**Exercise 6: Library Management System**

1. **Understand Search Algorithms:**

**Linear Search Algorithm**

**Definition:** Linear search, also known as sequential search, is a simple searching algorithm that checks each element in a list one by one until the desired element is found or the list ends.

**Algorithm Steps:**

1. Start from the first element in the list.
2. Compare the current element with the target value.
3. If the current element matches the target value, return the index of the element.
4. If the current element does not match, move to the next element.
5. Repeat steps 2-4 until the target value is found or the end of the list is reached.

**Time Complexity:**

* Best case: O(1) (the target value is the first element).
* Average case: O(n/2) = O(n) (the target value is somewhere in the middle).
* Worst case: O(n) (the target value is the last element or not present).

**Space Complexity:**

* O(1) (no extra space is required).

**Binary Search Algorithm**

**Definition:** Binary search is an efficient algorithm for finding an item from a sorted list of items. It works by repeatedly dividing the search interval in half.

**Algorithm Steps:**

1. Start with the entire list.
2. Find the middle element of the list.
3. If the middle element matches the target value, return the index of the middle element.
4. If the middle element is greater than the target value, repeat the search on the left half of the list.
5. If the middle element is less than the target value, repeat the search on the right half of the list.
6. Repeat steps 2-5 until the target value is found or the search interval is empty.

**Time Complexity:**

* Best case: O(1) (the target value is the middle element on the first check).
* Average case: O(log n).
* Worst case: O(log n).

**Space Complexity:**

* O(1) for the iterative version.
* O(log n) for the recursive version (due to the call stack).

**Analysis**

**Time Complexity Comparison:**

* **Linear Search:**
  + **Best Case:** O(1)
  + **Average Case:** O(n)
  + **Worst Case:** O(n)
* **Binary Search:**
  + **Best Case:** O(1)
  + **Average Case:** O(log n)
  + **Worst Case:** O(log n)

**When to Use Each Algorithm:**

* **Linear Search:**
  + Use when the list is small or unsorted.
  + Useful when sorting the list is not feasible or too costly.
  + Suitable for a one-time search operation where the overhead of sorting is not justified.
* **Binary Search:**
  + Use when the list is large and sorted.
  + More efficient for repeated search operations as the list is already sorted.
  + Ideal for scenarios where search speed is crucial and the list can be maintained in sorted order.

In conclusion, linear search is straightforward and useful for small or unsorted datasets, while binary search is highly efficient for large, sorted datasets. The choice of search algorithm depends on the specific requirements and constraints of the application.