Singly and Doubly Linked Lists

Singly Linked List

Code:

```java
class SinglyLinkedList {
    class Node {
        int data;
        Node next;

        Node(int data) {
            this.data = data;
        }
    }

    Node head;

    // Insert at beginning
    public void insertAtBeg(int data) {
        Node newNode = new Node(data);
        newNode.next = head;
        head = newNode;
    }
```

```java
    // Insert at end
    public void insertAtEnd(int data) {
        Node newNode = new Node(data);
        if (head == null) {
            head = newNode;
            return;
        }
        Node temp = head;
        while (temp.next != null)
            temp = temp.next;
        temp.next = newNode;
    }
```

```java
    // Insert at specific position
    public void insertAtPos(int data, int pos) {
        if (pos == 0) {
            insertAtBeg(data);
            return;
        }
        Node newNode = new Node(data);
        Node temp = head;
        for (int i = 0; i < pos - 1 && temp != null; i++)
            temp = temp.next;
        if (temp == null) {
            System.out.println("Position out of bounds.");
            return;
        }
        newNode.next = temp.next;
        temp.next = newNode;
    }
```

```java
    // Delete at beginning
    public void deleteAtBeg() {
        if (head == null) return;
        head = head.next;
    }

    // Delete at end
    public void deleteAtEnd() {
        if (head == null) return;
        if (head.next == null) {
            head = null;
            return;
        }
        Node temp = head;
        while (temp.next.next != null)
            temp = temp.next;
        temp.next = null;
    }
```

```java
// Delete at specific position
public void deleteAtPos(int pos) {
    if (head == null) return;
    if (pos == 0) {
        head = head.next;
        return;
    }
    Node temp = head;
    for (int i = 0; i < pos - 1 && temp.next != null; i++)
        temp = temp.next;
    if (temp.next == null) {
        System.out.println("Position out of bounds.");
        return;
    }
    temp.next = temp.next.next;
}
```

```
    // Display list
    public void display() {
        Node temp = head;
        while (temp != null) {
            System.out.print(temp.data + " -> ");
            temp = temp.next;
        }
        System.out.println("null");
    }

    public static void main(String[] args) {
        SinglyLinkedList list = new SinglyLinkedList();
        list.insertAtEnd(10);
        list.insertAtBeg(5);
        list.insertAtPos(7, 1);
        list.display();
```

```
        list.deleteAtBeg();
        list.display();

        list.deleteAtEnd();
        list.display();

        list.deleteAtPos(0);
        list.display();
    }
```

Output:

Output

```
5 -> 7 -> 10 -> null
7 -> 10 -> null
7 -> null
null
```