CSSE 332 -- Operating Systems

Producer Consumer Problem

Name: Solution Key

Question 1. (5 points) For the problem where we have one produce and one consumer, write down the state of the world or the concurrency state. Make sure to provide a description of each state variable as well as its type and **default value**.

> **Solution:** For the case of only one producer and one consumer, we only need to track if the item is available or not. So use an integer (or a boolean) variable initialized to 0.

Question 2. (5 points) Based on your answer to the first question, write down the *waiting conditions* for each of the producer and the consumer threads. Feel free to write it in pseudo-code to avoid unnecessary clutter due to `pthreads` syntax.

Waiting conditions for the producer thread:

> **Solution:** The producer should enter the waiting state when the item has been produced but has not been consumed yet. The producer would be woken up by the consumer when it has finished consuming the item.

Waiting conditions for the consumer thread:

> **Solution:** The consumer should enter the waiting state when the item is not present (i.e., either the producer has not produced it yet or is currently producing it). It would be awakened by the producer thread when it has finished producing the item.

Question 3. (5 points) Briefly explain the changes that would need to be made to your approach once more consumers are added to the problem.

Solution: Using a single condition variable would create a deadlocked state in the case when a consumer's signal would reach the other sleeping consumer rather than the sleeping producer. To avoid this case, we would need to use two different condition variables or augment our concurrency state (not recommended) to indicate who exactly are we aiming to wake up (it can get pretty messy pretty easily in this case).

Question 4. (5 points) In the space below, write down the condition variables commandements that should always be honored when solving concurrency problems in this class.

Solution:

1. You should always hold the lock when accessing the state of the world.
2. You should always hold the lock before your enter the waiting state. The condition wait statement will release the lock before sleeping.
3. You should not have a condition wait statement that is not protected by a **while** loop, even if you are 100% sure it is not needed. Send Mohammad a bill for the extra processor cycles you pay for in such a case.