```
 1: // Copyright 2015 fredm@cs.uml.edu for 91.204 Computing IV
 2: // Wed Mar 25 06:32:17 2015
 3:
 4: #define BOOST_TEST_DYN_LINK
 5: #define BOOST_TEST_MODULE Main
 6: #include <boost/test/unit_test.hpp>
 7:
 8: #include <stdint.h>
 9: #include <iostream>
10: #include <string>
11: #include <exception>
12: #include <stdexcept>
13:
14: #include "RingBuffer.hpp"
15:
16: BOOST_AUTO_TEST_CASE(RBcontructor) {
17:     // normal constructor
18:     BOOST_REQUIRE_NO_THROW(RingBuffer(100));
19:     // this should fail
20:     BOOST_REQUIRE_THROW(RingBuffer(0), std::exception);
21:     BOOST_REQUIRE_THROW(RingBuffer(0), std::invalid_argument);
22: }
23:
24: BOOST_AUTO_TEST_CASE(RBenque_dequeue) {
25:     RingBuffer rb(100);
26:     rb.enqueue(2);
27:     rb.enqueue(1);
28:     rb.enqueue(0);
29:     BOOST_REQUIRE(rb.dequeue() == 2);
30:     BOOST_REQUIRE(rb.dequeue() == 1);
31:     BOOST_REQUIRE(rb.dequeue() == 0);
32:     // this should throw an exception when dequeue an empty buffer
33:     BOOST_REQUIRE_THROW(rb.dequeue(), std::runtime_error);
34:     RingBuffer rb1(1);
35:     rb1.enqueue(1);
36:     // this should throw an exception when enqueue a full buffer
37:     BOOST_REQUIRE_THROW(rb1.enqueue(1), std::runtime_error);
38: }
39:
40: BOOST_AUTO_TEST_CASE(RBpeek) {
41:     RingBuffer rb(10);
42:     // throw exception when peek into empty Ringbuffer
43:     BOOST_REQUIRE_THROW(rb.peek(), std::runtime_error);
44:     rb.enqueue(654);
45:     rb.enqueue(2);
46:     BOOST_REQUIRE_NO_THROW(rb.peek());
47:     BOOST_REQUIRE(rb.peek() == 654);
48:     rb.dequeue();
49:     BOOST_REQUIRE(rb.peek() == 2);
50: }
```