**Exercise 6: Library Management System**

**1. Understand Search Algorithms**

* **Linear Search**: This algorithm checks each item in the list sequentially until the target is found or the list ends. It works on both sorted and unsorted lists, but can be slow for large datasets due to its O(n) time complexity.
* **Binary Search**: This algorithm divides the list into halves to locate the target, assuming the list is sorted. It repeatedly narrows down the search range, resulting in O(log n) time complexity. It’s much faster than linear search for large, sorted datasets.

**2. Setup**

* **Book Class**: A class Book represents each book with attributes like bookId, title, and author. This structure is used to manage and access book information efficiently within the library system.

**3. Implementation**

* **Linear Search**: Iteratively scans through the list of books to find one matching the title. Suitable for small or unsorted datasets but less efficient for larger collections.
* **Binary Search**: Searches a sorted list of books by repeatedly dividing the search interval in half. It is efficient for large datasets but requires that the list be sorted.

**4. Analysis**

* **Time Complexity Comparison**:
  + **Linear Search**: O(n) - Performance degrades linearly with the size of the list.
  + **Binary Search**: O(log n) - Much faster for large datasets but requires sorted data.
* **When to Use Each Algorithm**:
  + **Linear Search**: Use when the list is small, unsorted, or frequently updated. It’s straightforward and doesn’t require sorting.
  + **Binary Search**: Use when dealing with large, sorted lists where efficiency is crucial. Ensure the data is sorted beforehand to leverage its faster search capabilities.