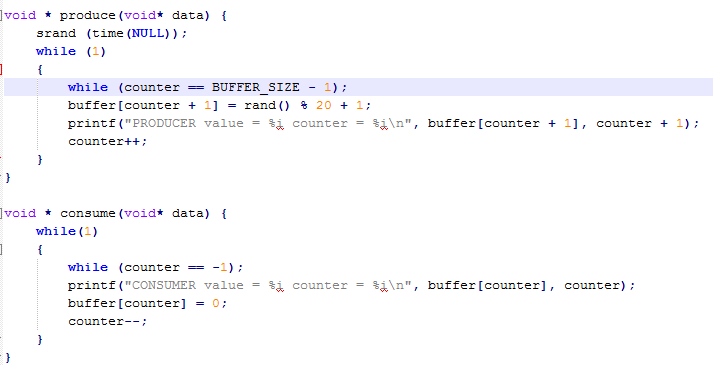
Name: Logan Passi  
Date: 04/05/20

1. Assume that produce and consume are POSIX threads that are started at the same time – what is wrong with this code? – 4pt  
     
     
     
   The counter variable is being being modified by both the producer and the consumer and this will result in the counter becoming out of sync.
2. What is the difference between a mutex and a semaphore and when would you use each? – 2pt  
     
     
   A mutex lock involves a boolean variable whose value indicates whether the lock is available or not. Where as a semaphore is an integer variable that is access through the wait() and signal() functions.

From my understanding a mutex only allows one thread at a time preventing multiple threads to run concurrently . While a semaphore allows a given number of threads to run concurrently but no more than that given number. So as for when to use each, use a mutex for a situation when you only want one thread running at a time and use a semaphore when you need multiple threads running concurrently.

1. What is a deadlock and give a tangible example of a deadlock scenario? – 2pt  
     
   A deadlock is when processes wait indefinitely within the semaphore. An example could be that if one program encounters an error making causing it to get stuck in a loop. This will prevent the other programs after it from running properly.
2. What is busy waiting and why is it bad? If the OS did not support events/pausing a process is there any other way to at least reduce the CPU waste from busy waiting or eliminate it? - 2pt  
     
   Busy Waiting is not ideal because it wastes CPU that could be doing something else..   
   Instead of busy waiting the process can block itself and that will put itself into a waiting queue.
3. What kind of synchronization issues can happen in an ecommerce environment? (Think of the backend database) – 2pt  
     
     
     
   If two people are trying to modify the same content at the same time it would make it become out of sync.
4. If I were to task you with designing your own OS (don’t worry I won’t ) would you choose a preemptive kernel or a nonpreemptive kernel? Why? – 3pt

A preemptive kernel because that is what most modern operating systems use so I would expect there to be more up to date documentation with a preemptive kernel to help me design an OS.

1. List 3 things you could use a FUSE file system for. -3pt  
     
     
   You could make your own file system

Make an ftp program with a fuse file system

You could use it with any program that would require you to need access to view and edit, some type of document or article as if they were files.

Extra credit:

1. Implement a solution to the Dining-Philosophers problem. Your book has some sample code, and I provide code on how to use monitors with ptypes. If you have questions/run into issues contact me.
2. Implement a solution for a bank which allows for depositing and withdrawing. You can get some inspiration from this document: <http://pages.cs.wisc.edu/~remzi/OSTEP/threads-monitors.pdf>

(Your solution must compile )