**Linear Search:**

Linear search is a simple search algorithm that sequentially checks each element in a list until the desired element is found or all elements have been checked. Here's how it works:

* Start from the beginning of the list.
* Compare each element with the target value.
* If the element matches the target, return its index (or value).
* If the end of the list is reached without finding the target, return a "not found" indication.

**Time Complexity:** O(n), where n is the number of elements in the list. This means the time taken increases linearly with the size of the input list.

**Binary Search:**

Binary search is a more efficient algorithm but requires the list to be sorted:

* It works by repeatedly dividing the search interval in half.
* Compare the middle element of the list with the target value.
* If the target matches the middle element, return its index (or value).
* If the target is less than the middle element, narrow the search to the lower half.
* If the target is greater, narrow it to the upper half.
* Repeat until the target is found or the search interval is empty.

**Time Complexity:** O(log n), where n is the number of elements in the list. Binary search operates on sorted arrays and reduces the search space by half in each step

**Time Complexity Comparison:**

* **Linear Search:** O(n)
* **Binary Search:** O(log n)

**When to Use Each Algorithm:**

* **Linear Search:** Use when the list is not sorted or when you need to search through a small list. It's straightforward and easy to implement.
* **Binary Search:** Use when the list is already sorted. It's much faster for large datasets because it reduces the search space exponentially with each comparison.