**Exercise 6: Library Management System**

**Explain linear search and binary search algorithms.**

**Linear Search**

Linear search is a basic algorithm that scans each element in a list one by one until it either finds the target or reaches the end.

**Procedure:**

1. Begin at the first element.
2. Compare each element with the target value.
3. If a match is found, return the index or element.
4. If not, move to the next item.
5. Continue this process through the entire list.
6. If the target isn’t found by the end, return null or indicate it’s not present.

**Efficiency:**

* **Time Complexity:** O(n), since it might have to examine all n elements in the worst case.

**Binary Search**

Binary search is a faster method designed for sorted lists. It repeatedly splits the list in half to eliminate large portions of data with each comparison.

**Procedure:**

1. Begin with the full sorted list.
2. Identify the middle element.
3. Compare the target with the middle:
   * If equal, return the index or element.
   * If the target is smaller, search the lower half.
   * If larger, search the upper half.
4. Repeat until the target is found or the sublist is empty.

**Efficiency:**

* **Time Complexity:** O(log n), since each step cuts the search range in half.

**Example:** In the array [3, 5, 7, 9, 11], to find 7, binary search checks the middle (which is 7) and finds the target immediately.

**Comparison of Time Complexities**

* **Linear Search:**
  + O(n) in the worst case, requiring up to n comparisons.
* **Binary Search:**
  + O(log n), drastically reducing comparisons by halving the list each time.

**Discuss when to use each algorithm based on the dataset size and order.**

* + 1. **Linear Search -**

**When to Use:**

* **Unsorted Data:** Ideal for datasets that aren’t sorted, as it doesn’t rely on any specific order.
* **Small Collections:** Performance differences between linear and binary search are minimal for small lists.
* **Occasional Lookups:** In cases where searches are rare and the dataset is small, linear search offers a simple and sufficient solution.

**Advantages:**

* Very easy to implement.
* Can be used on both sorted and unsorted data.

**Disadvantages:**

* Becomes inefficient on larger datasets, since it has to check every element.
* Performance drops linearly as the dataset grows.
  + 1. **Binary Search**

**When to Use:**

* **Sorted Datasets:** Only applicable when data is already sorted or can be sorted in advance.
* **Larger Data Volumes:** Offers much faster search times for big collections due to its logarithmic complexity.
* **Frequent Searching:** Perfect for static datasets with repeated lookup operations.

**Advantages:**

* Significantly more efficient than linear search for large datasets.
* Excellent for quickly finding elements in sorted collections.

**Disadvantages:**

* Requires data to be sorted beforehand, which may introduce extra processing.
* Slightly more complex logic compared to linear search.