**COGNIZANT DIGITAL NURTURE – 3.0**

**JAVA FSE**

**WEEK – 1 EXERCISES**

**DATA STRUCTURES AND ALGORITHMS**

**Exercise 4: Employee Management System**

**Step 1: Understand Array Representation:**

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

**Array Representation in Memory**

* **Contiguous Memory**: Arrays are stored in contiguous memory locations. This means that all elements of the array are located next to each other in memory.
* **Indexing**: Each element in the array can be accessed directly using its index. The index is multiplied by the size of each element to get the address of the desired element.
* **Fixed Size**: The size of the array is fixed at the time of its creation and cannot be changed.

**Advantages of Arrays**

* **Direct Access**: Elements can be accessed in constant time O(1) due to direct indexing.
* **Efficient Traversal**: Iterating over elements is efficient due to contiguous memory.
* **Memory Management**: Memory allocation is straightforward since all elements are stored together.

**Step 2: Setup**

***Refer Program Files***

**Step 3: Implementation**

***Refer Program Files***

**Step 4: Analysis**

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

**Time Complexity:**

* **Add Operation: O (1**) – Adding an element at the end of the array is a constant time operation if there's space available.
* **Search Operation: O (n)** – In the worst case, the search may need to check all elements in the array.
* **Traverse Operation: O (n)** – Traversing involves visiting each element once.
* **Delete Operation: O (n**) – Deleting involves searching for the element (O(n)) and then shifting subsequent elements (O (n)).

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

**Limitations of Arrays**

* **Fixed Size**: Arrays have a fixed size, which means their capacity cannot be adjusted dynamically. This can lead to wasted space or insufficient capacity.
* **Inefficient Deletion and Insertion**: Insertion and deletion operations can be inefficient if they involve shifting elements.
* **Memory Usage**: Contiguous memory allocation can be an issue if a large block of memory is not available.

**When to Use Arrays**

* **Static Data**: When the number of elements is known and does not change frequently.
* **Direct Access**: When constant time access to elements is required.
* **Memory Contiguity**: When contiguous memory allocation is beneficial for performance reasons