Problem 1: Producer-Consumer Problem Report

In order to synchronize the bounded buffer through the multiple threads, the semaphores wait() and notifyAll() were employed in BoundedBuffer.java . In the method consume(), which is located in BoundedBuffer.java, before the process enters it's critical section, the entry section checks if the buffer has an open slot. It will continue to wait until there is a slot. After it satisfies the entry condition, then it will enter the critical section where the variable count, which is shared between the producer and consumer threads is updated, then it enters the exit section in which it calls notifyAll() to let other threads know that it has finished executing in its critical section. The same idea was employed for produce(), which is called by the producers.

Simulation time from the 9 runs

	P = 2	**P = 5**	**P = 10**
C = 2	126.064	123.923	123.938
C = 5	123.159	50.686	48.844
C = 10	122.93	49.645	25.454

When reviewing the simulation times above, you can see that as the number of consumers increase, the simulation time decreases, which means that there is better performance. This makes sense because the more number of consumers means that the bounded buffer will have space and there will be less cases where the producer or consumer threads have to wait, which is why the simulation time keeps going down as the number of consumers increase.