```
 1 package Project4;
 2
 3 import org.junit.Before;
 4 import org.junit.Test;
 5
 6 import java.text.ParseException;
 7 import java.text.SimpleDateFormat;
 8 import java.util.Date;
 9 import java.util.GregorianCalendar;
10 import java.util.Random;
11
12 import static org.junit.Assert.*;
13
14 public class MySingleLinkedListTest {
15
16     private Car car1;
17     private Car car2;
18     private Car car3;
19     private Car car4;
20     private Car car5;
21     private Car car6;
22     private Truck truck1;
23     private Truck truck2;
24     private Truck truck3;
25     private Truck truck4;
26     private Truck truck5;
27     private Truck truck6;
28     private MySingleLinkedList list = new MySingleLinkedList();
29
30     @Before
31     public void createList() {
32
33         SimpleDateFormat df = new SimpleDateFormat("MM/dd/yyyy");
34         GregorianCalendar temp1 = new GregorianCalendar();
35         GregorianCalendar temp2 = new GregorianCalendar();
36         GregorianCalendar temp3 = new GregorianCalendar();
37         GregorianCalendar temp4 = new GregorianCalendar();
38         GregorianCalendar temp5 = new GregorianCalendar();
39         GregorianCalendar temp6 = new GregorianCalendar();
40
41         try {
42             Date d1 = df.parse("1/20/2010");
43             temp1.setTime(d1);
44             Date d2 = df.parse("12/20/2018");
45             temp2.setTime(d2);
46             Date d3 = df.parse("1/20/2019");
47             temp3.setTime(d3);
48             Date d4 = df.parse("3/20/2019");
49             temp4.setTime(d4);
50             Date d5 = df.parse("4/20/2019");
51             temp5.setTime(d5);
52             Date d6 = df.parse("1/20/2020");
53             temp6.setTime(d6);
54
55
56             car1 = new Car (temp1, "Outback", 11000,"LX", false);
```

```
 57             car2 = new Car (temp2, "Chevy", 12000,"EX", false);
 58             car3 = new Car (temp3, "Focus", 13000,"EX", true);
 59             car4 = new Car (temp4, "Outback2", 14000,"EX", false);
 60             car5 = new Car (temp5, "Chevy2", 15000,"LX", true);
 61             car6 = new Car (temp6, "Focus2", 16000,"LX", true);
 62             truck1 = new Truck(temp1,"F150",11000,"EX",false);
 63             truck2 = new Truck(temp2,"F250",12000,"LX",true);
 64             truck3 = new Truck(temp3,"F350",13000,"EX",false);
 65             truck4 = new Truck(temp4,"F450",14000,"EX",true);
 66             truck5 = new Truck(temp5,"F550",15000,"LX",false);
 67             truck6 = new Truck(temp6,"F650",16000,"EX",true);
 68
 69             list.add(car1);
 70             list.add(car2);
 71             list.add(car3);
 72             list.add(car4);
 73             list.add(car5);
 74             list.add(car6);
 75             list.add(truck1);
 76             list.add(truck2);
 77             list.add(truck3);
 78             list.add(truck4);
 79             list.add(truck5);
 80             list.add(truck6);
 81
 82         } catch (ParseException e) {
 83             throw new RuntimeException("Error in testing, creation of list");
 84         }
 85     }
 86
 87     @Test
 88     // here is the very small example to use.
 89     public void size() {
 90         assertEquals(12, list.size());
 91         createList();
 92         assertEquals(24, list.size());
 93         list.remove(0);
 94         assertEquals(23,list.size());
 95         list.remove(10);
 96         assertEquals(22, list.size());
 97         list.remove(9);
 98         assertEquals(21, list.size());
 99         list.remove(0);
100         assertEquals(20, list.size());
101         list.remove(0);
102         assertEquals(19, list.size());
103         list.remove(3);
104         list.remove(4);
105         list.remove(1);
106         list.remove(1);
107         for (int i = 0; i < 15; i++)
108             list.remove(0);
109         assertEquals(0, list.size());
110     }
111
112     @Test
```

```
113     public void clear() {
114         list.clear();
115         assertEquals(0, list.size());
116         assertNull(list.get(0));
117         assertNull(list.remove(0));
118     }
119 
120     @Test
121     public void add1() {
122         list.clear();
123         list.add(truck3);
124         list.add(truck5);
125         list.add(truck4);
126         list.add(truck2);
127         list.add(car2);
128         list.add(truck1);
129         list.add(car1);
130         list.add(car3);
131         list.add(car5);
132         list.add(truck6);
133         list.add(car6);
134         list.add(car4);
135         assertEquals(12, list.size());
136         get3();
137     }
138 
139     @Test
140     public void add2() {
141         list.clear();
142         list.add(car3);
143         list.add(car5);
144         list.add(car4);
145         list.add(car2);
146         list.add(truck2);
147         list.add(car1);
148         list.add(truck1);
149         list.add(truck3);
150         list.add(truck5);
151         list.add(car6);
152         list.add(truck6);
153         list.add(truck4);
154         assertEquals(12, list.size());
155         get3();
156     }
157 
158     @Test
159     public void add3() {
160         list.clear();
161         list.add(truck6);
162         list.add(truck5);
163         list.add(truck4);
164         list.add(truck3);
165         list.add(truck2);
166         list.add(truck1);
167         list.add(car6);
168         list.add(car5);
```

```
169         list.add(car4);
170         list.add(car3);
171         list.add(car2);
172         list.add(car1);
173         assertEquals(12, list.size());
174         get3();
175     }
176 
177     @Test (expected = IllegalArgumentException.class)
178     public void get1() {
179         list.get(-1);
180     }
181 
182     @Test (expected = IllegalArgumentException.class)
183     public void get2() {
184         list.get(12);
185     }
186 
187     @Test
188     public void get3() {
189         assertEquals(car1, list.get(0));
190         assertEquals(car2, list.get(1));
191         assertEquals(car3, list.get(2));
192         assertEquals(car4, list.get(3));
193         assertEquals(car5, list.get(4));
194         assertEquals(car6, list.get(5));
195         assertEquals(truck1, list.get(6));
196         assertEquals(truck2, list.get(7));
197         assertEquals(truck3, list.get(8));
198         assertEquals(truck4, list.get(9));
199         assertEquals(truck5, list.get(10));
200         assertEquals(truck6, list.get(11));
201     }
202 
203 
204     @Test (expected = IllegalArgumentException.class)
205     public void remove1() {
206         list.remove(-1);
207     }
208 
209     @Test (expected = IllegalArgumentException.class)
210     public void remove2() {
211         list.remove(12);
212     }
213 
214     @Test
215     public void remove3() {
216         list.remove(11);
217         list.remove(6);
218         list.remove(5);
219         list.remove(0);
220         assertEquals(8, list.size());
221         assertEquals(car2, list.get(0));
222         assertEquals(car5, list.get(3));
223         assertEquals(truck2, list.get(4));
224         assertEquals(truck5, list.get(7));
```

```
225
226         list.remove(1);
227         list.remove(4);
228         assertEquals(6, list.size());
229         assertEquals(car2, list.get(0));
230         assertEquals(car4, list.get(1));
231         assertEquals(truck2, list.get(3));
232         assertEquals(truck4, list.get(4));
233
234         list.remove(0);
235         list.remove(1);
236         list.remove(1);
237         list.remove(1);
238         list.remove(1);
239         list.remove(0);
240
241         assertNull(list.get(0));
242         assertEquals(0, list.size());
243     }
244
245     @Test
246     public void remove4() {
247         createList();
248         Random rand = new Random();
249         for (int rounds = 0; rounds < 10; rounds++) {
250             for (int i = 0; i < 24; i++) {
251                 list.remove(rand.nextInt(24 - i));
252                 assertEquals(23 - i, list.size());
253             }
254             createList();
255             createList();
256         }
257     }
258
259     @Test
260     public void remove5() {
261         list.remove(null);
262     }
263
264     @Test
265     public void remove6() {
266         list.remove(new Car());
267     }
268
269     @Test
270     public void remove7() {
271         list.remove(truck6);
272         list.remove(truck1);
273         list.remove(car6);
274         list.remove(car1);
275         assertEquals(8, list.size());
276         assertEquals(car2, list.get(0));
277         assertEquals(car5, list.get(3));
278         assertEquals(truck2, list.get(4));
279         assertEquals(truck5, list.get(7));
280
```

```
281         list.remove(car3);
282         list.remove(truck3);
283         assertEquals(6, list.size());
284         assertEquals(car2, list.get(0));
285         assertEquals(car4, list.get(1));
286         assertEquals(truck2, list.get(3));
287         assertEquals(truck4, list.get(4));
288
289         list.remove(car2);
290         list.remove(car5);
291         list.remove(truck2);
292         list.remove(truck4);
293         list.remove(truck5);
294         list.remove(car4);
295
296         assertNull(list.get(0));
297         assertEquals(0, list.size());
298
299         list.remove(car1);
300     }
301
302     @Test
303     public void display() {
304         assertNull(list.toString());
305     }
306 }
```