Lab 2
CS 111C

Purpose

This lab will give you experience modifying an existing ADT, as well as writing two versions of a method, (a) one that is independent of the ADT implementation, and (b) one that does not call any (ComparableListInterface) methods.

Lab

Modify ListInterface on page 230 of the textbook:

1) change Object to Comparable.

2) change the name of the interface from ListInterface to ComparableListInterface

Modify the ListArrayBased class, pages 231-232 in the textbook

1) change the name of the class from ListArrayBased to ComparableListArrayBased

2) change ListInterface to ComparableListInterface

3) change Object to Comparable

4) add a method corresponding to the following UML:

+isInAscendingOrder() : boolean {query}

isInAscendingOrder returns true if the list is in ascending sorted order, else returns false

Write two solutions to this problem, one in which isInAscendingOrder is independent of the ADT implementation, and one in which isInAscendingOrder does not call any ComparableListInterface methods. Test both of your solutions by writing a driver program, with sample dialog displayed:

Input a list of integers: **5 9 101 183 4893**
Your list of integers is in ascending order.
Do you want to continue (y/n): **y**

Input a list of integers: **5 9 101 183 48**
Your list of integers is not in ascending order.
Do you want to continue (y/n): **y**

Input a list of integers: **5 4 100 101 183 4893**
Your list of integers is not in ascending order.
Do you want to continue (y/n): **y**

Input a list of integers: **5 9 101 101 183 4893**
Your list of integers is in ascending order.
Do you want to continue (y/n): **y**

Input a list of integers: **-48 -7 0 5 9 101 183**
Your list of integers is in ascending order.
Do you want to continue (y/n): **y**

Input a list of integers: **14**
Your list of integers is in ascending order.
Do you want to continue (y/n): **y**

Input a list of integers:
Your list of integers is in ascending order.
Do you want to continue (y/n): **n**

To turn in

Remember to submit both your program listing(s), including both solutions, and captures of the program runs as shown above, for both solutions, on Insight.

Please note the following corrections to the text:

Page 231

```
public void add(int index, Object item)
                    throws ListIndexOutOfBoundsException
{
        if (numItems >= MAX_LIST)          // >= instead of >
…
                    // insert – 1 after numItems
                    for (int pos = numItems – 1; pos >= index; pos--)
```

Page 232

```
public void remove(int index)
                    throws ListIndexOutOfBoundsException
{
        …
                    // < numItems instead of <= size()
                    for (int pos = index + 1; pos ~~<= size()~~ < numItems; pos++)
```