```java
package Doubly_Linked_List;


public class DoublyLinkedList {
    private Node head = null;
    private int length = 0;

    public static void main(String[] args) {
        DoublyLinkedList list = new DoublyLinkedList();
        list.insert(8);
        list.insert(9);
        list.insert(10);
        list.insertTail(7);
        list.insert(100, 3);
        list.print();
        list.remove();
        list.removeTail();
        list.remove(1);
        list.print();
        System.out.println(list.isEmpty());
        System.out.println("Length = "+list.length());
    }

    public void insert(int data) {
        Node node = new Node();
        node.data = data;
        node.prev = null;
        node.next = null;
        if (length == 0) {
            head = node;
        } else {
            node.next = head;
            head.prev = node;
            head = node;
        } length++;
    }

    public void insertTail(int data) {
        Node node = new Node();
        node.data = data;
        node.prev = null;
        node.next = null;
        if (length == 0) {
            head = node;
            node.next = null;
        } else {
            Node nodeTemp = new Node();
            nodeTemp = head;
            while (nodeTemp.next != null) {
                nodeTemp = nodeTemp.next;
            }
            nodeTemp.next = node;
            node.prev = nodeTemp;
            node.next = null;
        }
        length++;

    }

    public void insert(int data, int Position) {
        if (Position < 0 || Position > length) {
        throw new IndexOutOfBoundsException("out of bounds!");
    }

        Node node = new Node();
        node.data = data;
        node.prev = null;
        node.next = null;
```

```
        if (length == 0) {
            head = node;
            node.next = null;
        } else {
            Node nodeTemp = new Node();
            nodeTemp = head;
            for (int i = 0; i < Position; i++) {
                nodeTemp = nodeTemp.next;
            }
            nodeTemp.next = node;
            node.prev = nodeTemp;
        }
        length++;
    }


    public void print() {
        Node current = head;
        for (int i = 0; i < length; i++) {
            if (i == length - 1) {
            System.out.println(current.data); return;
        }
            System.out.print(current.data); System.out.print("->");
            current = current.next;
        }
    }

    public void remove() { if (length == 0) {
        return;
    }
        if (length == 1) {
            head = null;
        } else {
            head = head.next; head.prev = null;
        }
        length--;
    }

    public void removeTail() {
        if (length == 0) {
        return;
        }
        if (length == 1) {
            head = null;
        } else {
            Node current = head;
            while (current.next != null) {
                current = current.next;
            }
            current.prev.next = null;
        }
        length--;
    }



    public void remove(int position) {
        if (position < 0 || position >= length) {
        throw new IndexOutOfBoundsException("out of bounds!");
        }
        if (position == 0) {
            remove(); return;
        }
        if (position == length - 1) {
            removeTail(); return;
        }
        Node current = head;
```

```
        for (int i = 0; i < position; i++) {
            current = current.next;
        }
        current.prev.next = current.next;
        current.next.prev = current.prev;
        length--;
    }

    public int length() {
        return length;
    }

    public boolean isEmpty() {
        return length == 0;
    }

    class Node {
        Node prev; int data; Node next;

        public Node() {
        }

        public Node(int data) {
            this.data = data;
            this.prev = null;
            this.next = null;
        }
    }

}
```