Single Linked List

Write a Python program to implement deletion operations in a single linked list.
Note:

- **Delete an element in the beginning**

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

class LinkedList:
    def __init__(self):
        self.head = None

    def insert(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
        else:
            current = self.head
            while current.next:
                current = current.next
            current.next = new_node

    def display(self):
        current = self.head
        if not current:
            print("List is empty")
        else:
            while current:
                print(current.data, end=" -> ")
                current = current.next
            print("None")

    def delete_beginning(self):
        if not self.head:
            print("List is empty, nothing to delete.")
        else:
            print(f"Deleting {self.head.data} from the beginning.")
            self.head = self.head.next

ll = LinkedList()
ll.insert(10)
ll.insert(20)
ll.insert(30)
print("Initial List:")
ll.display()

ll.delete_beginning()
```

```
print("After Deletion from Beginning:")
ll.display()
```

Output

```
Initial List:
10 -> 20 -> 30 -> None
Deleting 10 from the beginning.
After Deletion from Beginning:
20 -> 30 -> None
```

- **Delete an element in the end**

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

class LinkedList:
    def __init__(self):
        self.head = None

    # Insert a new node at the end of the list
    def insert(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
        else:
            current = self.head
            while current.next:
                current = current.next
            current.next = new_node

    def display(self):
        current = self.head
        if not current:
            print("List is empty")
        else:
            while current:
                print(current.data, end=" -> ")
                current = current.next
            print("None")

    def delete_end(self):
        if not self.head:
            print("List is empty, nothing to delete.")
        elif not self.head.next:
            print(f"Deleting {self.head.data} from the end.")
            self.head = None
```

```
        else:
            current = self.head
            while current.next.next:
                current = current.next
            print(f"Deleting {current.next.data} from the end.")
            current.next = None

ll = LinkedList()
ll.insert(10)
ll.insert(20)
ll.insert(30)
print("Initial List:")
ll.display()

ll.delete_end()
print("After Deletion from End:")
ll.display()
```

Output:

```
Initial List:
10 -> 20 -> 30 -> None
Deleting 30 from the end.
After Deletion from End:
10 -> 20 -> None
```

- **Delete an element before an element**

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

class LinkedList:
    def __init__(self):
        self.head = None

    def insert(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
        else:
            current = self.head
            while current.next:
                current = current.next
            current.next = new_node


    def display(self):
```

```
        current = self.head
        if not current:
            print("List is empty")
        else:
            while current:
                print(current.data, end=" -> ")
                current = current.next
            print("None")

    def delete_before(self, target):
        if not self.head or not self.head.next:
            print("List is too short, nothing to delete before the target.")
            return

        if self.head.next.data == target:
            print(f"Deleting {self.head.data} before {target}.")
            self.head = self.head.next
            return

        prev = None
        current = self.head
        while current.next and current.next.data != target:
            prev = current
            current = current.next

        if current.next is None:
            print(f"{target} not found in the list.")
        else:
            print(f"Deleting {current.data} before {target}.")
            prev.next = current.next

ll = LinkedList()
ll.insert(10)
ll.insert(20)
ll.insert(30)
ll.insert(40)

print("Initial List:")
ll.display()

ll.delete_before(40)
print("After Deletion Before 40:")
ll.display()
```

Output:

```
Initial List:
10 -> 20 -> 30 -> 40 -> None
Deleting 30 before 40.
After Deletion Before 40:
```

```
10 -> 20 -> 40 -> None
```