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

Arrays are contiguous blocks of memory that store elements of the same data type. Each element in an array can be accessed directly using its index.

Advantages of arrays:

Fast access time (O(1)) for any element if you know its index.

Simple and easy to use.

Memory efficient for storing elements of the same type.

Good cache locality which can improve performance.

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

Time complexity of operations:

Add (addEmployee):

Best/Average case: O(1) if there's space in the array.

Worst case: O(n) if the array needs to be resized.

Search (searchEmployee): O(n) - linear search through the array.

Traverse (traverseEmployees): O(n) - iterate through all elements.

Delete (deleteEmployee): O(n) - need to search for the employee first.

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

a>Fixed size: Arrays have a fixed size once created. Resizing requires creating a new array and copying elements, which is expensive.

b>Inefficient insertions/deletions: Inserting or deleting elements in

the middle of the array requires shifting other elements.

c>Wasted space: If the array is too large for the data, it wastes memory.

d>Poor for dynamic data: Not ideal when the number of elements changes frequently.

When to use arrays:

When we know the exact number of elements in advance.

When we need fast access to elements by index.

When memory usage needs to be predictable.

When the data is relatively static (not many insertions/deletions).

When we are implementing more complex data structures (like hash tables or heaps).