**Library Management System - Analysis**

1. **Understanding Search Algorithms :**
2. **Linear Search:** Linear search, or sequential search, involves checking each element in a list one by one until the target element is found or the list is exhausted.

* **Time Complexity**:
  + **Best Case**: O(1) (when the target is the first element)
  + **Average Case**: O(n) (when the target is in the middle)
  + **Worst Case**: O(n) (when the target is the last element or not present)

1. **Binary Search:** Binary search works on sorted lists by repeatedly dividing the search interval in half. It compares the target with the middle element of the interval and adjusts the interval based on the comparison.

* **Time Complexity**:
  + **Best Case**: O(1) (when the target is the middle element)
  + **Average Case**: O(log n)
  + **Worst Case**: O(log n) (when the search interval reduces to a single element)

1. **Analysis :**

* **Comparison of Time Complexity:**
* **Linear Search**:
  + Time complexity is O(n), making it less efficient for large data sets because it requires checking each element individually.
* **Binary Search**:
  + Time complexity is O(log n), making it more efficient for large, sorted data sets due to its ability to eliminate half of the remaining elements in each step.

1. **When to Use Each Algorithm:**

* **Linear Search**: Best used for small or unsorted data sets where sorting is not feasible or necessary. It’s also useful when the data is stored in a structure where binary search is not applicable.
* **Binary Search**: Ideal for large, sorted data sets where the overhead of sorting is outweighed by the faster search time. It’s crucial to ensure that the data is sorted before applying binary search.