**Singly Linked List:**

* **Structure:** A singly linked list consists of nodes where each node contains data and a reference (or pointer) to the next node in the sequence.
* **Characteristics:**
  + **Unidirectional:** Traversal is possible in one direction (forward) only.
  + **Simple Implementation:** Easier to implement and requires less memory than doubly linked lists.

**Doubly Linked List:**

* **Structure:** A doubly linked list consists of nodes where each node contains data, a reference to the next node, and a reference to the previous node.
* **Characteristics:**
  + **Bidirectional:** Traversal is possible in both directions (forward and backward).
  + **Additional Memory:** Requires more memory to store the extra previous pointer.

**Advantages of Linked Lists Over Arrays for Dynamic Data:**

1. **Dynamic Size:** Linked lists can grow and shrink in size dynamically, making them more flexible for data structures where the number of elements can vary.
2. **Efficient Insertions/Deletions:** Insertions and deletions in linked lists are more efficient than arrays, especially when dealing with large datasets, since there's no need to shift elements.
3. **Memory Utilization:** Linked lists can utilize memory more efficiently as they don't require a contiguous block of memory, unlike arrays.
4. **No Need for Preallocation:** Linked lists do not require preallocating memory, which can save space and avoid memory wastage.

**Disadvantages:**

1. **Memory Overhead:** Linked lists require additional memory for storing pointers, which can be significant compared to arrays.
2. **Sequential Access:** Linked lists do not support direct access to elements; elements must be accessed sequentially from the head.
3. **Cache Locality:** Due to non-contiguous memory allocation, linked lists are less cache-friendly than arrays, leading to potentially slower access times.