```
 1: // Copyright 2015 <Angel Z'heondre Calcano>
 2: // PS5b
 3: #include <stdint.h>
 4: #include <cstdlib>
 5: #include <stdint.h>
 6: #include <stdexcept>
 7: #include <iostream>
 8: #include <string>
 9: #include <vector>
10: #include <SFML/System.hpp>
11: #include "RingBuffer.hpp"
12:
13: int RingBuffer::size() { return _currentcapacity; }
14:
15: bool RingBuffer::isEmpty() { return _buffer.empty(); }
16:
17: bool RingBuffer::isFull() {
18:   if (_buffer.size() == (unsigned)_size)
19:     return true;
20:   if (_buffer.size() >(unsigned)_size)
21:     return false;
22:   else
23:     return false;
24: }
25:
26: void RingBuffer::enqueue(int16_t x) {
27:   if (_currentcapacity == _size) {
28:     throw
29:       std::runtime_error("Can't enqueue on a full ring");
30:   }
31:   if( _currentcapacity > _size) {
32:     throw
33:       std::runtime_error("Can't enqueue on a full ring");
34:   }
35:   _buffer[_last] = x;
36:   _last++;
37:   _currentcapacity++;
38:   if (_last == _size)
39:     _last = 0;
40: }
41:
42: int16_t RingBuffer::dequeue() {
43:   if (_currentcapacity <= 0)
44:     throw
45:       std::runtime_error(" Can't dequeue from empty ring");
46:   double _hold;
47:   _hold = _buffer[_first];
48:   _first++;
49:   _currentcapacity--;
50:   if (_first == _last) _first = _last = 0;
51:   if (_first == _size) _first = 0;
52:   return _hold;
53: }
54:
55: int16_t RingBuffer::peek() {
56:   if (_currentcapacity == 0)
57:     throw
58:       std::runtime_error("Can't peek on an empty ring");
59:   if (_buffer.empty())
60:     throw
61:       std::runtime_error("Can't peek on an empty vector array");
```

```
   62:   return _buffer[_first];
   63: }
```