**Exercise 5: Task Management System**

**Scenario:**

You are developing a task management system where tasks need to be added, deleted, and traversed efficiently.

**1. Understand Linked Lists**

**Singly Linked List**

* Each node contains **data** and a **reference to the next node**.
* Efficient for **insertions/deletions** at the beginning.
* One-way traversal only.

**Doubly Linked List**

* Each node has references to **both next and previous nodes**.
* Allows **two-way traversal** and more flexible deletions.
* Requires more memory due to extra pointer.

**Singly Linked List** is sufficient for basic task management with simple insert/delete operations.

**Analysis**

|  |  |  |
| --- | --- | --- |
| **Operation** | **Time Complexity** | **Reason** |
| Add task | O(n) | Traverse to end for insertion |
| Search Task | O(n) | Traverse and match ID |
| Traverse Tasks | O(n) | Visit each node |
| Delete Task | O(n) | Search and update links |

**Time Complexity**

**Advantages of Linked Lists over Arrays**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Array** | **Linked List** |
| **Size** | **Fixed (unless using dynamic structures)** | **Dynamic (grows as needed)** |
| **Insert/Delete (middle)** | **O(n) (needs shifting)** | **O(1) with pointer (if known)** |
| **Memory Usage** | **May allocate unused space** | **Only uses what's needed** |
| **Indexing** | **Fast (O(1) access)** | **Slow (O(n) traversal)** |

**Use linked lists when frequent inserts/deletes are needed and index-based access is not required.**