Homework Turnin

Email:	rgalanos@fcps.edu
Section:	6G
Course:	TJHSST APCS 2016-17
Assignment:	05-01
Receipt ID:	9e57a74b1574fe7a36fdffc5c04759bc

Turnin Successful!

The following file(s) were received:

TJArrayList_Driver.java (4055 bytes)

```
1.  //  name:        date:
2.  //  implements the List interface with a backing array of Objects.
3.  //  also overrides toString().
4.
5.  public class TJArrayList_Driver
6.  {
7.     public static void main(String [] args)
8.     {
9.        TJArrayList myList = new TJArrayList();
10.
11.       myList.add("Apple");
12.       myList.add("Banana");
13.       myList.add("Fig");
14.       myList.add(2, "Cucumber");
15.       myList.add(3, "Dates");
16.       System.out.println(myList);
17.       System.out.println("Size is " + myList.size());
18.       try
19.       {
20.          myList.add(12, "Peach");
21.       }
22.       catch(IndexOutOfBoundsException e)
23.       {
24.          System.out.println(e);
25.       }
26.       System.out.println("Get 2: " + myList.get(2));
27.       System.out.print("Set at 2: ");
28.       myList.set(2, "Cherry");
29.       System.out.println(myList);
30.       Object obj = myList.remove(2);
31.       System.out.println("Removed " + obj+ " from " + myList);
32.       System.out.println("Size is " + myList.size());
33.       System.out.print("Add too many items: ");
34.       for(int i = 3; i <= 10; i++)
35.          myList.add(new Integer(i));
36.       System.out.println(myList);
37.       System.out.println("Size is " + myList.size());
38.       System.out.println("Contains \"Breadfruit\"?  " + myList.contains("Breadfruit"));
39.       System.out.println("Contains \"Banana\"?  " + myList.contains("Banana"));
40.    }
41. }
42.
43. class TJArrayList
44. {
45.    private int size;                    //stores the number of objects
46.    private Object[] myArray;
47.    public TJArrayList()                 //default constructor makes 10 cells
48.    {
49.       size = 0;
50.       myArray = new Object[10];
```

```
51.    }
52.    public int size()
53.    {
54.       return size;
55.    }
56.    /* appends obj to end of list; increases size;
57.          must be an O(1) operation when size < array.length,
58.             and O(n) when it doubles the length of the array.
59.          @return true  */
60.    public boolean add(Object obj)
61.    {
62.       if(size<myArray.length)
63.       {
64.          myArray[size] = obj;
65.       }
66.       else
67.       {
68.          Object[] temp = new Object[size*2];
69.          for(int i=0;i<size;i++)
70.          {
71.             temp[i] = myArray[i];
72.          }
73.          temp[size] = obj;
74.          myArray = temp;
75.       }
76.       size++;
77.       return true;
78.    }
79.       /* inserts obj at position index.  increments size.
80.          */
81.    public void add(int index, Object obj) throws IndexOutOfBoundsException  //this the way the real ArrayList is
82.    {
83.       if(index > size || index < 0)
84.          throw new IndexOutOfBoundsException("Index: " + index + ", Size: " + size);
85.       Object[] temp = new Object[size+1];
86.       for(int i=0;i<index;i++)
87.          temp[i] = myArray[i];
88.       for(int x = size-1;x>=index;x--)
89.          temp[x+1] = myArray[x];
90.       temp[index] = obj;
91.       myArray = temp;
92.       size++;
93.    }
94.       /* return obj at position index.
95.          */
96.    public Object get(int index)
97.    {
98.       return myArray[index];
99.    }
100.    /* replaces obj at position index.
101.          */
102.    public void set(int index, Object obj)
103.    {
104.       myArray[index] = obj;
105.    }
106.     /*  removes the node from position index. shifts elements
107.         to the left.   Decrements size.
108.         @return the object at position index.
109.         */
110.    public Object remove(int index)
111.    {
112.       Object temp2 = myArray[index];
113.       Object[] temp = new Object[size-1];
114.       for(int i=0;i<index;i++)
115.          temp[i] = myArray[i];
116.       for(int x = index;x<size-1;x++)
117.          temp[x] = myArray[x+1];
118.       size--;
119.       myArray = temp;
120.       return temp2;
121.    }
122.     /*
123.       this method compares objects and should use .equals(), not ==
124.       */
125.    public boolean contains(Object obj)
126.    {
127.       boolean bool = false;
128.       for(int i=0;i<size;i++)
129.          if(myArray[i].equals(obj))
130.             bool = true;
131.       return bool;
```

```
132.     }
133.        /*returns a String of Objects in the array with square brackets and commas
134.           */
135.     public String toString()
136.     {
137.        String str = "[" + myArray[0];
138.        for(int i=1;i<size;i++)
139.           str += ", " + myArray[i];
140.        str += "]";
141.        return str;
142.     }
143. }
```