**1. Linear Search**

**Description:**  
• Checks each element one by one.  
• Doesn’t require sorted data.  
• Simple to implement.

**Algorithm:**

1. Start from index 0.
2. Compare each element with the target.
3. If found, return index.
4. If not found till the end, return -1.

**Time Complexity:**

| **Case** | **Complexity** |
| --- | --- |
| Best | O(1) (found at start) |
| Average | O(n) |
| Worst | O(n) |

**2. Binary Search**

**Description:**  
• Works only on sorted data.  
• Repeatedly divides the array in half to find the target.

**Algorithm (in words):**

1. Find the middle element.
2. If it matches the target, return index.
3. If target < mid, search in left half.
4. If target > mid, search in right half.
5. Repeat until found or array is empty.

**Time Complexity:**

| **Case** | **Complexity** |
| --- | --- |
| Best | O(1) (found at middle) |
| Average | O(log n) |
| Worst | O(log n) |

**3. Comparison**

| **Feature** | **Linear Search** | **Binary Search** |
| --- | --- | --- |
| Data requirement | Unsorted / Sorted | Sorted only |
| Time Complexity | O(n) | O(log n) |
| Simplicity | Simple | More complex |
| Use case | Small or unsorted datasets | Large sorted datasets |

**4. Understanding Asymptotic Notation**

**Big O Notation:**  
• Big O notation describes the upper bound of an algorithm’s runtime or space requirement as the input size increases.  
• It helps in evaluating how an algorithm performs at scale, independent of hardware.  
• Examples:

* O(1): Constant time
* O(n): Linear time
* O(log n): Logarithmic time

**Best, Average, and Worst-Case Scenarios:**

| **Scenario** | **Meaning** |
| --- | --- |
| Best case | Minimum time the algorithm takes (ideal) |
| Average case | Expected time across typical inputs |
| Worst case | Maximum time for the hardest input |

**5. When to Use Which**

| **Situation** | **Recommended Search** |
| --- | --- |
| Small dataset | Linear Search |
| Data is unsorted | Linear Search |
| Data is sorted and large | Binary Search |
| Search is performed many times | Binary Search (after sorting once) |