Searching – Linear Search

1. Introduction

Linear Search is the simplest searching technique used to **find a specific element in a list or array**.

It works by **checking each element one by one**, starting from the beginning, until the required element is found or the list ends.

Because of its simplicity, linear search is often the **first searching algorithm** taught to beginners.

---

2. What is Linear Search?

Linear search is a **sequential search algorithm** where each element is compared with the target value.

If a match is found:

- The position of the element is returned

If no match is found:

- The search is declared unsuccessful

---

3. Why Do We Use Linear Search?

Linear search is used because:

- It is easy to understand and implement
- It works on **both sorted and unsorted arrays**
- No special data arrangement is required
- Suitable for small datasets

Although it is not the most efficient algorithm, it is very useful for beginners and small inputs.

---

4. Basic Idea Behind Linear Search

The idea is straightforward:

- Start from the first element
- Compare each element with the target value
- Stop when a match is found
- If the end of the array is reached without a match, the element does not exist

5. Logic for Linear Search (Plain English)

1. Start from the first element of the array
2. Compare the current element with the target value
3. If both are equal, stop the search
4. If not equal, move to the next element
5. Repeat until the element is found or the array ends
6. If the array ends, return “not found”

6. Visualization of Linear Search

Given array:

```
[10, 25, 30, 45, 60]
```

Target value:

```
30
```

Search flow:

```
10 → 25 → 30 (Found)
```

The search stops as soon as the element is found.

7. Types of Linear Search Results

- **Best Case:**

  Target found at the first position

- **Average Case:**

  Target found somewhere in the middle

- **Worst Case:**

  Target found at the last position or not found at all

---

8. Time and Space Complexity

Case	**Time Complexity**
Best Case	O(1)
Average Case	O(n)
Worst Case	O(n)

- **Space Complexity:** O(1)

  (No extra memory required)

---

9. Advantages of Linear Search

- Simple and easy to implement
- No requirement of sorted data
- Works with all data types
- Ideal for small datasets

---

10. Limitations of Linear Search

- Inefficient for large datasets
- Slower compared to advanced searching techniques
- Requires checking each element sequentially

---

11. Real-World Applications

- Searching a name in a contact list
- Finding a roll number in an attendance list
- Searching a word in a small document
- Finding an item in an unsorted inventory
- Checking presence of a value in sensor data

12. Comparison with Other Search Methods

Feature	**Linear Search**	**Binary Search**
Data Requirement	Unsorted allowed	Must be sorted
Time Complexity	O(n)	O(log n)
Implementation	Simple	Complex
Efficiency	Low	High

13. Summary

- Linear search checks elements one by one
- Works on any array (sorted or unsorted)
- Easy to implement
- Time complexity is O(n)
- Suitable for small datasets