**Exercise 4: Employee Management System**

1. Explain how arrays are represented in memory and their advantages.
   * Array Representation in Memory
     1. Contiguous Memory Allocation: It simply means that the elements of the arrays are stored in contiguous memory locations. It simply means that the memory locations of elements in the array are placed together one after the other.
     2. Fixed Size: At the time of declaration, a fixed amount of memory space is allocated to the array. This amount cannot be altered during the runtime.
     3. Indexing: Elements of an array are indexed starting from 0 generally. It, therefore, provides access to the elements in constant time through their index, that is in O(1).
   * Advantages of Arrays
     1. Direct Access: The elements of an array can be directly accessed through their indices, so this retrieval is going fast, i.e., of the order O(1).
     2. Memory Efficiency: Arrays have less overhead as they do not store additional information like pointers.
2. Analyze the time complexity of each operation (add, search, traverse, delete).
   * Add Employee: O(1) - When space is there in the array then in constant time always, an employee can be added to the last.
   * Search Employee: O(n): At most all employees need to be searched in a worst-case to get a match.
   * Traverse Employees: O(n) - To traverse all Employees, we need to run a loop with the entire array
   * Delete Employee: O(n) - Time taken to search the employee that we want to delete is O(n) and to update the array will take O(1) which ultimately becomes the O(n).
3. Discuss the limitations of arrays and when to use them.
   * Limitations of Arrays
     1. Fixed Size: The size of the array has to be fixed while declaration. They can not grow dynamically. That means, after the array is full, you can't add more elements without creating a new, bigger array.
     2. Inefficient Insertions/Deletions: For the middle of an array, insertions or deletions can be very slow since all following elements have to be shifted.
     3. Memory Waste: Unused elements of array lead to memory waste.
   * When to Use Arrays
     1. Static Data: In the case when one knows in advance the number of elements, and that number changes rarely, then an array is a good fit.
     2. Fast Access: If you want to gain quick access to the elements and you already know the indeces the elements belong to, then arrays are a recommended way.
     3. Simple Data Storage: When there is a need for simple, small data storage and manipulation then the array provides a simple way of doing that.