Employee Management System

- Array Representation in Memory:

Contiguous Memory Allocation: Arrays are stored in contiguous memory locations. This means that all elements of the array are placed next to each other in memory.

Indexing: Each element of the array can be accessed directly using its index, which provides fast access times.

Fixed Size: The size of an array is fixed at the time of its creation. You cannot change the size of an array once it has been allocated.

- Advantages of Arrays:

Fast Access: Direct access to elements using the index provides

O (1) time complexity for accessing an element.

Simple Data Structure: Easy to understand and use for storing a collection of elements.

Efficient Memory Usage: Minimal memory overhead compared to other data structures like linked lists.

- Time Complexity:

Add Operation: O (1) if the array is not full; otherwise, O(n) if resizing the array is necessary (which is not handled in the current implementation).

Search Operation: O(n) in the worst case because we may need to traverse the entire array.

Traverse Operation: O(n) since every element needs to be visited.

Delete Operation: O(n) because it involves searching for the element and then shifting the subsequent elements.

- Limitations of Arrays:

Fixed Size: The size of the array must be defined at the time of its creation and cannot be changed. This can lead to wasted memory or insufficient space.

Inefficient for Dynamic Data: Adding or deleting elements from the middle of the array requires shifting elements, which is inefficient.

Linear Search: Searching for an element requires O(n) time in the worst case, which is not optimal for large datasets.

- When to Use Arrays:

Static Data: When the size of the dataset is known in advance and does not change.

Fast Access: When you need constant time access to elements by their index.

Simple Use Cases: When simplicity is more important than flexibility or efficiency in adding/deleting elements.