Searching – Search in Sorted Array

1. Introduction

Searching in a **sorted array** refers to finding the position of a target element when the elements of the array are arranged in **ascending or descending order**.

Since the array is already sorted, we can use **efficient searching techniques** instead of checking each element sequentially.

---

2. Why Sorted Arrays Are Important for Searching

Sorted arrays make searching faster because:

- Elements follow a predictable order
- Half of the search space can be eliminated in each step
- Efficient algorithms like **Binary Search** can be applied
- Performance improves significantly for large datasets

---

3. Common Searching Methods for Sorted Arrays

1️⃣ Linear Search (Less Efficient)

- Works on sorted arrays
- Does not use sorting advantage
- Time complexity remains O(n)

2️⃣ Binary Search (Preferred)

- Uses the sorted nature of data
- Divides the array into halves
- Time complexity becomes O(log n)

---

4. Basic Idea of Searching in a Sorted Array

The idea is to:

- Compare the target element with the middle element
- If equal, return the position
- If smaller, search the left part
- If larger, search the right part
- Repeat until found or search space ends

This method ensures fewer comparisons.

---

5. Logic for Searching in a Sorted Array (Plain English)

1. Set two pointers: start and end
2. Find the middle index
3. Compare the middle element with the target
4. If equal, element is found
5. If target is smaller, move to left half
6. If target is larger, move to right half
7. Repeat until start exceeds end

---

6. Visualization Example

Sorted array:

```
[5, 10, 15, 20, 25, 30]
```

Target:

```
20
```

Search process:

```
Middle → 15 (target > 15)
Right half → [20, 25, 30]
Middle → 25 (target < 25)
Left half → [20]
Found
```

7. Time and Space Complexity

Aspect	Complexity
Time Complexity	O(log n)
Space Complexity	O(1)

(Binary search approach)

8. Advantages of Searching in Sorted Array

- Faster searching
- Efficient for large datasets
- Fewer comparisons
- Better performance than unsorted search

9. Limitations

- Array must be sorted beforehand
- Sorting itself may take extra time
- Not suitable for frequently changing data
- Insertion and deletion are costly

10. Real-World Applications

- Searching student records by roll number

- Dictionary word lookup
- Searching prices in sorted catalogs
- Database indexing
- System-level search operations

11. Difference Between Sorted and Unsorted Search

Feature	Sorted Array	Unsorted Array
Search Technique	Binary Search	Linear Search
Time Complexity	O(log n)	O(n)
Performance	High	Low

12. Summary

- Sorted arrays allow efficient searching
- Binary search is the preferred method
- Time complexity is O(log n)
- Suitable for large datasets
- Requires sorted data