**Que1. Explain the different types of linked lists (Singly Linked List, Doubly Linked List)?**

Ans: Singly Linked List:

* Structure: Each node contains data and a reference (or link) to the next node in the sequence.
* Traversal: Starts from the head node and follows the links to reach subsequent nodes.
* Memory: Requires less memory compared to doubly linked lists as it only stores one reference per node.
* Operations: Easy to add or remove nodes at the beginning, but more complex at the end or middle due to the need to traverse from the head.

Doubly Linked List:

* Structure: Each node contains data and two references, one to the next node and one to the previous node.
* Traversal: Can be traversed both forward and backward.
* Memory: Requires more memory compared to singly linked lists due to storing two references per node.
* Operations: Easier to add or remove nodes at any position since each node has a reference to its predecessor and successor.

**Que2. Analyse the time complexity of each operation?**

Add Operation:

* Time Complexity: O(n)
* Explanation: Adding a task involves traversing to the end of the list, which requires O(n) time in the worst case.

Search Operation:

* Time Complexity: O(n)
* Explanation: Searching for a task requires traversing the list, which involves O(n) time in the worst case.

Delete Operation:

* Time Complexity: O(n)
* Explanation: Deleting a task requires traversing the list to find the task, which involves O(n) time in the worst case.

Traverse Operation:

* Time Complexity: O(n)
* Explanation: Traversing the list involves visiting each node once, leading to O(n) time complexity.

**Que3. Discuss the advantages of linked lists over arrays for dynamic data?**

* Linked lists can grow or shrink in size dynamically without the need for resizing operations.
* Adding or removing nodes, especially at the beginning or middle of the list, is more efficient (O(1) time complexity) compared to arrays, which require shifting elements.
* Linked lists do not pre-allocate memory, avoiding the issue of wasted space in over-allocated arrays.
* Linked lists serve as a foundation for implementing more complex data structures like stacks, queues, and graphs.