**1.Understanding Linked Lists**

Types of Linked Lists

|  |  |
| --- | --- |
| Type | Description |
| Singly Linked List | Each node points to the next node only. Simple, uses less memory. |
| Doubly Linked List | Each node points to both previous and next nodes. Easier reverse traversal. |

Why Use Linked Lists?

* Dynamic size (no need to pre-define capacity like arrays)
* Insertion and deletion are efficient (O(1) at head or known position)
* No shifting of elements like in arrays

**4.Analysis**

Time Complexity Analysis

|  |  |  |
| --- | --- | --- |
| **Operation** | **Time Complexity** | **Explanation** |
| Add Task | O(n) | Traverse to the end to insert |
| Search Task | O(n) | Linear search through the list |
| Traverse List | O(n) | Loop through each node |
| Delete Task | O(n) | Find node by ID and adjust links. |

Advantages of Linked List Over Arrays:

|  |  |
| --- | --- |
| **Advantages** | **Explanation** |
| Dynamic Size | No need to pre-define capacity |
| Efficient Insertion/Deletion | No shifting needed like in arrays |
| Memory Efficient for Inserts | Only allocate memory when needed |

Limitations:

* No direct access by index (unlike arrays)
* Slightly more memory per node (due to next pointer)
* Slower access time for random elements (O(n))