**Array Representation and Advantages:**

Arrays are one of the most basic and widely used data structures in programming. They are stored in **contiguous memory locations**, meaning all elements are placed next to each other in memory. Because of this, arrays allow **direct access** to any element using its index. For example, accessing arr[3] takes the same amount of time no matter how big the array is, which makes it very efficient.

One of the main advantages of arrays is **fast element access** using indexing, which is done in constant time (O(1)). Arrays are also simple to implement and are supported in almost every programming language. They are best when we know the size of the data in advance and don’t need to frequently insert or remove elements.

**Time Complexity Analysis of Array Operations:**

* **Add (Insertion at end):** O(1) if there is space; but O(n) if elements need to be shifted (e.g., inserting at the beginning or middle).
* **Search:** O(n) in general, unless the array is sorted. For sorted arrays, we can use binary search (O(log n)).
* **Traverse:** O(n), as we need to visit each element once.
* **Delete:** O(n) because elements need to be shifted after deletion (unless we’re deleting the last element, which is O(1)).

So while some operations like access are very fast, others like insert or delete in the middle are slower in arrays.

**Limitations and When to Use Arrays:**

Arrays come with a few limitations. First, they have a **fixed size**, so we must know in advance how many elements we want to store. If we underestimate the size, we may run out of space; if we overestimate, memory is wasted. Also, inserting or deleting elements (especially not at the end) is slow because it requires shifting elements.

Despite these limitations, arrays are useful when:

* We need **fast access** to elements using index.
* The number of elements is **known and fixed**.
* We’re working with **simple data** that doesn’t need dynamic resizing.

In cases where we need frequent insertions or deletions, or dynamic size, it’s better to use other data structures like linked lists or dynamic arrays (e.g., ArrayList in Java or Python lists).