Project Six

Due: Wednesday 10/23/2013 @ 11:55
Points: 30

Objective:

1. Become familiar with the dynamic allocation of array storage
2. Become familiar with java.util.ArrayList

Description:

Create your ListArray class that is functionally similar to the formal Java ArrayList Utility class.

Requirements:

Your ListArray should contain the following methods and characteristics:

- Should contain a default (no-args) constructor `ListArray()`
- Should contain an alternative (overloaded) constructor `ListArray(short[] anArray)` that allows the user to create an initial ListArray that is equivalent to an array that is passed to the constructor
- Should contain a copy-constructor `ListArray(ListArray anExample)` that will create an ListArray identical to one that is passed to it
- Should contain a `void add(short value)` method
- Should contain a `void add(int index, short value)` method
- Should contain a `short remove(int index)` method
- Should contain `boolean contains(short value)` method
- Should contain a `short get(int index)` method
- Should contain an `int indexOf(short value)` method
- Should contain an `boolean isEmpty()` method
- Should contain a `int size()` method
- Should override the `public String toString()` method inherited from the java.util.AbstractCollection class.
- Should override the `public boolean equals(Object o)` method inherited from the java.util.List class.

Notes:

For detailed description of each of the above methods, refer to the original Java documentation on java.util.ArrayList.

Example main:

```
4    public static void main(String[] args) {
5        ListArray list1 = new ListArray();
6
7        short[] array = { 7, 26, 15, 236, 27, 995 };
8        ListArray list2 = new ListArray(array);
9        System.out.println(list2);              // should print 7 26 15 27 995
10
11       System.out.println(list2.equals(list1)); // should return false
12
13       ListArray list3 = new ListArray(list2);
14       System.out.println(list3.equals(list2)); // should return true
15       System.out.println(list2.equals(list3)); // should return true
16
17       System.out.println(list2.get(2));       // should print 15
18       System.out.println(list3.get(2));       // should print 15
19
20       System.out.println(list3.size());       // should print 6
21
22       if(list1.isEmpty()) {
23           for(int i = 0; i < list2.size(); i++) {
24               list1.add(list2.get(i));
25           }
26       }
27
28       System.out.println(list2.equals(list1)); // should return true
29       System.out.println(list1);  // should print 7 26 15 236 27 995
30       list1.remove(3);
31       System.out.println(list1); // should print 7 26 15 27 995
32
33       list2.remove(2);
34       System.out.println(list2); // should print 7 26 236 27 995
35
36       list3.remove(1);
37       System.out.println(list3); // should print 7 15 236 27 995
38
39       list1.add(list1.indexOf((short)995), (short)-1);
40       System.out.println(list1); // should print 7 26 15 27 -1 995
41
42       if(list1.contains((short)-1))
43           System.out.println("list1 does not contain all positive numbers");
44       else
45           System.out.println("list1 contains all positive numbers");
46   }
```

Example output:

```
Problems  Javadoc  Console
<terminated> Project6 [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (Oct 15, 2013 11:00:26 PM)
7 26 15 236 27 995
false
true
true
15
15
6
true
7 26 15 236 27 995
7 26 15 27 995
7 26 236 27 995
7 15 236 27 995
7 26 15 27 -1 995
list1 does not contain all positive numbers
```

Grading guidelines:

Click here to view the Rubric