```
// Copyright 2022 Anson Cheang

// This is used to make the storage for the buffer.

#include "CircularBuffer.h"
#include <iostream>
#include <string>

CircularBuffer::CircularBuffer(size_t capacity) {
    std::string message =
    "CircularBuffer constructor: capacity must be greater than zero";
    if (capacity < 1) {
        throw std::invalid_argument(message);
    }
    currentSize = 0;
    maxCapacity = capacity;
}

size_t CircularBuffer::size() {
    return list.size();
}

bool CircularBuffer::isEmpty() {
    return list.size() <= 0;
}

bool CircularBuffer::isFull() {
    return list.size() == maxCapacity;
}

void CircularBuffer::enqueue(int16_t x) {
    if (isFull()) {
        throw std::runtime_error("enqueue: can’t enqueue to a full ring s
ize");
    }
    list.push_back(x);
}

int16_t CircularBuffer::dequeue() {
    if (isEmpty()) {
        throw std::runtime_error("dequeue: can’t dequeue an empty ring");
    }
    int16_t val = list.front();
    list.pop_front();
    return val;
}

int16_t CircularBuffer::peek() {
    if (isEmpty()) {
        throw std::runtime_error("peek: can’t peek an empty ring");
    }
    return list.front();
}

unsigned int CircularBuffer::getCap() {
    return maxCapacity;
}
```