**1.**

* **Explain how arrays are represented in memory and their advantages.**

**Ans :** Arrays are a fundamental and linear data structure. They are represented in memory as a contiguous block of memory locations. Each element in the array is of the same data type and can be accessed using an index number that starts from 0.

**Advantages:**

* **Constant Time Access:** we can get any element using its index in constant time, O(1).
* **Low Overhead:** Arrays have less overhead as compared to other data structures like lists or trees.
* **Cache-Friendly:** Because arrays are contiguous blocks of memory, they are cache-friendly and can derive performance benefits from it.

**4.**

* **Analyze the time complexity of each operation (add, search, traverse, delete).**

**Ans :**

**Time Complexity**

* **Add Operation:** Time Complexity: O(1) (constant time if there is space)
* **Search Operation:** Time Complexity: O(n) (linear time, as each element may need to be checked)
* **Traverse Operation:**  Time Complexity: O(n) (linear time, as each element needs to be visited)
* **Delete Operation:** Time Complexity: O(n) (linear time, as each element after the deleted one needs to be shifted)
* **Discuss the limitations of arrays and when to use them.**

**Ans :** Limitations of array are discussed below :

* **Fixed Size**: Arrays have a fixed size once created. If more space is needed, a new array must be created and elements copied over.
* **Inefficient Insertions and Deletions**: Inserting or deleting elements requires shifting elements, which can be inefficient in terms of time complexity.
* **Lack of Flexibility**: Arrays are not suitable for dynamic data where the size may change frequently.

**When to Use Arrays**:

* Use arrays when the number of elements is known beforehand and doesn't change often. For example, fixed-size collections where performance is critical.