**WEEK1\_ALGORITHMS\_DATA\_STRUCTURES**

**EXERCISE-5**

**Explanation of Linked Lists**

**Different Types of Linked Lists:**

**1.Singly Linked List:**

* **Structure:** Each node contains data and a reference (or link) to the next node in the sequence.
* **Operations:** Efficient for operations where only forward traversal is needed.
* **Example:** Useful for simple list operations where backward traversal is not necessary.

**2.Doubly Linked List:**

* **Structure:** Each node contains data, a reference to the next node, and a reference to the previous node.
* **Operations:** Allows bidirectional traversal (both forward and backward).
* **Example:** Useful when you need to traverse in both directions or perform operations where both directions are needed.

**Analysis**

**Time Complexity**

* **Add**: O(n) in the worst case for adding to the end of the list.
* **Search**: O(n) due to linear search through the list.
* **Traverse**: O(n) as it involves iterating through the entire list.
* **Delete**: O(n) in the worst case for searching and removing the node.

**Advantages of Linked Lists Over Arrays:**

* **Dynamic Size**: Linked lists can grow or shrink in size dynamically, unlike arrays which have a fixed size.
* **Efficient Insertions/Deletions**: Inserting or deleting elements does not require shifting elements, as in arrays.

**When to Use Linked Lists:**

* **Dynamic Data Size**: When the number of elements changes frequently.
* **Frequent Insertions/Deletions**: When operations involving frequent insertions or deletions are required.