CSE250, Spring 2014 Assignment 2 Due Fri. 2/14 online 11:59pm

This assignment is for online-submission only at the (UB CSE-traditional?) **midnight time**.

Office Hours from Week 3 onward:
K.W. Regan, Mon. 11:00–11:50am, Wed. 1–3pm, *often* Thu. 4:45–5:45pm
Ladan Golshanara, Thu. 1–3pm
Changsha Ma, Mon. 2–4pm.

Reading. For next week, read the `Java2C++` notes on the course webpage. The "Weeks 1–2" lecture slides parallel those notes. If you need more background on stacks, queues, and double-ended queues (deques) beyond what we expect from the second-semester course or cover in class, also read Sections 5.1, 6.1, and 6.4, but steer clear of the advanced implementation details and C++ code in the later sections—this assignment keeps things simple using a non-circular array.

By next week we'll assume you have read all the `Hello---.cpp` files in the `Java2C++` directory on `timberlake` (note they are mirrored on the course webpage too, for temporary ease of download into your home systems), and now you should read `StackTest.java` and `DequeTest.java` for this assignment. Recitations next week will include a "hands-on" of translating `StackTest.java` into C++ which parallels the online-sub part of this assignment—it is AOK for you to look in advance at the various C++ versions, which are only on `timberlake`. Also peek ahead to the files with "Link" in their name.

Programming exercise: Get the file `DequeTest.java` from the directory `~regan/cse250/Java2C++/` on `timberlake` (if you don't have it already). Translate it into an equivalent C++ program `DequeTestNNN.cpp`, where `NNN` are your initials. (Note that C++ unlike Java does not mandate any agreement between file and class names, and we are not using separate `.h` and `.cpp` files for this exercise—not yet.)

Some hints in the recitation-exercise file `StackTest.java` and also `StringClient.java` are valid for this task. Your translation should follow the rules laid out in lectures and recitations, *except* that you may translate Java `String` using the simple C++ value-type `string`, rather than use `const string*` as technically indicated in `HelloString.cpp` in lecture. Otherwise you must translate object references via pointers, and must use `virtual` and `const` as needed. There is a 2-page handy Java-to-C++ translation checklist on the CSE course webpage, but it is not completely exhaustive.

Finally in a report question, answer whether you see any difference in output between the C++ and Java programs. *It might not be a mistake.* If so, see if you can explain it. Place your answer in a comment at the bottom of your .cpp file, and submit it by

```
submit_cse250 foo
```

Here as usual "foo" means the name of your file (sometimes to allow for submitting multiple files one could write "fooze"). Following policy from previous terms, now you are welcome to develop your C++ code on your home system, but *you must test it on* `timberlake` *so that it compiles and runs there.* (45 pts. total on the problem set)