1. 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.

There were three different types of tests, testEmpty, testOne, and testMany to simulate the three different conditions for which the queue will face.

testEmpty checks what the front() and dequeue() method returns when the queue is empty. It also compares it to what the java implementation of a queue returns when the front() and dequeue() method are called for an empty queue.

testOne checks to make sure that when there is a single element in the queue, the front() and dequeue() method return the same outputs as the java implementation.

testMany tests what happens when there are more than 10 inputs to the linked list. This is in order to make sure that the implementation is not actually a circularly linked array. Furthermore, testMany runs two tests consecutively in order to identify what happens when the queue is enqueued, dequeued, and then enqueued again.

I believe some implementations may fail the test with more than 10 inputs because the implementation may be an array and thus will have a fixed size. By exceeding the size, the implementation will fail the test. I believe that some implementations will fail the second test because there is a distinction between when a queue is empty because it was just created or when it is empty because it had been just dequeued. If the implementation is incorrect, the use of a queue after being emptied will point to the incorrect node.

2. 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.

I am very confident that my implementation is correct. I think my test cases are sufficient because a queue is a very simple ADT. I tested when the queue is empty, has one element, and when it has many elements which covers most of the use cases for a queue. Another test case was when the queue is enqueued, dequeued, and enqueued again because having an empty queue on initialization and an empty queue after dequeuing are two completely different things. Compared to another ADT like a heap, the queue ADT doesn’t have that many tests required to provide a sufficient level of confidence. Because of this, I am confident that I have covered most of the properties that would make the queue implementation correct.