Data Structures: ArrayList-like Structures, Stacks, and Queues

Re-submit Assignment

Due Oct 23 by 11:59pm **Points** 32 **Submitting** a file upload **File Types** zip

Description

Build three classes that conform to the following interfaces. Use arrays in creating your classes (e.g., do not use the built-in ArrayList class when creating the ArrayList-like interface). Extend the sample driver below to completely test the FIFO nature of a Queue, the LIFO nature of a stack and the arbitrary inserts and removes in an ArrayList-like structure.

Sample Driver: **ArrayBasedDataStructuresDriver.java**

ArrayList-like Interface

- void insert(Object, int index): Add the object at the specified index.
- Object remove(int index): Remove and return the object at the specified index.
- int size()
- String toString()
- boolean isEmpty()
- int getIndexOf(Object): Returns -1 if not found
- boolean equals(Object): Compare sizes and elements in the data structure.
- Object get(int index): Returns the object at index specified.

Stack Interface (LIFO)

- void push(Object)
- Object pop()
- int size()
- String toString()
- boolean isEmpty()
- boolean equals(Object)

Queue Interface (FIFO)

- void enqueue(Object)
- Object dequeue()
- int size()
- String toString()

- boolean isEmpty()
- boolean equals(Object)

Hints and Tips

- Deliverables: You should turn in four .java files, three are for your classes and the fourth is your test harness (driver). Put the java files inside of a single .zip file. The file names do not need to include your name, but be sure your name is in the comments inside each file.
- You should practice building drivers and even more advanced testing harnesses yourself. The above driver is merely a starting point that you can use to initially exercise your code. In lecture, we will talk about what a testing harness should do. For this assignment, you can just display the results of your test to screen. Once we've gone through exception handling, you will have all the skills needed to write the harness the way I described in class.
- A reminder that Stacks should be LIFO, queues should be FIFO, and ArrayLists can add and remove in arbitrary order.
- The data structures should hold objects of class Object.
- Consider completing your ArrayList class, then using it for the other two classes. You only have to solve the trickier issues once.
- Q: The list of methods I'm supposed to write seems to be missing methods I need in order for me to actually create and use an instance of the class. For instance, my ArrayList-like class, if it does not have an add or append method, cannot be used for anything. Can I create additional methods? A: Yes, you should create any methods you believe necessary in order to make the class work. The list of methods given above are just the ones we're looking for to grade.
- Q: Do these structures hold a fixed size set of items (i.e., static), or can they grow at runtime? A: Your software should be able to automatically resize your array once capacity is reached, and may be tested beyond 100 elements. Remember that stacks, queues, etc. are (in the abstract) infinite capacity data structures.
- Q: What is FIFO and LIFO? A: Good question.
- Q: How will you test this code? A: Tests may use the same strategies and techniques shown to you in the first few HWs, but more complete.
- Q: Are comments (file headers, method headers, inline comments) important? A: Comments are now worth almost 2 letter grades, so I'd include them.
- Q: I think there is a way to relate these classes via inheritance. Should I do this? A: We'll cover this later in the quarter; you should use no inheritance for this assignment.

About This Document

Original assignment by Rob Nash. Minor edits by Johnny Lin, January 2015.

Data Structures V1

Criteria	Ratings			Pts
Code compiles and driver executes	**4.0 pts** **Full Marks**	**2.0 pts** **Partial Marks**	**0.0 pts** **No Marks**	4.0 pts
Clean, descriptive, and comprehensive comments	**8.0 pts** **Excellent**	**5.0 pts** **Good**	**3.0 pts** **Average**	**0.0 pts** **Poor**	8.0 pts
List class	**6.0 pts** **Full Marks**	**3.0 pts** **Partial Marks**	**0.0 pts** **No Marks**	6.0 pts
Queue class	**4.0 pts** **Full Marks**	**2.0 pts** **Partial Marks**	**0.0 pts** **No Marks**	4.0 pts
Stack class	**4.0 pts** **Full Marks**	**2.0 pts** **Partial Marks**	**0.0 pts** **No Marks**	4.0 pts
Extended driver tests every substantive method of the classes	**6.0 pts** **Full Marks**	**3.0 pts** **Partial Marks**	**0.0 pts** **No Marks**	6.0 pts
				Total Points: 32.0