PA7 Generics and Interfaces

Due: Wednesday 10/28 by 11:30PM
Submission:

- PA7.pdf - answers to the written questions.
- StackInterface.java - generic interface for stacks.
- ListStack.java - generic implementation of a stack backed by a linked list.
- QueueInterface.java - generic interface for queues.
- ListQueue.java - generic implementation of a queue back by a linked list.

Overview

Congrats! You have made it to the shortest PA of the semester. I spent some time trying to think of ways to make this PA harder, but then I realized, why do I need to make it artifically harder? If you learn how to use generics, that is the only goal of this assignment. So please spend some of the time you would normally devote to writing a larger assignment to a careful reviewing and understanding of the links posted on Piazza regarding generics and the video released on generics.

The purpose of this assignment is to give you practice writing and implementing interfaces, and writing generic classes.

The assignment consists of a written part answering questions and a coding part. You can do the written or the coding part first, it is up to you.

Assignment

Written

1. What is the purpose of generics? Why do we need them?

2. Why don't we just use Object as the parameterized type for our generic classes? Then we could put absolutely any object into our list. i.e. When we made MyLinkedList generic, why don't we always declare it like:

```
MyLinkedList<Object> llist = new MyLinkedList<Object>();
```

3. In class we made MyLinkedList generic and not MyArrayList. Why did we choose to make MyLinkedList generic first? Are there any issues when combining generics and arrays?

4. How many type parameters are you allowed to use in a generic class?

5. Explain Type Erasure. How does it work?

Code

For PA7, you will take your implementations of a Stack backed by a linkedList (ListStack.java) and a queue backed by a linkedList (ListQueue.java) and make these implementations generic. **This may require adding zero additional lines of code to these files! Wow!** You will also adapt the StackInterface.java and QueueInterface.java files that we provided in the starter code.

First update StackInterface.java. The version we gave you for PA6 has been provided in the starter code for PA7 as well. This interface is specific to stacks of integers. Make it generic!

Update QueueInterface.java. The version we gave you for PA6 has been provided in the starter code for PA7 as well. This interface is specific to queues of integers. Make it generic!

Update your implementation of ListStack to be generic.

Update your implementation of ListQueue to be generic.

Also, do not forget to override equals, toString, and write a copy constructor just as you did for the last assignment. You will also need a constructor with zero arguments.

A small note: when writing the equals methods for these classes, you might encounter a warning like: "Type safety: Unchecked cast from Object to ListStack<E>". Do not worry about it. There is only so much we can cover in this class, we will let this one slide by.

Error Handling

We will use very bad error handling practices in this PA. We have to do this until we learn about exceptions. If the user calls an erroneous method you should ignore it as best as you can. Of course, some of the methods still require a return value. Since you have to return an object of type E. Instead of returning -1 or some other error code. Just return null.

Grading Criteria

We are not providing testcases for this PA. As long as you implement the same methods from PA6 our autograder will be able to call those methods on your classes. That is how we will test your implementations. Of course don't forget to write a copy constructor and override the equals and toString methods. You should not print anything in any of your classes. i.e. do not call 'System.out.println' anywhere in your code. You can (and should) use print statements to debug but be sure to remove them all before submitting your files.

We encourage you to write your own JUnit testcases to ensure your classes work properly, but we will not be collecting or grading these test files. We will test your classes with our own testcases.

Your grade will consist of similar style and code clarity points we have looked for in earlier assignments.

Write your own code. We will be using a tool that finds overly similar code. Do not look at other students' code. Do not let other students look at your code or talk in detail about how to solve this programming project. Do not use online resources that have solved the same or similar problems. It is okay to look up, ”How do I do X in Java”, where X is indexing into an array, splitting a string, or something like that. It is **not** okay to look up, ”How do I solve {a programming assignment from CSc210}” and copy someone else's hard work in coming up with a working algorithm.