

ait-tr / cohort44E Publicgenerated from [ait-tr/cohort-xx.x](#)[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)[cohort44E](#) / [basic_programming](#) / [lesson_34](#) / [code](#) / [demo20240529_GenericLinkedList](#) / [test](#)
/ [MyLinkedListTest.java](#)

...



Andy179176 20240529

461fab · 9 hours ago



158 lines (135 loc) · 4.38 KB

Code

Blame

Raw



```
1  import org.junit.jupiter.api.Assertions;
2  import org.junit.jupiter.api.DisplayName;
3  import org.junit.jupiter.api.Test;
4
5  import static org.junit.jupiter.api.Assertions.*;
6
7  class MyLinkedListTest {
8
9      @Test
10     void add() {
11         MyList<String> list = new MyLinkedList<>();
12         list.add("Jack");
13         list.add("John");
14         list.add("Nick");
15
16         Assertions.assertEquals(3, list.size());
17         Assertions.assertEquals("Jack", list.get(0));
18         Assertions.assertEquals("John", list.get(1));
19         Assertions.assertEquals("Nick", list.get(2));
20     }
21     @Test
22     void addInteger() {
23         MyList<Integer> list = new MyLinkedList<>();
24         list.add(1);
25         list.add(3);
26         list.add(5);
27
28         Assertions.assertEquals(3, list.size());
29         Assertions.assertEquals(1, list.get(0));
30         Assertions.assertEquals(3, list.get(1));
31         Assertions.assertEquals(5, list.get(2));
32     }
33
34     @Test
35     @DisplayName("add after removing")
36     void add1() {
```

```
37     MyList<String> list = new MyLinkedList<>();
38     list.add("Jack");
39     list.add("John");
40     list.add("Nick");
41     list.remove(2);
42     list.add("Ann");
43
44     Assertions.assertEquals(3, list.size());
45     Assertions.assertEquals("Jack", list.get(0));
46     Assertions.assertEquals("John", list.get(1));
47     Assertions.assertEquals("Ann", list.get(2));
48 }
49
50 @Test
51 void get() {
52     MyList<String> list = new MyLinkedList<>();
53     list.add("Jack");
54     list.add("John");
55     list.add("Nick");
56
57     Assertions.assertEquals("John", list.get(1));
58 }
59
60 @Test
61 void size() {
62     MyList<String> list = new MyLinkedList<>();
63     list.add("Jack");
64     list.add("John");
65     list.add("Nick");
66     Assertions.assertEquals(3, list.size());
67 }
68
69 @Test
70 @DisplayName("size should be 0 if list is empty")
71 void size1() {
72     MyList<String> list = new MyLinkedList<>();
73     Assertions.assertEquals(0, list.size());
74 }
75
76
77 @Test
78 @DisplayName("regular remove")
79 void remove() {
80     MyList<String> list = new MyLinkedList<>();
81     list.add("Jack");
82     list.add("John");
83     list.add("Nick");
84     list.remove(1);
85
86     Assertions.assertEquals(2, list.size());
87     Assertions.assertEquals("Jack", list.get(0));
88     Assertions.assertEquals("Nick", list.get(1));
89 }
90
```

```
91     @Test
92     @DisplayName("remove tail element")
93     void remove1() {
94         MyList<String> list = new MyLinkedList<>();
95         list.add("Jack");
96         list.add("John");
97         list.add("Nick");
98         list.remove(2);
99
100        Assertions.assertEquals(2, list.size());
101        Assertions.assertEquals("Jack", list.get(0));
102        Assertions.assertEquals("John", list.get(1));
103    }
104
105    @Test
106    @DisplayName("remove head element")
107    void remove2() {
108        MyList<String> list = new MyLinkedList<>();
109        list.add("Jack");
110        list.add("John");
111        list.add("Nick");
112        list.remove(0);
113
114        Assertions.assertEquals(2, list.size());
115        Assertions.assertEquals("John", list.get(0));
116        Assertions.assertEquals("Nick", list.get(1));
117    }
118
119    @Test
120    @DisplayName("remove() last element")
121    void remove3() {
122        MyList<String> list = new MyLinkedList<>();
123        list.add("Jack");
124        list.add("John");
125        list.add("Nick");
126        String removed = list.remove();
127        boolean isSizeCorrect = list.size()==2;
128        boolean isValueCorrect = removed.equals("Nick");
129        removed = list.remove();
130        isSizeCorrect = isSizeCorrect? list.size()==1:false;
131        isValueCorrect = isValueCorrect? removed.equals("John"):false;
132        removed = list.remove();
133        isSizeCorrect = isSizeCorrect?list.size()==0:false;
134        isValueCorrect = isValueCorrect? removed.equals("Jack"):false;
135
136        Assertions.assertTrue(isValueCorrect);
137        Assertions.assertTrue(isSizeCorrect);
138        Assertions.assertNull(list.get(0));
139    }
140
141    @Test
142    @DisplayName("remove() last element from single element list")
143    void remove4() {
144        MyList<String> list = new MyLinkedList<>();
```

```
145         list.add("Jack");
146         String removedString = list.remove();
147
148         Assertions.assertAll(
149             ()->Assertions.assertEquals(0, list.size()),
150             ()->Assertions.assertEquals("Jack", removedString),
151             ()->Assertions.assertNull(list.get(0))
152         );
153     }
154
155     @Test
156     void set() {
157     }
158 }
```