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

**Linear Search**

* **Definition**: Linear search is a straightforward algorithm that checks each element in the list sequentially until the desired element is found or the list ends.
* **Time Complexity**: O(n), where n is the number of elements in the list.
* **Use Case**: Suitable for unsorted or small datasets.

**Binary Search**

* **Definition**: Binary search is an efficient algorithm that finds the position of a target value within a sorted array. It repeatedly divides the search interval in half.
* **Time Complexity**: O(log n), where n is the number of elements in the list.
* **Use Case**: Suitable for large, sorted datasets.

1. **Compare the time complexity of linear and binary search.**

**Time Complexity**

1. Linear Search:
   * Best Case: O(1) – The desired element is the first in the list.
   * Average Case: O(n) – The desired element is in the middle of the list.
   * Worst Case: O(n) – The desired element is the last in the list or not present.
2. Binary Search:
   * Best Case: O(1) – The desired element is the middle element of the list.
   * Average Case: O(log n) – The list is divided in half each time.
   * Worst Case: O(log n) – The element is not present, requiring maximum comparisons.
3. **Discuss when to use each algorithm based on the data set size and order.**

* **Linear Search**:
  + **Unsorted Data**: When the data is unsorted, linear search is the only option.
  + **Small Datasets**: For small datasets, the simplicity of linear search may outweigh its inefficiency.
* **Binary Search**:
  + **Sorted Data**: Binary search requires that the data be sorted beforehand.
  + **Large Datasets**: For lar