University of Waterloo
CS240E, Winter 2021
Programming Question 1

(Programming part due Wednesday, February 3, 2021 at 5pm)

The integrity of the grade you receive in this course is very important to you and the University of Waterloo. As part of every assessment in this course you must read and sign an Academic Integrity Declaration (AID) before you start working on the assessment and submit it **before the deadline of February 3** along with your answers to the assignment; i.e. **read, sign and submit PQ01-AID.txt now or as soon as possible**. The agreement will indicate what you must do to ensure the integrity of your grade. If you are having difficulties with the assignment, course staff are there to help (provided it isn't last minute).

The Academic Integrity Declaration must be signed and submitted on time or the assessment will not be marked.

Please read `http://www.student.cs.uwaterloo.ca/~cs240e/w21/guidelines/guidelines.pdf` for guidelines on submission.

Question 1 [Programming, 18 marks]

In this problem, we study an online algorithm for finding the median of a sequence: we suppose that we receive the entries of the sequence one at a time, and we want to print the medians of all these partial sequences as we go. Here is one example:

- We first receive 15. We print 15, the median of the sequence (15)
- Next, we receive 10. We print 10, the median of the sequence (15, 10)
- Next, we receive 1. We print 10, the median of the sequence (15, 10, 1)
- Next, we receive 20. We print 10, the median of the sequence (15, 10, 1, 20)
- Next, we receive 30. We print 15, the median of the sequence (15, 10, 1, 20, 30)

Re-computing the median from scratch every time would be too slow. Here is an idea for a better algorithm: use two heaps h_{lo} and h_{hi}, each of which will roughly contain half of the elements seen so far: if we have seen n elements, h_{lo} should contain the $\lceil n/2 \rceil$ smallest elements, h_{hi} should contain the $\lfloor n/2 \rfloor$ largest ones. (You may assume that all elements are distinct, so that there are no ties.)

Figuring out the details of what kind of heaps (min-oriented or max-oriented) the two heaps h_{hi} and h_{lo} should be, and how to update them when inserting a new element is part of this assignment.

Implement a class `OnlineMedian` that gives an efficient ($o(n)$) implementating for returning the median of a current set while inserting elements and/or deleting the median.

The provided file `OnlineMedian.cc` gives you a stub for the required methods `insert`, `getMedian`, `deleteMedian` and `size`. Submit this file (after suitable additions). Your file should either not have a main routine, or it should be surrounded by `#ifndef TESTING` as done in the stub.

Evaluation:

- 8 marks for correctly returning the median for a number of tests.
- 7 marks for doing so within the run-time that one can achieve when using two heaps as outlined in the idea. (These marks will only be given if you return the correct result for at least some of the tests.)
- 3 marks for good programming style, indentation, comments, etc.