**EX 6**

**Understanding Search Algorithms**

**Linear Search**

* **Description**: A straightforward search algorithm that checks each element in the list sequentially until the desired element is found or the list ends.
* **Time Complexity**:
  + Best-case: O(1) (if the element is at the first position)
  + Average-case: O(n)
  + Worst-case: O(n)
* **Space Complexity**: O(1) (no additional space required)

**Binary Search**

* **Description**: A search algorithm that finds the position of a target value within a sorted array. It compares the target value to the middle element of the array; if they are not equal, it eliminates half of the remaining elements and repeats the process on the remaining half.
* **Time Complexity**:
  + Best-case: O(1)
  + Average-case: O(log n)
  + Worst-case: O(log n)
* **Space Complexity**: O(1) (no additional space required for iterative version; O(log n) for recursive version due to call stack)

**Analysis**

**Time Complexity Comparison**

1. **Linear Search**:
   * Best-case: O(1) (element is the first one in the list)
   * Average-case: O(n)
   * Worst-case: O(n)
2. **Binary Search**:
   * Best-case: O(1)
   * Average-case: O(log n)
   * Worst-case: O(log n)

**When to Use Each Algorithm**

1. **Linear Search**:
   * **Unsorted Data**: Linear search is suitable when the dataset is unsorted, as it does not require any preconditions about the order of elements.
2. **Binary Search**:
   * **Sorted Data**: Binary search requires the dataset to be sorted. It is efficient for large datasets due to its O(log n) time complexity.