**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 datasets or unsorted data | | Large sorted datasets |

1. **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) |