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

**Linear Search**

* **Description:** Linear search is a basic algorithm that examines each element in a list one by one until it finds the target element or reaches the end of the list.
* **Time Complexity:**
  + **Best Case:** O(1) (if the target element is the first one)
  + **Average Case:** O(n)
  + **Worst Case:** O(n)

**Binary Search**

* **Description:** Binary search is a more efficient algorithm used on sorted lists. It repeatedly splits the search range in half, comparing the target value with the middle element of the range, and narrows down the search interval based on this comparison.
* **Time Complexity:**
  + **Best Case:** O(1) (if the target element is the middle one)
  + **Average Case:** O(log n)
  + **Worst Case:** O(log n)

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

* **Linear Search:**
  + **Best Case:** O(1) (if the target element is the first one)
  + **Average Case:** O(n)
  + **Worst Case:** O(n)
* **Binary Search:**
  + **Best Case:** O(1) (if the target element is the middle one)
  + **Average Case:** O(log n)
  + **Worst Case:** O(log n)

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

**Linear Search:**

* Best for small or unsorted datasets.
* Easy to implement and does not require the data to be pre-sorted.
* Suitable when the cost of sorting the data exceeds the benefits of faster search times.

**Binary Search:**

* Ideal for large, sorted datasets.
* Requires the data to be sorted beforehand, which may involve an initial cost but allows for significantly faster searches afterwards.
* Most effective when performing multiple searches on a static dataset, as the initial sorting cost is spread over many search operations.