Linked List

Linked Lists

What is a Linked List?

A linear data structure where each element is a separate object.

Each element (node) contains:

- Data
- A reference (or link) to the next node in the sequence.

Linked Lists

- Types of Linked Lists:
- **Singly Linked List:** Each node points to the next node.
- **Doubly Linked List:** Each node points to both the next and previous nodes.
- **Circular Linked List:** The last node points back to the first node.

Linked Lists

Benefits:

- Dynamic size
- Efficient insertions and deletions

Drawbacks:

- Extra memory for pointers
- Sequential access, no direct access to elements

Insert at End of Linked List

Objective:

Add a new node to the end of the linked list.

Steps:

- **Create a new node** with the data to be inserted.
- **Traverse the linked list** to find the last node.
- **Update the last node's next pointer** to point to
- the new node.
- **Set the new node's next pointer** to null (end of the list).

Insert at End of Linked List

1. Create a new node with the given data.
2. If the list is empty, set head to the new node.
3. Else, traverse the list to find the last node.
4. Set the last node's next to the new node.
5. Set the new node's next to null.

Display of Linked List

Objective:

Print all nodes in the linked list.

Steps:

- **Start from the head** of the linked list.
- **Traverse through each node,** printing the data.
- **Move to the next node** until you reach the end (null).

```
1. Initialize a current node as head.
2. While the current node is not null:
    - Print the data of the current node.
    - Move to the next node.
3. End when the current node is null.
```

Insert at Start of Linked List

Objective:

Add a new node at the beginning of the linked list.

Steps:

- **Create a new node** with the data to be inserted.
- **Set the new node's next pointer** to point to the current head.
- **Update the head** to point to the new node.

```
1. Create a new node with the given data.
2. Set the new node's next to head.
3. Update head to the new node.
```

Insert After a Specific Value

Objective:

Insert a new node after a node with a specific value.

Steps:

- **Search for the node** with the specific value.
- **Create a new node** with the data to be inserted.
- **Update the new node's next pointer** to point to the current node's next.
- **Update the current node's next pointer** to point to the new node.

```
1. Search for the node with the given value.
2. If found, create a new node with the data to be inserted.
3. Set the new node's next to the current node's next.
4. Update the current node's next to the new node.
5. If not found, handle the case (e.g., show an error).
```