**EMPLOYEE-MANAGEMENT SYSTEM**

**How Arrays are Represented in Memory**

Arrays are stored in contiguous memory locations, which means that all elements are placed next to each other in memory. Arrays provide constant-time (O(1)) access to elements using indices. The address of any element can be calculated if the base address and size of elements are known. For example, if the base address is ‘1000’ and each element occupies 4 bytes, meaning is on integer type then the address at an index i would be 1000+4\*i.

* **Advantages**:
  + **Fast Access**: Direct access to elements using indices.
  + **Predictable Memory Usage**: Fixed size allows for easy calculation of memory requirements.
  + **Easy to manage** : Since arrays don’t have any overhead like pointers or references, it makes it easier to manage and store.

**Time Complexity of Each Operation**

* **Add**: O(1) - Adding an employee to the end of the array.
* **Search**: O(n) - Searching for an employee requires checking each element.
* **Traverse**: O(n) - Traversing the array involves visiting each element.
* **Delete**: O(n) - Deleting an employee involves searching for the employee and shifting subsequent elements.

**Limitations of Arrays**

* **Fixed Size**: Arrays have a fixed size, so you must know the maximum number of elements in advance or use resizing techniques.
* **Inefficient Deletions and Insertions**: Deleting or inserting elements involves shifting elements, leading to O(n) time complexity.
* **No dynamic Memory Allocation**: The size of an array cant be changed at run-time.

**When to Use Arrays**

* **Static Data Size**: When the number of elements is known in advance and does not change frequently.
* **Fast Access Required**: When constant-time access to elements using indices is necessary.
* **Cache-Friendly Operations**: When operations involve frequent traversal, benefiting from the contiguous memory allocation.