**Library Management System**

**1.Understand Search Algorithms**

**Linear Search:**

* **Definition:** Linear search scans each element in the list sequentially until the target element is found or the end of the list is reached.
* **Time Complexity:**
  + **Best Case:** O(1) – If the element is found at the first position.
  + **Average Case:** O(n) – On average, half of the elements are checked.
  + **Worst Case:** O(n) – If the element is at the end or not present.

**Binary Search:**

* **Definition:** Binary search works on a sorted list by repeatedly dividing the search interval in half. It compares the target value to the middle element and narrows down the search range.
* **Time Complexity:**
  + **Best Case:** O (1) – If the element is at the middle position.
  + **Average Case:** O (log n) – The search space is reduced exponentially.
  + **Worst Case:** O (log n) – Continues halving until the target is found or the search space is empty.
* **Precondition:** The list must be sorted before applying binary search.

**4. Analysis**

**Time Complexity Comparison:**

1. **Linear Search:**
   * **Best Case:** O (1) – Element found immediately.
   * **Average Case:** O (n) – Average time to find the element.
   * **Worst Case:** O (n) – Element is at the end or not found.
2. **Binary Search:**
   * **Best Case:** O (1) – Element is at the middle.
   * **Average Case:** O(log n) – Reduces the search space logarithmically.
   * **Worst Case:** O(log n) – Continues halving until the search space is exhausted.

**When to Use Each Algorithm:**

* **Linear Search:**
  + Use when the dataset is small or unsorted.
  + Simple to implement and understand.
* **Binary Search:**
  + Use when the dataset is large and sorted.
  + More efficient for large datasets due to logarithmic time complexity.
  + Requires preprocessing (sorting) if the dataset isn’t already sorted.