1. **Linear Search:**

* **Algorithm:** Linear search sequentially checks each element of the list until it finds the target value or reaches the end of the list.
* **Time Complexity:** O(n), where n is the number of elements in the list.
* **Use Case:** Suitable for small to medium-sized unsorted lists or when the cost of sorting is higher than the search itself.

**Binary Search:**

* **Algorithm:** Binary search works on a sorted list by repeatedly dividing the search interval in half. If the target value is less than the middle element, the search continues in the left half; otherwise, it continues in the right half.
* **Time Complexity:** O(log n), where n is the number of elements in the list.
* **Use Case:** Suitable for large-sized sorted lists where frequent searches are needed.

4. **Analysis**

**Time Complexity:**

* **Linear Search:** O(n) – The search goes through each element one by one.
* **Binary Search:** O(log n) – The search interval is halved each time, making it much faster for large datasets.

**When to Use Each Algorithm:**

* **Linear Search:**
  + Use when the list is unsorted or small.
  + Sorting the list first might not be efficient for one-time or infrequent searches.
* **Binary Search:**
  + Use when the list is already sorted or when you perform frequent searches on a large dataset.
  + The initial cost of sorting is amortized over many searches, making it more efficient in the long run.