Assignment 6

Exceptions & Using ADTs

Introduction

In this assignment, you will add exceptions to a generic Queue class and then use a queue to implement a program that simulates a lineup of people waiting to enter a musical/sporting event venue. The focus of Part 1 is on throwing exceptions when methods are called incorrectly (or under incorrect circumstances). In Part 2, you will use the queue (that you have updated so that it correctly works with exceptions) to model a lineup at an event.

The automated grading of your assignment will include some different and additional tests to those found in the `A6Tester.java` file, as it does not include a comprehensive set of sets for each method. You are expected to write additional tests until you are convinced each method has full test coverage. The displayResults and test coverage videos provide more information about code testing.

Objectives

Upon finishing this assignment, you should be able to:

- Write methods that throw exceptions
- Write code that handles exceptions (using try and catch)
- Write code that uses an ADT to solve a problem

Submission and Grading

Attach `GenericQueue.java` and `EventLine.java` to the BrightSpace assignment page. Remember to click **submit** afterward. You should receive a notification that your assignment was successfully submitted.

If you chose not to complete some of the methods required, you **must** provide a stub for the incomplete method(s) in order for our tester to compile. If you submit files that do not compile with our tester, you will receive a zero grade for the assignment. It is your responsibility to ensure you follow the specification and submit the correct files. Additionally, your code must not be written to specifically pass the test cases in the tester, instead, it must work on all valid inputs. We may change the input values during grading and we will inspect your code for hard-coded solutions. This video explains stubs.

Be sure you submit your assignment, not just save a draft. All late and incorrect submissions will be given a zero grade. A reminder that it is OK to talk about your assignment with your classmates, but not to share code electronically or visually (on a display screen or paper). Plagiarism detection software will be run on all submissions.

Instructions

Part 1:

1. Download all of the `.java` files found in the *Assignments* > *Assignment 6* page on BrightSpace.
2. Read through the tests provided in `A6Tester.java`.
3. Compile and run `A6Tester.java`. The first few tests should pass.
4. Eventually you will see some tests enclosed in try-catch blocks. In order for these tests to pass, you will need to throw exceptions in the corresponding methods. If you read through the `Queue.java` interface you will see the documentation has been updated to specify which methods throw exceptions.
5. In each of the methods that throw an exception, determine what input scenario triggers the exception, and then add the code to throw the exception under those circumstance(s).
6. **Remember:** After adding the exceptions in the correct places in `GenericQueue.java` and saving it, compile and run `A6Tester.java`. All tests for Part 1 should pass. If any tests do not pass, it means you are either not detecting the correct circumstances that throw the exception, or you are not throwing the correct exception. Reminder: the `Queue.java` interface specifies which exception should be thrown in each method that throws an exception.

Part 2:

1. Read the documentation provided above each method in `EventLine.java`.
2. You will need to implement each method using **only** the fields *provided for you* in the EventLine class.
3. Compile and run `A6Tester.java`. Work through implementing each method one at a time. Debug the method until all of the tests pass for that method before proceeding to the next method.
4. **Note:** None of the methods in the EventLine class throw an exception. If they call a method that throws an exception, they will need to handle that exception (ie. the exception does **not** make it back so that it is instead caught in A6Tester).

CRITICAL: Any compile or runtime errors will result in a **zero grade** (if the tester crashes it will not be able to award you any points for any previous tests that may have passed). Make sure to compile and run your program before submitting it!