**Understand Array Representation**

**Array Representation in Memory:**

* **Contiguous Memory Allocation:** Arrays are stored in contiguous memory locations. This means each element is placed right next to the previous one in memory, allowing for efficient indexing.
* **Indexing:** The primary advantage of arrays is that they allow for O(1) time complexity for accessing any element by its index. This is because the address of any element can be calculated using its index and the base address of the array.
* **Advantages of Arrays:**
  + **Direct Access:** Fast access to elements using index.
  + **Cache-Friendly:** Due to contiguous memory allocation, arrays are cache-friendly.
  + **Simple Data Structure:** Easy to understand and implement.

**Analysis**

**Time Complexity Analysis:**

1. **Add Employee:**
   * **Time Complexity:** O(1) (assuming the array is not full)
2. **Search Employee:**
   * **Time Complexity:** O(n) (linear search through the array)
3. **Traverse Employees:**
   * **Time Complexity:** O(n) (visit each element in the array)
4. **Delete Employee:**
   * **Time Complexity:** O(n) (linear search to find the element and shift subsequent elements)

**Limitations of Arrays:**

* **Fixed Size:** Arrays have a fixed size, which means you need to allocate enough space beforehand. This can lead to either wasted memory or insufficient space.
* **Insertion and Deletion:** Inserting and deleting elements can be inefficient as it may require shifting elements.
* **Better Alternatives:** For dynamic data management, other data structures like ArrayList, LinkedList, HashMap, etc., can be more efficient.