**EXERCISE 5: TASK MANAGEMENT SYSTEM**

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

Linked List is a linear data structure, in which elements are not stored at a contiguous location, rather they are linked using pointers. Linked List forms a series of connected nodes, where each node stores the data and the address of the next node.

**Singly Linked List:**

Each node in a singly linked list contains two parts: the data and a pointer to the next node.

The list starts with a head pointer, pointing to the first node, and the last node points to null indicating the end of the list.

Traversal can only be done in one direction, from the head to the end.

**Doubly Linked List:**

Each node in a doubly linked list contains three parts: the data, a pointer to the next node, and a pointer to the previous node.

Traversal can be done in both directions, from the head to the end and from the end back to the head.

**Linked List Operations:**

* **Insertion:** Adding a new node to a linked list involves adjusting the pointers of the existing nodes to maintain the proper sequence. Insertion can be performed at the beginning, end, or any position within the list
* **Deletion:** Removing a node from a linked list requires adjusting the pointers of the neighbouring nodes to bridge the gap left by the deleted node. Deletion can be performed at the beginning, end, or any position within the list.
* **Searching:** Searching for a specific value in a linked list involves traversing the list from the head node until the value is found or the end of the list is reached.

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

**Singly Linked list,**

* Insertion and Deletion at the beginning – O(1)
* Insertion and Deletion at the end– O(n)
* Insertion and Deletion at given position – O(n)
* Traversal – O(n)
* Searching – O(n)

**Doubly Linked list,**

* Insertion and Deletion at the beginning – O(1)
* Insertion and Deletion at the end– O(1)
* Insertion and Deletion at given position – O(n)
* Traversal – O(n)
* Searching – O(n)

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

* The size of a LinkedList can grow or shrink dynamically, so you don’t have to worry about setting an initial size.
* LinkedList is an efficient data structure for inserting or deleting elements in the middle of the list as only the links between the elements must be changed and not shifting.
* Linked lists allocate memory as needed for each element, which can be more efficient if the number of elements is unknown or varies significantly. They avoid the need to over-allocate memory or frequently resize as with dynamic arrays.