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

There are mainly two types of linked lists: singly linked list and doubly linked list.

In a **singly linked list**, each node contains two parts: the data and a reference to the next node in the sequence. It allows traversal in only one direction from the head node to the last node. This structure is simple and uses less memory, but does not support backward traversal or efficient deletion from the end.

In a **doubly linked list**, each node contains three parts: the data, a reference to the next node, and a reference to the previous node. This allows traversal in both directions forward and backward. While it provides more flexibility and faster deletion of nodes from both ends, it requires extra memory to store the additional pointer for the previous node.

**#Analyze the time complexity of each operation**

**Time Complexity Analysis of Task Management System**  
In the task management system using a singly linked list:

* **Add Task**: O(n) in the worst case if the task is added at the end, since it requires traversal to the last node.
* **Search Task**: O(n) as each node may need to be checked to find the matching task ID.
* **Traverse Tasks**: O(n) because every node is visited once to display all tasks.
* **Delete Task**: O(n) since it requires traversal to find the node before the one to delete.

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

**Advantages of Linked Lists over Arrays for Dynamic Data**  
Linked lists provide dynamic memory allocation, meaning memory is allocated as needed, unlike arrays which require a fixed size upfront. Inserting or deleting elements in a linked list does not require shifting elements, as it does in arrays. This makes linked lists more efficient when frequent insertions and deletions are involved, especially in the middle or at the beginning of the list. Additionally, linked lists do not need contiguous memory, making them more flexible in memory usage.