Mary Garrity, Ledong Shi, Enrique Javier

CUS 1151

Homework 1 Write Up

Why did you choose your particular tests in the QueueTest.java file? For testEmpty and testOne, a couple sentences will do. For testMany, explain why you think some implementations might fail those tests.

For testEmpty, we first check that the front method and the dequeue method return the same value for our class and the java implementation. We then use the enqueue method to add one element to each queue, then use the front method of each queue to check that both classes return the same value.

For testOne we're first using the enqueue method of our class and the enqueue method of the java implementation to add one element to each queue, then we check that the front method of our class returns the same value as the front method of the Java implementation. Next we call the dequeue method of each class to remove the element just added, and then we check that the front method of our class returns the same value as the java implementation. We then use the enqueue method of each class to add another element to each queue and then check that the front method of our class returns the same value as the java implementation.

For testMany, we use the enqueue methods of our class and the enqueue method of the java implementation to add two elements to each queue. We then check that the front method of our class returns the same value as the java implementation. We then use the dequeue method to remove one element from both our class and the java implementation. And we check that the front method of our class returns the same value as the java implementation. We then use

enqueue to add another element to each queue and then check that the front method of our class returns the same value as the java implementation. We then use enqueue to add four elements to each queue, and then check that the front method of our class returns the same value as the java implementation. We then use the dequeue method to remove an element of each queue, and then we check that the front method of our class returns the same value as the front method as the java implementation. Some implementations of the testMany test might fail because we can never test every possible combination of inputs and we can never test an infinite number of inputs.

After running your tests on your code, how confident are you that your implementation is correct? Explain why you think your test cases are sufficient, or alternately explain what additional tests might be prudent. These explanations should be at a high level and do not require any implementation.

After testing our code I am confident that our implementations are correct. I believe that our test cases are sufficient because for testEmpty, testOne, and testMany, we test the front, enqueue, and dequeue methods on both our class and the java implementation. Our program prints “true” and “false” after running the tests we implemented.

If we wanted to add additional tests, we could have also checked that both queues are returning null in testEmpty. In testOne, we could also have called the front method until it returned ‘not null’ when we were checking that each queue has an element. We could have also called dequeue to remove the first element of each queue and called front to make sure the queues were empty. In testMany, we could also have called the front method until it returned ‘not null’ when we were checking that each queue has an element. We could have also called

dequeue to remove the first element of each queue and called front to make sure the queues are not empty.