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

**A.** Linear Search:

Linear Search is a straightforward algorithm used to find an element in a list. It works by sequentially checking each element of the list until the desired element is found or the list is exhausted.

Characteristics:

* Time Complexity: O(n), where n is the number of elements in the list. In the worst case, you might need to check every element.
* Space Complexity: O(1), as it only requires a constant amount of additional memory for variables.
* Unsorted Requirement: The list does not need to be sorted.

Binary Search:

Binary Search is a more efficient algorithm for finding an element in a sorted list. It works by repeatedly dividing the search interval in half.

Characteristics:

* Time Complexity: O(log n), where n is the number of elements in the list. It reduces the search space by half each step.
* Space Complexity: O(1) for iterative implementations or O(log n) for recursive implementations due to call stack usage.
* Sorted Requirement: The list must be sorted for binary search to work correctly.

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

**A.** Time Complexity:

* Linear Search:
  + Best Case: O(1)
  + Average/Worst Case: O(n)
* Binary Search:
  + Best Case: O(1)
  + Average/Worst Case: O(log n)

**Q. Discuss when to use each algorithm based on the data set size and order.**

**A.** Linear Search: Suitable for unsorted datasets or small datasets where the overhead of sorting is not justified.

Binary Search: Ideal for large, sorted datasets where the faster search time justifies the initial sorting overhead.