P10 HELP DESK

Posted on April 15, 2019 by dahl

- **SUBMIT your completed assignment by 9:59PM on Wednesday, April 24th 2019.** We will accept and grade submissions made through 9:59PM on Thursday, April 25th 2019 (HARD DEADLINE). But, NO CREDIT will be granted for any work that is submitted even one second after this hard deadline.
- **Pair Programming is NOT allowed for this assignment. You have to work on and complete this assignment INDIVIDUALLY.**

This assignment involves developing a system to prioritize support tickets for a the help desk of a local business. You will implement a priority queue using the heap data structure, for this purpose. After developing this system, you will need to develop several tests that demonstrate the correctness of your implementation under a variety of specific circumstances. These tests will be used to evaluate the correctness of other implementations, just as other tests will be used to evaluate the correctness of yours.

OBJECTIVES AND GRADING CRITERIA

The main goals of this assignment include giving you practice and experience both implementing a priority queue and using the heap data structure. You will also get further practice developing your own test methods.

5	Online Tests: these automated grading test results are visible upon uploading your submission. You are allowed multiple opportunities to correct the organization and functionality of your code (if necessary).
45 points	Offline Tests: these automated grading tests are run after the assignment's deadline has passed. They check for similar functionality and organizational correctness as the Online Tests. Since you will not have opportunities to make corrections after seeing the feedback from these tests, you should consider and test the correctness of your own code as thoroughly as possible.

0. GETTING STARTED

Create a new Java8 project in Eclipse. You can name this project whatever you like, but P10 Help Desk is a descriptive choice. You'll be submitting your code for this assignment through gradescope. The four files that you will be submitting include: SupportTicket.java, HelpDesk.java, HelpDeskInterface.java, and HelpDeskTestSuite.java.

1. SUPPORT TICKETS

Create a new class called SupportTicket, which implements the Comparable<SupportTicket> interface. This class should contain a constructor that takes a String message in as input, and stores that string within an instance field. This String message should then be returned whenever toString() is called on that SupportTicket object. When the constructor of the SupportTicket class is provided a null String argument, a NullPointerException should be thrown to indicate that this is a problem.

In order to prioritize these tickets within a priority queue (in the next step), you must first implement the compareTo() method that is part of the Comparable interface that this class implements. This should be defined so that the natural ordering of support tickets corresponds to the length of the message inside them: longer messages should be considered later in this natural ordering (aka larger). Whenever the two messages being compared have the same length, their lexographical (alphabetical) order should be used to attempt to break such ties. For example, a SupportTicket with the message ZZZ should be considered later in the natural ordering (aka larger) than a SupportTicket with the message AAA.

2. HELP DESK PRIORITY QUEUE

Next create a new class called HelpDesk to contain you implementation of a priority queue. This class must implement the HelpDeskInterface.java interface: source code available here. Note that this interface includes only three required public methods, and that there are additional required protected methods listed in comments. This class must also include a single public constructor that takes the internal array capacity as an int parameter. This capacity is fixed, and does not need to grow when filled. No additional methods are necessary, but if you choose to implement additional helpers they must be private. The ONLY fields that you may include within your HelpDesk class are the following:

```
protected SupportTicket[] array; // zero-indexed max-heap
protected int size;
```

The priority queue that this class represents will be implemented as a max-heap: where the SupportTicket at the root of this heap is always the support ticket containing the longest message, ie the ticket that is latest in the natural ordering of messages. The root of this heap must always be at index zero within your array.

3. HELP DESK TEST SUITE

Another team will be building the GUIs (Graphical User Interfaces) for this system. So your last task it so develop a comprehensive set of test methods that demonstrate the correctness of your implementation. Create a class called HelpDeskTestSuite, and implement eight or more test methods to check the widest possible range of errors that any HelpDesk implementation might contain. Each of these test methods must be public static methods that take zero parameters and return a Boolean value: false to indicate a failed test, and true otherwise. Having these methods print feedback about any defects that they find is optional, but highly recommended. You should also implement a main method that calls each of your tests and displays which (if any) of those tests detected implementation defects.

In order to access and test the protected methods defined in the HelpDesk class, your HelpDeskTestSuite class should extend HelpDesk. You shouldn't ever need to create any HelpDeskTestSuite objects, but this will give you access to those protected methods so that you can test their correctness.

In addition to evaluating your HelpDesk implementation, the automated grading tests for this assignment will also test the methods within HelpDeskTestSuite. This will be done by running your test methods with different implementations of the HelpDesk class: some containing defects and others that are correct. They will check whether your test methods correctly identify defects that are present, and whether they correctly identify good working implementations. In order for this to work, it is critical that your tests do not make use of any methods beyond those specified in HelpDesk, and it is also important that you only check for behavior that is required for this assignment.

SUBMISSION

Congratulations on finishing this final CS300 assignment! After verifying that your work is correct, and written clearly in a style that is consistent with the course style guide, you should submit your final work through gradescope.com. The four files that you must submit include: SupportTicket.java, HelpDesk.java, HelpDeskInterface.java, and HelpDeskTestSuite.java. Your score for this assignment will be based on your "active" submission made prior to the hard deadline of **9:59PM on Thursday, April 25th**.