## **Exercise 6:Library Management System**

## **Linear Search**

**Definition:**

Linear search is a simple searching algorithm where we **check each element one by one** in a list until the desired value is found or the list ends.

**How it works:**

* Start from the first element.
* Compare it with the target value.
* If matched, return its position.
* If not, move to the next element.
* Repeat until the value is found or the list ends.

**Time Complexity:**

* **Best case:** O(1) (if the element is at the start)
* **Worst case:** O(n) (if at the end or not present)

## **Binary Search**

**Definition:**

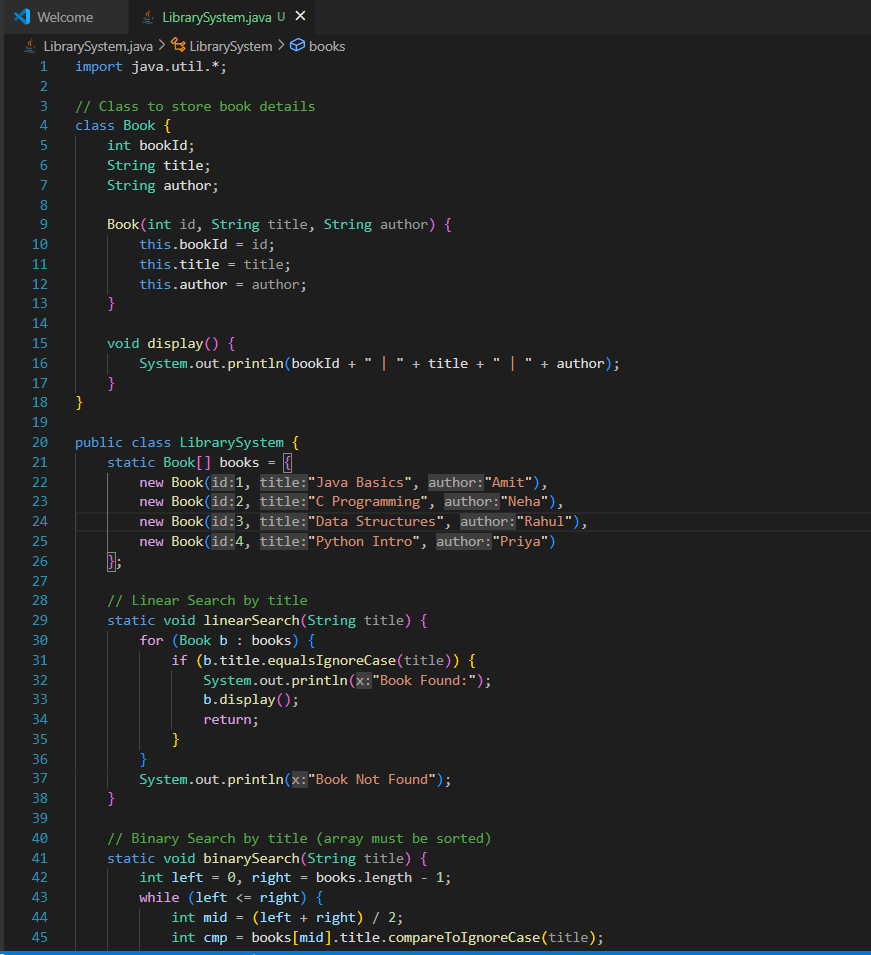
Binary search is an efficient searching algorithm that works on **sorted data only** by repeatedly dividing the search range in half.

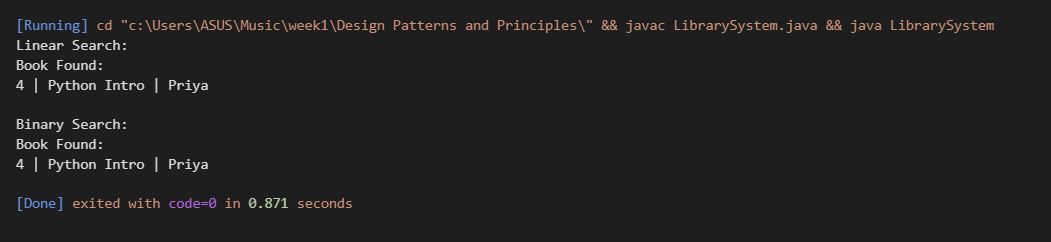
**How it works:**

* Start with the middle element.
* If it matches the target, search ends.
* If the target is less, search the left half.
* If greater, search the right half.
* Repeat until found or no elements are left.

**Time Complexity:**

* **Best case:** O(1)
* **Worst case:** O(log n)





## **Comparison of Time Complexity**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm** | **Best Case** | **Average Case** | **Worst Case** | **When to Use** |
| **Linear Search** | O(1) | O(n) | O(n) | Small or unsorted data |
| **Binary Search** | O(1) | O(log n) | O(log n) | Large and sorted data only |

## **Explanation:**

* **Linear Search**:
  + Checks one by one.
  + Takes longer for big lists.
  + Works on both sorted and unsorted data.
* **Binary Search**:
  + Divides the search space in half each time.
  + Much faster for large, sorted data.
  + Won’t work on unsorted lists.

## **When to Use Which Search Algorithm**

**Linear Search:**

* Use for **small or unsorted data**.
* Simple to implement.
* No need for sorting.

**Binary Search:**

* Use for **large, sorted data**.
* Much faster than linear search.
* Requires sorted list.

*.*