

Credit
3110
Assignment
DoublyLinkedList

How has your program changed from planning to coding to now? Please explain?

```
    }

    public static void main(String[] args)
    {
        DoublyLinkedList list = new DoublyLinkedList();

        list.addAtFront("Raj");
        list.addAtEnd("Jeni");
        list.addAtEnd("Yuri");

        System.out.println("display forward");
        list.displayList();

        System.out.println("display reverse");
        list.displayReverseList();

        System.out.println("remove jeni");
        list.remove("Jeni");

        System.out.println("display forward");
        list.displayList();

        System.out.println("display reverse");
        list.displayReverseList();
    }
}

public class DoublyLinkedList
{
    private Node head;
    private Node tail;

    public DoublyLinkedList()
    {
        head = null;
        tail = null;
    }

    // adds node to front
    public void addAtFront(String value)
    {
        Node newNode = new Node(value);

        if (head == null)
        {
            head = newNode;
            tail = newNode;
        }
        else
        {
            newNode.next = head;
            head.prev = newNode;
            head = newNode;
        }
    }

    // adds node to end
    public void addAtEnd(String value)
    {
        Node newNode = new Node(value);

        if (tail == null)
        {
            head = newNode;
            tail = newNode;
        }
    }
}
```

```

46     Node newNode = new Node(value);
47
48     if (tail == null)
49     {
50         head = newNode;
51         tail = newNode;
52     }
53     else
54     {
55         tail.next = newNode;
56         newNode.prev = tail;
57         tail = newNode;
58     }
59 }
60
61 // removes first matching value
62 public void remove(String value)
{
63     Node current = head;
64
65     while (current != null)
66     {
67         if (current.data.equals(value))
68         {
69             if (current == head)
70             {
71                 head = current.next;
72             }
73             else
74             {
75                 current.prev.next = current.next;
76             }
77
78             if (current == tail)
79             {
80                 tail = current.prev;
81             }
82             else
83             {
84                 current.next.prev = current.prev;
85             }
86         }
87     }
88
89     current = current.next;
90 }
91
92
93 // displays list forward
94 public void displayList()
{
95     Node current = head;
96
97     while (current != null)
98     {
99         System.out.print(current.data + " ");
100        current = current.next;
101    }
102
103    System.out.println();
104 }
105
106
107 // displays list backward
108 public void displayReverseList()
109 {
110     Node current = tail;
111
112     while (current != null)
113     {
114         System.out.print(current.data + " ");
115         current = current.prev;
116     }
117
118    System.out.println();
119 }
120
121
122 // node class
123 private class Node
{
124     private String data;

```

```
L22⊕  private class Node
L23  {
L24      private String data;
L25      private Node next;
L26      private Node prev;
L27
L28⊕  public Node(String d)
L29  {
L30      data = d;
L31      next = null;
L32      prev = null;
L33  }
L34 }
L35 }
```
