Linked List Operation

Insert Before a Specific Value

Objective:

Add a new node before a node containing a specific value.

Steps:

- **Create a new node** with the data to be inserted.
- **Check if the list is empty** or if the specific value is at the head.
- **Traverse the list** to find the node with the specific value and keep track of the previous node.
- **Update the previous node's next pointer** to point to the new node.
- **Set the new node's next pointer** to point to the current node (the one with the specific value).

Insert Before a Specific Value

1. Create a new node with the given data.
2. If the list is empty, show an error or handle the case.
3. If the head contains the specific value:
   - Insert at the start.
4. Traverse the list to find the node with the specific value.
   - Keep track of the previous node.
5. Update the previous node's next to the new node.
6. Set the new node's next to the current node.

Delete from Start

Objective:

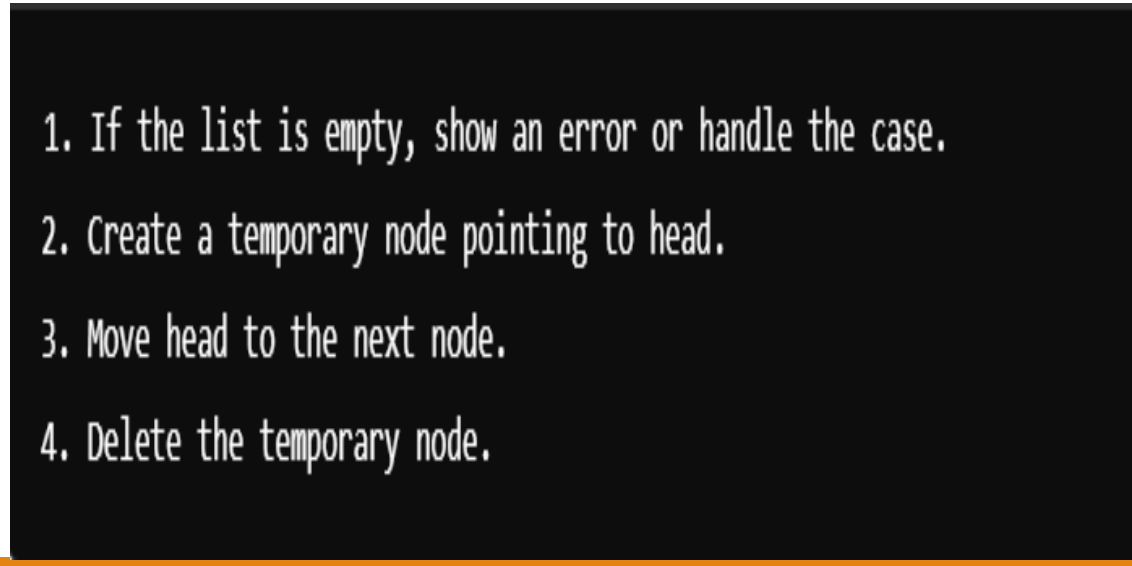
Remove the first node from the list.

Steps:

Check if the list is empty. If it is, there's nothing to delete.

Update the head pointer to point to the second node.

Delete the original first node to free memory.



Delete from End

Objective:

Remove the last node from the list.

Steps:

Check if the list is empty. If it is, there's nothing to delete.

Traverse the list to find the second-to-last node.

Update the second-to-last node's next pointer to NULL.

Delete the last node to free memory.

```
1. If the list is empty, show an error or handle the case.
2. If there's only one node, delete it and set head to NULL.
3. Traverse to find the second-to-last node.
4. Delete the last node.
5. Set the second-to-last node's next to NULL.
```

Delete After a Specific Value

Objective:

Remove the node immediately after a node containing a specific value.

Steps:

Traverse the list to find the node with the specific value.

Check if the node's next pointer is not NULL (if there's a node to delete).

Update the next pointer of the current node to skip over the node to be deleted.

Delete the node after the specific value to free memory.

```
1. Traverse the list to find the node with the specific value.
2. If the next node is NULL, there's nothing to delete.
3. Set a temporary node to the node after the specific value.
4. Update the current node's next to skip the node to delete.
5. Delete the temporary node.
```

Delete Before a Specific Value

Objective:

Remove the node immediately before a node containing a specific value.

Steps:

Check if the list has at least two nodes. If not, deletion is not possible.

Traverse the list to find the node before the specific value.

Keep track of the previous node and the node to be deleted.

Update the next pointer of the node before the node to be deleted.

Delete the node before the specific value to free memory.

```
1. If the list is empty or has only one node, handle the case.
2. If the head is immediately before the specific value, delete from the start.
3. Traverse the list to find the node before the specific value.
4. Keep track of the previous node and the node to delete.
5. Update the previous node's next to skip the node to delete.
6. Delete the node to be removed.
```