4. Analysis:

Time Complexity:

Linear Search:

- Best Case: O(1) → if the element is the first in the list.

- Average Case: O(n/2) → generally written as O(n).

- Worst Case: O(n) → if the element is not present or is the last one.

Binary Search:

- Best Case: O(1) → if the middle element is the target.

- Average Case: O(log n)

- Worst Case: O(log n)

Space Complexity:

- Both algorithms require O(1) additional space, assuming the input list is already in memory.

Use Cases:

- Use Linear Search:

• When the list is unsorted.

• When only a few elements need to be checked.

• When working with small data sets.

- Use Binary Search:

• When the dataset is large and sorted.

• When the list is frequently searched and infrequently modified (to avoid re-sorting).

Conclusion:

- Linear search is simple and universal but inefficient for large datasets.

- Binary search is powerful for performance but requires pre-sorted data.

- For scalable systems like a library database, binary search should be preferred with a one-time sort or indexing mechanism.