Employee Management System

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

Ans. An array is a collection of elements stored in contiguous memory locations, where each element is accessed using an index starting from 0. The memory address of any element in the array can be directly calculated using the formula: address = base address + (index × size of each element).

This direct address computation allows for efficient, constant-time O(1) access to any element by its index.

Arrays offer several advantages, including fast access to elements, a simple and linear memory layout that is easy to manage, and minimal overhead, making them highly efficient for handling fixed-size collections of data where the size is known in advance.

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

Ans. Adding an employee to an array has a time complexity of O(1) as it simply inserts the employee record at the next available index.

Searching for an employee requires a linear search through the array elements, resulting in a time complexity of O(n).

Traversing all employee records also takes O(n) time since each element must be accessed sequentially.

Deleting an employee involves first locating the employee, which takes O(n) time, and then shifting the subsequent elements one position to fill the gap, which also takes O(n), making the overall deletion operation O(n).

1. Discuss the limitations of arrays and when to use them?

Ans. Limitations of Arrays:

* Fixed size: Cannot grow dynamically after initialization.
* Inefficient deletions: Requires shifting elements on deletion.
* Inefficient insertions at arbitrary positions: Requires shifting elements.
* No built-in methods for resizing or easy dynamic operations like collections.

When to Use Arrays**:**

* When the number of elements is known and fixed.
* When memory predictability and fast random access via index are needed.
* In performance-critical, low-level, or memory-constrained applications.