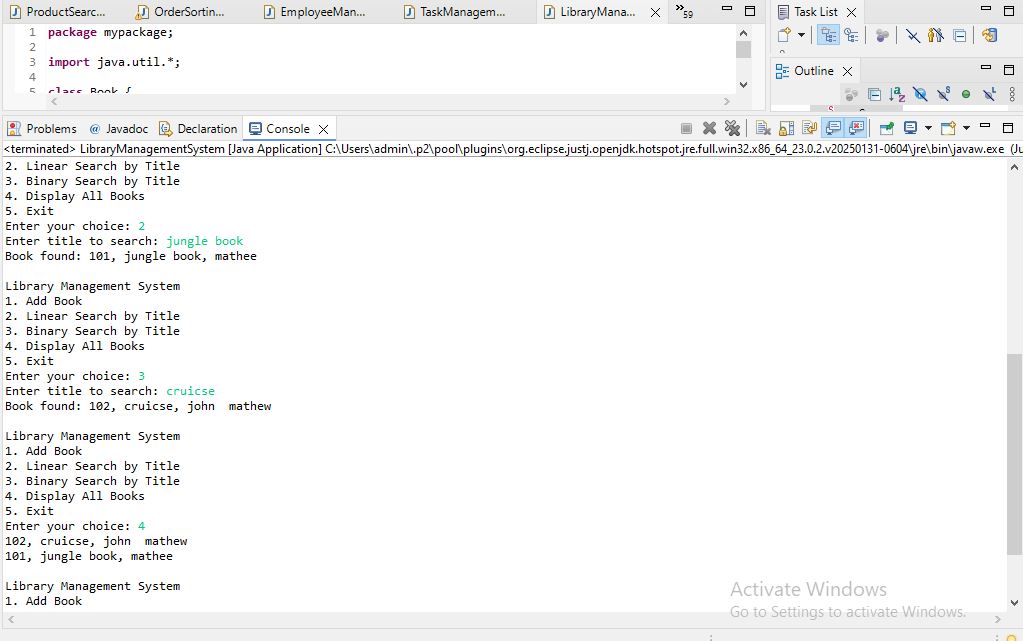
System.***out***.println(b.bookId + ", " + b.title + ", " + b.author);

}

}

}





**Linear Search:**

Linear search is a simple search algorithm that sequentially checks each element of the list until a match is found or the whole list has been searched. It does not require the list to be sorted.

**Binary Search:**

Binary search is a more efficient search algorithm that works only on sorted lists. It repeatedly divides the search interval in half. If the search key is less than the middle element, it continues the search in the left half, otherwise in the right half.

**Setup:**

We create a class Book with the following attributes:

* bookId (integer)
* title (string)
* author (string)

The system allows adding books to a list. The list can be searched using both linear and binary search.

1. Implementation:

**Linear Search:**

* Iterate through each book in the list.
* Compare the title of each book with the search query.
* If a match is found, return the book details.
* If no match is found after traversing the entire list, return "Book not found."

**Binary Search:**

* First, sort the list of books based on title.
* Set left and right pointers at the start and end of the list.
* Calculate the mid index.
* Compare the mid element's title with the search query.
* If it matches, return the book.
* If the search query is smaller, search in the left half.
* If the search query is larger, search in the right half.
* Repeat until the book is found or the search space is empty.

1. Analysis:

**Time Complexity:**

**Linear Search:**

* Best case: O(1) (first element matches)
* Average case: O(n)
* Worst case: O(n)

**Binary Search:**

* Best case: O(1) (middle element matches)
* Average case: O(log n)
* Worst case: O(log n)

**Comparison:**

Linear search works on both sorted and unsorted data but is inefficient for large datasets. Binary search is much faster but requires the data to be sorted beforehand.

When to use:

* Use linear search when the dataset is small or unsorted.
* Use binary search when the dataset is large and sorted.