**TASK MANAGEMENT SYSTEM**

A Singly Linked List is a type of data structure where each node points only to the next node in the sequence. Each node contains data and a link to the next node, so you can only move forward through the list. This makes it easy to add or remove nodes, but finding a specific node or moving backward can be tricky.

A Doubly Linked List improves on this by having each node link to both the next and the previous node. This bidirectional linking allows for easier navigation and modification of nodes, but it uses more memory and is a bit more complex to manage.

**Time Complexity for Tasks in Linked Lists:**

* **Add Task**: O(n) – You may need to go to the end of the list to add a new task.
* **Search Task**: O(n) – Finding a specific task requires checking each node.
* **Delete Task**: O(n) – You first find the task and then remove it.
* **Traverse Tasks**: O(n) – You need to go through each node to see all tasks.

**Advantages of Linked Lists Over Arrays:**

* **Dynamic Size**: Linked lists can grow or shrink as needed, while arrays have a fixed size.
* **Efficient Insertions/Deletions**: Adding or removing nodes is quicker in a linked list, especially in the middle or beginning, unlike arrays where elements need to be shifted.
* **Memory Use**: Linked lists only use memory for the elements present, while arrays reserve memory regardless of how full they are. However, linked lists do use extra memory for pointers.

Linked lists are particularly useful when you need frequent insertions and deletions, while arrays are better for situations with a fixed size and less frequent changes.