**Exercise 5: Task Management System**

**Understanding Linked Lists:-**

**Singly Linked List**

Definition: A singly linked list is a linear data structure in which, for each of the elements called 'nodes,' there is an element having a value known as data and a reference or link pointing to the next node in line.

Structure: Every node is comprised of two parts:

data: The value or information a node holds.

next: Reference to the next node in the list.

**Doubly Linked List**

Definition: A doubly linked list is much like a singly linked list, but with an additional link to the previous node, enabling travel in both forward and backward directions.

Structure: Every node will contain three parts:

data: The value or information the node holds.

next: A reference to the next node in the list.

prev: A reference to the previous node in the list.

**Analysis:-**

**1) Time Complexity**

Add Task: O(n) in the worst scenario since we do have to traverse the complete list to add the task at the end.

Search Task: O(n) in the worst scenario since we do have to go through the complete list of finding the task.

Traverse Tasks: O(n) since we do need to visit each node one time.

Delete Task: O(n) in the worst case because we could require traversing all the way down the list to find and delete the task.

**2) Linked Lists as Data Structures without Disadvantages Over Arrays for Dynamic Data Dynamic Size:-**

Linked to the actual growth and shrinkage through addition or removal of nodes, but it is fixed.

Ease of Insertion/Deletion: Just like accessing elements, insertion and deletion of nodes in a linked list are quite simple and can be done in O(1) time if we know the position, while in an array, shifting elements occur leading to O(n) time complexity. Memory Utilization of Linked Lists Does not Require Contiguous Memory Allocation: This can lead to better memory utilization compared to arrays.