

Credit: CSE3110 - Iterative Algorithm 1  
Assignment: StackList

Describe the errors you've encountered while working on this assignment. What caused the error and how do you overcome the error?

Error - Tried to print current but should've printed current.getData

```
60 public String getEnd()  
61 {  
62     Node current = head;  
63     current = current.getNext();  
64     String s = "";  
65  
66     while(current.getNext() != null)  
67     {  
68         current = current.getNext();  
69     }  
70  
71     s += current;  
72  
73     return s; //current.getData();  
74  
75 }  
76  
77 //Part 2
```

Problems @ Javadoc Console X Git Staging

<terminated> LinkedListTesting [Java Application] C:\Program Files\Eclipse\eclipse\plugins\org.e

List has 4 items.  
B  
A  
C  
D

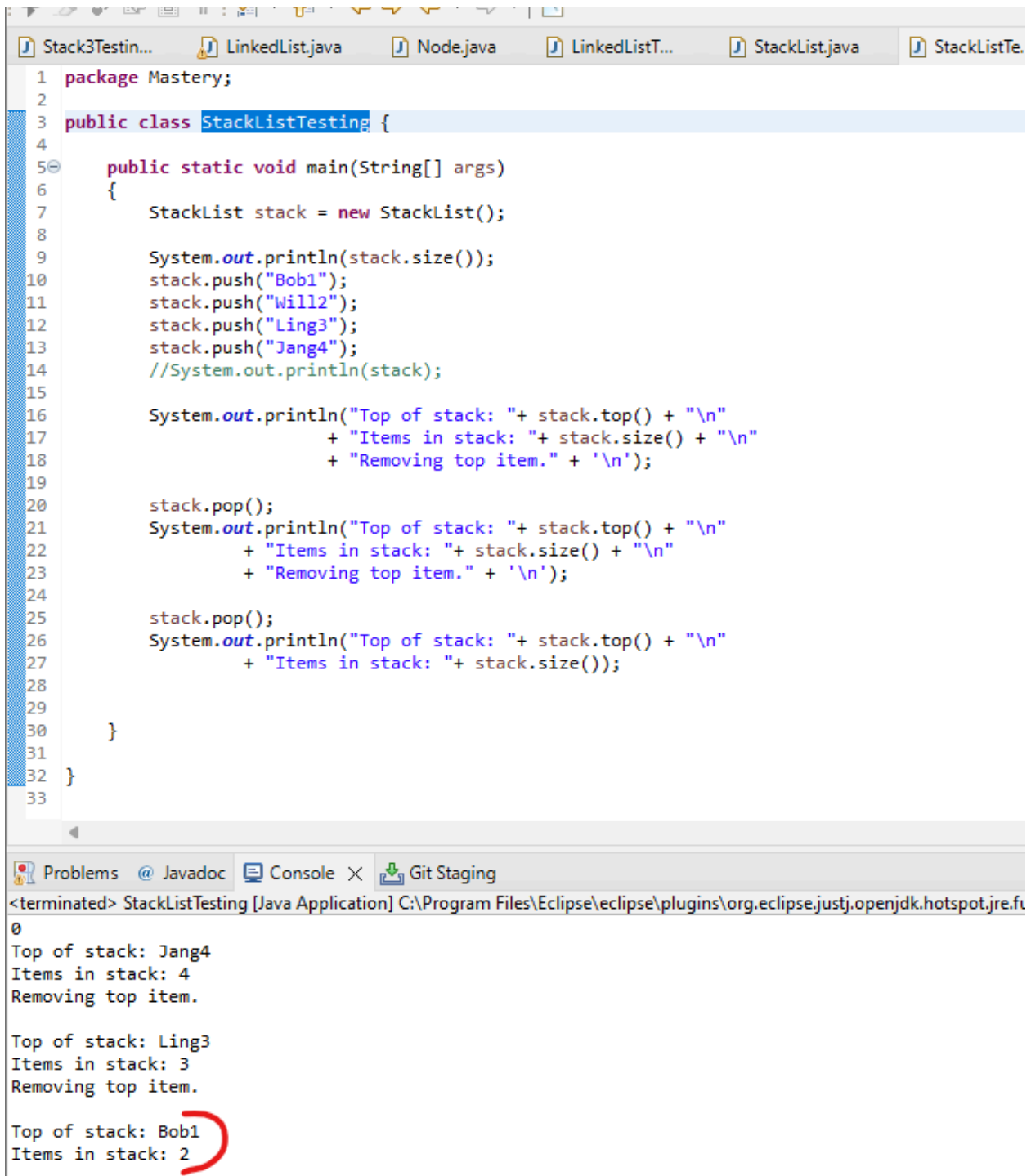
LinkedListParts123\_SkillBuilders.Node@3dd3bcd  
Removing D from list.  
Emptying list.  
There are no items in list.

Solution: Got rid of the String s variable and returned the node.getData();. I didn't need the string variable since in the node class the getData method returns a string.

```
return current.getData();
```

Error - Methods wouldn't work when there were less than 2 or 2 nodes in the list in the stack. This was because the linked list wouldn't be able to add objects to the end if it was empty. The current variable in the linked list method would read the next node as null which would lead to

an error. I had to add and remove the placeholder node depending on the size of the list. Stack and list size were kept separate.



```
1 package Mastery;
2
3 public class StackListTesting {
4
5     public static void main(String[] args)
6     {
7         StackList stack = new StackList();
8
9         System.out.println(stack.size());
10        stack.push("Bob1");
11        stack.push("Will2");
12        stack.push("Ling3");
13        stack.push("Jang4");
14        //System.out.println(stack);
15
16        System.out.println("Top of stack: " + stack.top() + "\n"
17                           + "Items in stack: " + stack.size() + "\n"
18                           + "Removing top item." + '\n');
19
20        stack.pop();
21        System.out.println("Top of stack: " + stack.top() + "\n"
22                           + "Items in stack: " + stack.size() + "\n"
23                           + "Removing top item." + '\n');
24
25        stack.pop();
26        System.out.println("Top of stack: " + stack.top() + "\n"
27                           + "Items in stack: " + stack.size());
28
29    }
30 }
31
32
33
```

<terminated> StackListTesting [Java Application] C:\Program Files\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.f

0  
Top of stack: Jang4  
Items in stack: 4  
Removing top item.

Top of stack: Ling3  
Items in stack: 3  
Removing top item.

Top of stack: Bob1  
Items in stack: 2

```
75
76 public void addAtEnd(String str)
77 {
78     Node newNode = new Node(str);
79     Node current = head;
80
81     while(current.getNext() != null)
82     {
83         current = current.getNext();
84     }
85
86     current.setNext(newNode);
87
88 }
89
90 //Part 3
91
92 public void makeEmpty()
93 {
94     String[] words = toString().split(" " + '\n');
95
96     for (String s: words)
97     {
98         remove(s);
99     }
100 }
101
102 //Part 4: Stack to Stack()
103
```

Outline

- LinkedListParts
  - LinkedList
    - head : Node
    - LinkedList()
    - addAtFront()
    - remove(String)
    - size() : int
    - getEnd() : String
    - addAtEnd(String)
    - makeEmpty()
    - toString() : String

Problems @ Javadoc Console X Git Staging

<terminated> StackListTesting [Java Application] C:\Program Files\Eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_21.0.3.v20240426-1530\jre\bin\javaw.exe (Nov 20, 2024)

Exception in thread "main" java.lang.NullPointerException: Cannot invoke "LinkedListParts123\_SkillBuilders.Node.getNext()" because "current" is null

at Chapter13/LinkedListParts123\_SkillBuilders.LinkedList.addAtEnd(LinkedList.java:81)

at Chapter13/Mastery.StackList.push(StackList.java:39)

at Chapter13/Mastery.StackListTesting.main(StackListTesting.java:10)

Solution: Added a placeholder node to add and remove depending on the list size. In the constructor, the stack is displayed to the user as empty but the list has a placeholder node so that methods won't run into a "current node = null" error.

In the pop method, if the list size is currently 2 or less, the top item (which was the front of the list, now is the back) is recorded, then the placeholder variable is added to the front of the list / back of the stack, and the front of the stack / back of the list is removed.

For the push method, if the list size is 1 or smaller, then the placeholder variable is removed since now the list and stack will be bigger than 1 and I don't have to worry about the current node = null error.

```

// Constructor
public StackList()
{
    list = new LinkedList();
    top = 0; // Stack size
    list.addAtFront("placeholder"); // Placeholder so that the stack doesn't run into a node = null therefore can't run error
}

// Remove top of Stack
public Object pop()
{
    Object topItem; // Object variable to keep track of the
    topItem = list.getEnd();

    top--; // Decrease size by one

    // Has to be less than or equal to 2 because if not then
    // Have to use placeholder to prevent the node from being null
    if(list.size() <= 2)
    {
        topItem = list.getFront(); // The front of the list
        list.addAtFront("placeholder"); // Prevent null node
        list.remove(list.getEnd()); // Remove the end which was the top item
    }

    return topItem;
}

// Push objects to stack
public void push(Object item)
{
    // Linked list methods are made for strings
    String s = "" + item; // Empty string
    top++; // Increase size

    // Same placeholder reasoning, holds the top of the stack
    // Placeholder variable is the only one that can be null
    if(list.size() <= 1)
    {
        list.addAtEnd(s);
        list.remove("placeholder");
    }
    // If the stack is bigger than one then
    else
    {
        list.addAtEnd(s);
    }
}

```