**Task Management System**

**1.Understand Linked Lists**

**Linked Lists Overview:**

* **Singly Linked List:**  A singly linked list consists of nodes where each node has a reference (or link) to the next node in the sequence. It allows for efficient insertion and deletion of nodes at the beginning and end, but searching and accessing nodes by index are less efficient compared to arrays.
* **Doubly Linked List:**  A doubly linked list consists of nodes where each node has references to both the next node and the previous node. This allows for more flexible traversal (both forward and backward) and can make some operations more efficient compared to singly linked lists.

**Advantages of Linked Lists over Arrays:**

* **Dynamic Size:** Linked lists can grow and shrink dynamically without the need to resize, unlike arrays which require resizing and copying when full.
* **Efficient Insertions/Deletions:**  Linked lists allow for efficient insertions and deletions from the middle of the list, as only the links need to be updated. Arrays require shifting elements for these operations.

**4.Analysis**

**Time Complexity Analysis:**

1. **Add Operation:**
   * **Best Case:** O (1) – When adding to an empty list or if the list is maintained with a tail pointer (not covered here).
   * **Average Case:** O (n) – Traversing the list to find the end.
   * **Worst Case:** O (n) – Same as the average case, because we may need to traverse the entire list to find the end.
2. **Search Operation:**
   * **Best Case:** O (1) – When the task is at the head of the list.
   * **Worst Case:** O (n) – When the task is at the end of the list or not in the list.
3. **Traverse Operation:**
   * **Time Complexity:** O (n) – All n elements must be visited.
4. **Delete Operation:**
   * **Best Case:** O (1) – When the task to be deleted is at the head of the list.
   * **Average Case:** O (n) – When the task is somewhere in the middle or end, requiring traversal to find it.

* **Worst Case:** O (n) – Same as average case, because the entire list may need to be traversed.

**5.Advantages of Linked Lists over Arrays:**

* **Dynamic Size:** Linked lists can easily grow and shrink, whereas arrays require resizing.
* **Efficient Insertions/Deletions:** Insertions and deletions are more efficient (especially in the middle) because no elements need to be shifted, just the links are updated.
* **Memory Utilization:** Linked lists do not waste memory as they only use as much space as needed for the elements.