**Exercise 6: Library Management System**

**Step 1: Understand Search Algorithms**

**Linear Search**:

* **Algorithm**: Linear search involves checking each element in the list sequentially until the desired element is found or the end of the list is reached.
* **Time Complexity**: O(n), where n is the number of elements in the list. The worst-case scenario is that the element is at the end of the list or not present at all.
* **Use Case**: Useful when the list is unsorted or small.

**Binary Search**:

* **Algorithm**: Binary search involves dividing the sorted list in half, comparing the middle element to the target, and recursively searching the left or right half based on the comparison.
* **Time Complexity**: O(log n), where n is the number of elements in the list. The list must be sorted for binary search to work.
* **Use Case**: More efficient than linear search for large, sorted datasets

**Step 4: Analysis**

**Time Complexity**:

* **Linear Search**: O(n), where n is the number of books. The search time increases linearly with the number of books.
* **Binary Search**: O(log n), where n is the number of books. The search time increases logarithmically, making it much faster for large datasets.

**When to Use Each Algorithm**:

* **Linear Search**: Use linear search when the dataset is small or unsorted. It is simple and does not require preprocessing (like sorting).
* **Binary Search**: Use binary search when the dataset is large and sorted. Although it requires an initial sorting step (O(n log n) time complexity for sorting), the subsequent searches are much faster (O(log n)).