CPSC 2151

Lab 6

Due: Friday, October 2nd at 10:00 pm

In this lab you will be converting you `IQueue` interface and implementations to be generic.

Instructions

1. Create a copy of your project from lab 5 to edit for lab 6.
    a. You should have gotten feedback from your TAs about lab 4. Fix any issues with contracts and/or code that they have identified before proceeding with this week's lab.
2. Convert `IQueue`, `AbsQueue`, `ArrayQueue` and `ListQueue` to all be generic
    a. Remember, you will need to perform some casting with the `Object` class with a generic array
3. Make the changes to `QueueApp` to allow for it to use the new generic implementations of `IQueue`, but still use a queue of `Integer`s
4. Make a new class called `DoubleQueueApp`. `DoubleQueueApp` will be essentially the same as `QueueApp`, but it will work with a queue of `Double`s.
    a. **Note:** You can make multiple build configurations in IntelliJ so you can run either `DoubleQueueApp` or `QueueApp`
    b. **Note:** We are not adding any new functionality, so no new sample input and output are provided.
5. Create a `makefile` and test your code on SoC Unix before submitting. Your `makefile` should have the following targets
    a. `default`: used to compile the code. Remember, with Java you can compile a set of files that has multiple entry points, so we do not need to compile our `Integer` and `Double` implementations separately.
    b. `runInt`: Used to run `QueueApp`
    c. `runDouble`: Used to run `DoubleQueueApp`
    d. `clean`: remove all compiled `.class` files

Note

IntelliJ can be more lenient with generics than our Unix compiler is, so make sure to test on SoC Unix to avoid any issues. If you get an error message about a type issue, and to include "-Xlint" to see more details, that is telling you to add "-Xlint" as a flag to your compile command, i.e "`Javac -Xlint cpsc2150/…`" Most of the time these errors and warning are due to missing a <> in the constructor call for a generic type or passing a generic type into a method and leaving out what type to expect. So, for instance if you have a method that takes in an `IQueue` of `Double`s, then the parameter type needs to be `IQueue<Double>` even though IntelliJ often accepts just `IQueue` or `IQueue<>`.

Partners

You may work with one partner on this lab assignment and you are encouraged to do so. Make sure you include both partners' names on the submission. You only need to submit one copy. Remember that

working with a partner means working *with* a partner, not dividing up the work. You will need this code for a later lab, so make sure both partners have a copy of it.

Before Submitting

You need to make sure your code will run on SoC Unix machines and create a `makefile`.

Submitting your file

You will submit your files using handin in the lab section you are enrolled in. If you are unfamiliar with handin, more information is available at https://handin.cs.clemson.edu/help/students/. You should submit a zipped directory with your package directory and your `makefile`. The TA should be able to unzip your directory and type `make` compile your code, `make runInt` (or `make runDouble`) to run it and `make clean` to delete any `.class` files.

NOTE: Make sure you zipped up your files correctly and didn't forget something! Always check your submissions on handin to ensure you uploaded the correct zip file.