Lab 6

Objectives

- Programming experience implementing the Stack and Queue data structures
- Programming experience using Generics in Java

Part1

In this part of the lab you will be implementing the Stack interface.

1. Download the files for Part1 of this lab into your Lab6 folder.
2. Implement each of the methods in StackArrayBased.java according to the documentation in the Stack interface.

CHECKPOINT (Ungraded) – Now might be a good time to check-in with the TA if you are failing any of the tests in Lab6TesterPart1.java.

3. Implement the methods reverseString and doBracketsMatch in Lab6TesterPart1.java according to the documentation and tests provided. You must use a stack in your implementation

CHECKPOINT (Ungraded) – Now might be a good time to check-in with the TA if you are failing any of the tests in Lab6TesterPart1.java.

Part2

Now you will implement a generic Queue.

1. Download the files for Part2 of this lab into your Lab6 folder.
2. Change the interface (Queue.java) to be of generic type. This will force you to make changes in multiple places
    a. The class that implements the interface and its dependent classes (QueueRefBased.java and QueueNode.java) must be made generic
    b. Any code that creates a new QueueRefBased must now indicate the type of data that Queue will hold. This type must be the object type. For example:
    Queue<Integer> intQ = new QueueRefBased<Integer>();
    Queue<Character> charQ = new QueueRefBased<Character>();

    TIP: make changes to one class at a time, compiling/updating until no compilation errors.
    For example, start with Queue.java, repeatedly make the changes/compile until complete:
        javac -Xlint:unchecked Queue.java
    Then move on to QueueNode.java and do the same, and finally for QueueRefBased.java.

3. Uncomment the code provided in the tester following the line:
    //Testing generic queue with Character type

4. If you have updated your code in all of the necessary places it should compile without warnings.

    NOTE: if you get the warning like the following…
    Note: Lab6TesterPart2.java uses unchecked or unsafe operations.
    Note: Recompile with -Xlint:unchecked for details.

    Do as suggested and recompile as follows (it will indicate the line numbers of where the problem code is): javac -Xlint:unchecked Lab6TesterPart2.java

5. Complete the implementation of the stubs provided in QueueRefBased.java according to the documentation in the Queue interface and the tests provided.

SUBMISSION (Graded) – Submit the **QueueRefBased.java** file into the Lab 6 submission page on ConneX.

Finished early? – start your Assignment!