Project 3

<u>Due: 5pm, Monday, November 15</u>

For this project, you will be implementing a thread-safe stack data structure. You have been provided with an abstract class called BoundedStack that includes protected fields, which can be used to support an array-based stack implementation with a fixed capacity. You are to implement a class called "SynchronizedBoundedStack" that is a concrete subclass of the abstract class BoundedStack. Your implementation of this BoundedStack class should be thread-safe, meaning:

- If multiple threads call some combination of push() and/or pop() at the same time, there should be no race conditions that cause the stack to become corrupted.
    - You should use the ReentrantLock class to resolve any race conditions.
- Implement all the required conditions
- If a method completes, and it's completion could benefit waiting threads, then the method should invoke the signalAll() method of the appropriate Condition object.
- Design your own producer and consumer classes (threads)

Rubric:

Requirement	Points
Stack implementation works correctly in a single-threaded environment	30
Stack implementation correctly employs locks to resolve race conditions inside of push() and pop().	35
Stack implementation correctly rechecks preconditions inside push(), pop() and peek() using a while loop and a condition.	35
Total	**100**
Bonus (10 random questions from Chap 06 and Chap 09)	10