**Exercise 5: Task Management System**

**Scenario:**

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

**Steps:**

1. **Understand Linked Lists:**
   * Explain the different types of linked lists (Singly Linked List, Doubly Linked List).

**Singly Linked List:**

* A singly linked list is a linear data structure where each element (node) contains a reference to the next node in the sequence.
* It has the following characteristics:
  + Each node has two components: data and a pointer to the next node.
  + The last node points to null, indicating the end of the list.
  + It's efficient for insertions and deletions but less efficient for random access compared to arrays.

**Doubly Linked List:**

* A doubly linked list is similar to a singly linked list but each node contains two pointers: one pointing to the next node and another pointing to the previous node.
* It has the following characteristics:
  + Each node has three components: data, a pointer to the next node, and a pointer to the previous node.
  + Allows traversal in both forward and backward directions.
  + More complex to implement than singly linked lists due to the additional pointer.

1. **Setup:**
   * Create a class **Task** with attributes like **taskId**, **taskName**, and **status**.
2. **Implementation:**
   * Implement a singly linked list to manage tasks.
   * Implement methods to **add**, **search**, **traverse**, and **delete** tasks in the linked list.
3. **Analysis:**
   * Analyze the time complexity of each operation.

**Time Complexity Analysis:**

* **Add Task:** O(n), where n is the number of nodes, as it may require traversal to the end of the list.
* **Search Task:** O(n) in the worst case, as it may require traversal through all nodes.
* **Traverse Tasks:** O(n), as it involves visiting each node once.
* **Delete Task:** O(n), as it may require traversal to find the node to delete.
  + Discuss the advantages of linked lists over arrays for dynamic data.

**Advantages of Linked Lists over Arrays:**

* **Dynamic Size:** Linked lists can grow and shrink dynamically, whereas arrays have a fixed size.
* **Efficient Insertions/Deletions:** Inserting or deleting elements in a linked list is efficient as it involves adjusting pointers, while in arrays, it requires shifting elements.
* **Memory Utilization:** Linked lists use memory efficiently as they allocate memory as needed, while arrays may waste memory if their capacity is not fully utilized.