4. Employee Management System

How arrays are represented in memory and their advantages

- **Memory Representation**:

  - **Contiguous Memory Allocation**: Arrays are stored in contiguous memory locations. Each element is located at an index offset from the base address of the array. For instance, if the base address is `B` and the element size is `S`, the address of the element at index `i` is `B + i * S`.
  - **Fixed Size**: Arrays have a fixed size, which is determined at the time of allocation. The size cannot be changed during runtime.

Analysis:

Add:

- Best Case: O(1) (when adding at the end and space is available).
- Average Case: O(1) on average due to occasional resizing.
- Worst Case: O(n) (when resizing the array).

Search:

- Linear Search: O(n) (need to check each element).

Traverse: O(n) (visit each element once).

Delete:

- Best Case: O(1) (when deleting the last element).
- Worst Case: O(n) (when deleting an element from the start or middle, requiring shifts).

Limitations of Arrays

1. **Fixed Size**:

   - Arrays have a fixed size, which can lead to wasted memory if the allocated size is larger than needed or frequent resizing if the size is smaller than needed.
2. **Resizing Overhead**:

   - Resizing an array involves creating a new array and copying all elements, which is time-consuming.
3. **Insertion and Deletion**:

   - Insertion and deletion operations, especially in the middle of the array, require shifting elements, leading to O(n) time complexity.
4. **Inefficient Memory Usage**:

- ○ Large arrays may lead to inefficient memory usage if the array is sparsely populated.