```
package doublelinkedlist;

public class DoubleLinkedList {

    class Node {

        private int value;
        private Node next;
        private Node previous;

        public Node(int value) {
            this.value = value;
        }
    }
    private Node head;
    private Node tail;
    private int size;

    public void addFirst(int item) {
        // Creating newNode memory & assigning data value

        var newnode = new Node(item);
        newnode.next = head;
        newnode.previous = null;
        // if DLL had already >=1 nodes

        if (head != null) {
            head.previous = newnode;
        }
        // changing head to this

        head = newnode;
        tail = newnode;

        size++;
    }

    public void addLast(int item) {
        var newnode = new Node(item);
        newnode.next = null;

        // assign data
        // since this will be the last node its next will be NULL
        newnode.next = null;

        //if we are entering the first node
        if (head == null) {
            head = newnode;
            newnode.previous = null;
            return;
        }

        Node last = head;

        // traverse to the current last node
        while (last.next != null) {
```

```
            last = last.next;
        }

        // assign current last node's next to this new node
        // assign new node's previous to this last node
        last.next = newnode;
        newnode.previous = last;
        // new_node becomes the last node now

        size++;
    }

    private boolean isEmpty() {
        return head == null;
    }

    public void show() {
        Node temp;
        temp = head;
        Node temp2 = null;
        System.out.print("In Forward: \n");

        System.out.print("null<-->");
        while (temp != null) {
            System.out.print(temp.value + "<-->");
            temp2 = temp;

            temp = temp.next;
        }
        System.out.print("null");
        System.out.println("\n");
        System.out.print("In backward: \n");
        System.out.print("null<-->");

        while (temp2 != null) {
            System.out.print(temp2.value + "<-->");
            temp2 = temp2.previous;
        }
        System.out.print("null");
        System.out.println("\n");

    }

    public void reverse() {
        Node temp = tail;
        while (temp != null) {
            System.out.print(temp.value + " ");
            temp = temp.previous;
        }
        System.out.println();

    }

    public static void main(String[] args) {
        var dll = new DoubleLinkedList();
        dll.addFirst(1);
        dll.addFirst(2);
```

```
        dll.addFirst(3);
        dll.show();
        dll.addLast(4);
        dll.addLast(5);
        dll.addLast(6);
        dll.show();

    }

}
```