Assignment 4

Overview

Using Generics to create a `Backpack` class that will accept a generic type `<E>` and work with those types.

To-do

1. Create pseudocode for `allIndiciesOf()` and `removeMultipleInstancesOf()`.
2. Modify the `Backpack` class to implement `BackpackInterface`.
3. Write a `BackpackTester` class to test all methods in `Backpack`.

Pseudocode

allIndiciesOf()

Although the execution seemed easy on pseudocode, it turned out to be a lot harder than what I have written down, due to primitive lists requiring an arraysize on their initialization. My code is a bit different than what was outline in my pseudocode because of this.

```
Initialize a temporary list.
Check every spot in container and compare it with the item sought.
  if it matches, append that to the temporary list
Return the list once finished.
```

removeMultipleInstancesOf()

```
Check to see if there is only one instance OR no instance of the specified variable.
  return false and exit if either is true.
Run removeAll(), which will remove all instances of a specified value. return true afterwards.
```

Backpack.java

```
1  import java.util.ArrayList;
2
3  public class Backpack<E> implements BackpackInterface<E> {
4
5    protected ArrayList<E>container;
6
7    /* Table of contents for Backpack():
8     * Constructor                  Ln. 19
9     * add()                        Ln. 24
10    * removeOne()                  Ln. 29
11    * printContents()              Ln. 37
12    * contains()                   Ln. 44
13    * isEmpty()                    Ln. 49
14    * removeAll()                  Ln. 54
15    * allIndiciesOf()              Ln. 66
16    * removeMultipleInstancesOf()  Ln. 85
17    */
18
19   //This is the constructor for backpack
20   public Backpack(){
21     container = new ArrayList<E>();
22   } //End constructor
23
24   //Adds element to backpack
25   public void add(E toAdd){
26     container.add(toAdd);
27   } //End add()
28
29   /*Removes first instance of specified item
```

```
30      *  returns true if successful.
31      */
32    public boolean removeOne(E toRemove){
33      boolean result = container.remove(toRemove);
34      return result;
35    } //End removeOne()
36
37    //Prints out all the contents of the backpack
38    public void printContents(){
39      for (E anObject : container)
40          System.out.print( anObject+" ");
41      System.out.println("");
42    } //End printContents()
43
44    //Returns true if backpack has the specified element.
45    public boolean contains(E itemSought) {
46      return container.contains(itemSought);
47    } //End contains()
48
49    //Returns true if backpack is empty.
50    public boolean isEmpty() {
51      return container.isEmpty();
52    } //Ends isEmpty()
53
54    /*Removes all matching elements in backpack
55     * returns true if successful
56     */
57    public boolean removeAll(E itemSought) {
58      if (container.contains(itemSought)){
59        while(container.remove(itemSought)) {}
60        return true;
61      } else {
62        return false;
63      }
64    } //End removeAll()
65
66    //Returns an array with addresses to the elements
67    public int[] allIndicesOf(E itemSought) {
68      int i;
69      int[] returnList;
70      ArrayList<Integer> tempList = new ArrayList<Integer>();
71
72      for (i = 0; i < container.size(); i++) {
73        if (itemSought == container.get(i))
74            tempList.add(i);
75      }
76
77      returnList = new int[tempList.size()];
78      for (i = 0; i < tempList.size(); i++) {
79        returnList[i] = tempList.get(i);
80      }
81
82      return returnList;
83    } //End allIndiciesOf()
84
85    /* Will remove all instances of the specified element
86     * UNLESS the element appears once.
87     * returns True if at least one item was removed.
88     */
89    public boolean removeMultipleInstancesOf(E toRemove) {
90      if (container.indexOf(toRemove) == container.lastIndexOf(toRemove))
91        return false;
92      else
93        return removeAll(toRemove);
94    } //End removeMultipleInstancesOf()
95
96  } //End Backpack()
```

BackpackTester.java

Code

```java
import java.util.Arrays;


public class BackpackTester {

  public static void main(String[] args) {
    /* Step 1:
     * Constructing backpacks of different types
     */
    System.out.println("Constructing bags (String and Integer)");
    Backpack<String> wordBag = new Backpack<String>();
    Backpack<Integer> numBag = new Backpack<Integer>();
    System.out.println("============");

    /* Step 2:
     * Add Strings to stringBag and nums to numBag
     * try adding strings to numBag
     */
    System.out.println("Adding contents to bag");
    wordBag.add("Space");
    wordBag.add("Pirate");
    wordBag.add("Space");
    wordBag.add("Cowboy");
    numBag.add(1);
    numBag.add(1);
    numBag.add(2);
    numBag.add(2);
    numBag.add(2);
    numBag.add(2);
    numBag.add(2);
    numBag.add(3);
    //numBag.add("String"); yields a compile error
    System.out.println("============");

    /* Step 3:
     * Print contents
     */
    System.out.println("Printing contents of bags...");
    wordBag.printContents();
    numBag.printContents();
    System.out.println("============");

    /* Step 4:
     * Checking to see if a specified string is in the list
     */
    System.out.println("Searching for 'Space', then 'Landlovers'...");
    System.out.println(wordBag.contains("Space"));
    System.out.println(wordBag.contains("Landlovers"));
    System.out.println("Searching for 1 then 4...");
    System.out.println(numBag.contains(1));
    System.out.println(numBag.contains(4));
    System.out.println("Seeing if wordBag is empty...");
    System.out.println(wordBag.isEmpty());
    System.out.println("Returning indicies of 2 in numBag");
    System.out.println(Arrays.toString(numBag.allIndicesOf(2)));
    System.out.println("============");

    /* Step 5:
     * Remove functions
     */
    System.out.println("Removing ONE 2 from numBag...");
```

```
62      numBag.removeOne(2);
63      numBag.printContents();
64      System.out.println("Removing ALL 2's from numBag");
65      numBag.removeAll(2);
66      numBag.printContents();
67      System.out.println("Removing multiple occurances of 1's");
68      numBag.removeMultipleInstancesOf(1);
69      numBag.printContents();
70      System.out.println("Trying to remove 'multiple 3s'");
71      numBag.removeMultipleInstancesOf(3);
72      numBag.printContents();
73      System.out.println("============");
74    } //End main()
75
76  } //End BackpackTester()
```

Output

```
Constructing bags (String and Integer)
============
Adding contents to bag
============
Printing contents of bags...
Space Pirate Space Cowboy
1 1 2 2 2 2 2 3
============
Searching for 'Space', then 'Landlovers'...
true
false
Searching for 1 then 4...
true
false
Seeing if wordBag is empty...
false
Returning indicies of 2 in numBag
[2, 3, 4, 5, 6]
============
Removing ONE 2 from numBag...
1 1 2 2 2 2 3
Removing ALL 2's from numBag
1 1 3
Removing multiple occurances of 1's
3
Trying to remove "multiple 3s"
3
============
```