**Arrays are represented in memory:**

Arrays are represented in memory as a contiguous block of memory locations, each of which can store a value of a specific data type. The memory locations are indexed by an integer, starting from 0, which allows for efficient access and manipulation of the array elements.

**Advantages of arrays:**

1. **Efficient memory usage:** Arrays store elements in contiguous memory locations, which makes them memory-efficient.
2. **Fast access:** Arrays allow for fast access to elements using their index, which makes them suitable for applications that require frequent access to elements.
3. **Simple implementation:** Arrays are easy to implement and understand, making them a popular choice for many applications.

**The time complexity of each operation:**

1. **Add operation:** The time complexity of the add operation is O(1) because it simply adds an element to the end of the array.
2. **Search operation:** The time complexity of the search operation is O(n) because it iterates through the entire array to find an employee by ID.
3. **Traverse operation:** The time complexity of the traverse operation is O(n) because it iterates through the entire array to print employee information.
4. **Delete operation:** The time complexity of the delete operation is O(n) because it iterates through the entire array to find an employee by ID and then shifts elements to the left to fill the gap.

**The limitations of arrays and when to use them:**

Arrays have several limitations, including:

1. **Fixed size:** Arrays have a fixed size that is determined at compile-time, which can lead to memory waste or insufficient memory.
2. **Slow search and delete:** Arrays have slow search and delete operations because they require iterating through the entire array.
3. **No dynamic resizing:** Arrays do not support dynamic resizing, which can lead to memory issues.

**Arrays are suitable for applications that require**:

1. **Efficient memory usage:** Arrays are memory-efficient because they store elements in contiguous memory locations.
2. **Fast access:** Arrays allow for fast access to elements using their index.
3. **Simple implementation:** Arrays are easy to implement and understand.