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

**A.** Array Representation in Memory:

* Contiguous Memory Allocation: Arrays are stored in contiguous memory locations. This means each element is located at a successive memory address.
* Indexing: The position of an element in an array is accessed using an index, starting from 0 for the first element. For example, in an array arr, arr[0] accesses the first element, arr[1] accesses the second element, and so on.
* Fixed Size: Arrays have a fixed size defined at the time of creation. This means the size of the array cannot be changed once it is allocated.

Advantages of Arrays:

* Constant Time Access: Since arrays use contiguous memory locations, accessing an element by its index takes constant time, O(1).
* Ease of Iteration: Arrays allow for easy iteration over elements, making traversal straightforward.
* Memory Efficiency: Arrays are memory-efficient as they do not require extra storage for pointers or references.

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

**A.** Time Complexity:

* Add Employee: O(1) - Adding an employee to the next available index is a constant time operation.
* Search Employee: O(n) - In the worst case, we may need to search through all employees.
* Traverse Employees: O(n) - We need to visit each employee once.
* Delete Employee: O(n) - In the worst case, we may need to search through all employees to find the one to delete. Removing an employee involves moving the last element to the deleted position, which is a constant time operation.

**Q. Discuss the limitations of arrays and when to use them.**

**A.** Limitations of Arrays:

* Fixed Size: Arrays have a fixed size, which means the number of employee records cannot exceed the initial array size. This limits the scalability.
* Costly Insertions and Deletions: Although we have optimized the delete operation by replacing the deleted element with the last element, in general, insertions and deletions can be costly if we need to maintain order.
* Wasted Space: If the array size is much larger than the number of employees, it can lead to wasted memory space.

When to Use Arrays:

* Arrays are suitable when the number of elements is known in advance and remains relatively stable.
* They are useful when you need fast access to elements by index.
* Arrays are efficient for static data structures where the dataset does not change frequently.