Library Management System

**Understanding Search Algorithms**

**Linear Search**

* **Description**: Linear search, also known as sequential search, is the simplest search algorithm. It works by checking every element in the list sequentially until the desired element is found or the list ends.
* **Algorithm**:
  1. Start from the first element of the list.
  2. Compare the current element with the target element.
  3. If the current element matches the target, return the index of the element.
  4. If the current element does not match the target, move to the next element.
  5. Repeat steps 2-4 until the end of the list is reached.
* **Time Complexity**: O(n) where n is the number of elements in the list.

**Binary Search**

* **Description**: Binary search is a more efficient algorithm but requires the list to be sorted. It works by repeatedly dividing the search interval in half.
* **Algorithm**:
  1. Start with two pointers, one at the beginning (low) and one at the end (high) of the list.
  2. Find the middle element.
  3. If the middle element is equal to the target, return the index of the middle element.
  4. If the target is less than the middle element, repeat the search on the left half.
  5. If the target is greater than the middle element, repeat the search on the right half.
  6. Repeat steps 2-5 until the target is found or the interval is empty.
* **Time Complexity**: O(log n) where n is the number of elements in the list.

**Time Complexity Comparison**

* **Linear Search**: O(n)
* **Binary Search**: O(log n)

**When to Use Each Algorithm**

* **Linear Search**:
  + Use when the list is small or unsorted.
  + Simple to implement and doesn't require preprocessing (sorting).
  + Suitable for a one-time or infrequent search in an unsorted list.
* **Binary Search**:
  + Use when the list is large and sorted.
  + Much faster for large datasets due to its logarithmic time complexity.
  + Requires the list to be sorted, which might add overhead if the list is frequently updated.