CONCORDIA UNIVERSITY
Department of Computer Science
and Software Engineering

SOEN 331-W: Introduction to Formal Methods for Software Engineering
Winter 2019

Assignment 3 on Contract programming

Dr. Constantinos Constantinides, P.Eng.

February 27, 2019

Contents

1 General information

Date posted: Wednesday 27 February, 2019.

Date due: Wednesday 13 March by 23:59.

Demos: Thursday 14 March and Friday 15 March (See Section 6).

Weight: 5% of the overall grade.

2 Introduction

This assignment must be done by teams of 3 or 4. Each team should designate a leader who will make an appointment for a live demonstration of your program and submit your assignment electronically.

3 The Priority Queue ADT

The Priority Queue abstract data type (ADT) is similar to an ordinary Queue ADT where each element is inserted with an associated key. The key determines the element's priority. A Priority Queue ADT does not enforce a first-in-first-out (FIFO) policy. Instead, elements are dequeued by order of priority. For two (or more) elements of equal priority, it is not specified which one has priority. Some priority queues are implemented in such a way as to enforce a FIFO policy among equal priority elements. The interface of the Priority Queue ADT contains the following main functionality:

`insert(el ElementType, key KeyType)` Inserts the pair (el, key) into the priority queue.

`remove()` Removes and returns the element with the smallest key.

`min()` Returns but does not remove the element with the smallest key.

4 Your assignment

Your assignment is to implement a bounded Priority Queue ADT in your language of choice together with contractual specifications. Functions `remove()` and `min()` follow the rules of

a regular queue in the case where the collection contains more than one entry with the same key. You must use a *binary heap* for the implementation of the Priority Queue ADT. You must define both the heap and the priority queue from scratch, instead of reusing library components.

Based on your choice of programming language, contracts may be provided through native support (e.g. Clojure), through third party support (e.g. adbc in Java), through macro facilities (e.g. CLOS for Common Lisp), or through language extensions (e.g. AspectJ for Java). An extensive list of technologies is provided in the "design by contract" wikipedia article at `https://en.wikipedia.org/wiki/Design_by_contract#Language_support`.

5 What to submit

Please follow the instructions carefully:

1. Prepare a README.txt file with detailed instructions on what tools, if any, we need in addition to the language compiler, how to install and configure these tools and how to run your code.

2. Place the above file with all your source code files into a folder named after the University id of the person who will submit, and zip the folder.

3. Submit your `zip` file at the Electronic Assignment Submission portal (`https://fis.encs.concordia.ca/eas`) under **Assignment 3**.

6 Demonstration

Would the leader of your team see the instructor in person (**during office hours only**) to make an appointment to reserve a time slot to demonstrate your program to the TAs. We will run one session on Thursday 14 March between 13:00 - 18:00, and two parallel sessions on Friday 15 March between 16:30 - 19:30.

Your demonstration should cover the main functionality of the Priority Queue ADT, simulating assertion failures. The demonstration should take approximately 10 minutes. You should also be prepared to answer questions by the TAs. As this is an assessment exercise, your entire team must be present during the demonstration.