Computer Project #5

Assignment Overview

This assignment focuses on synchronization of multiple threads within an application. You are to design and implement the functions to complete a C/C++ program to manage a bounded buffer. You will use POSIX threads and semaphores to solve the producer/consumer problem.

It is worth 40 points (4% of course grade) and must be completed no later than 11:59 PM on Thursday, 10/19.

Assignment Deliverables

The deliverables for this assignment are the following files:

proj05.makefile – the makefile which produces "proj05"
proj05.student.c – the source code file for your solution

Be sure to submit your files for grading via the CSE handin system before the project deadline.

Assignment Specifications

1. The program will create P producer threads and C consumer threads which share access to a bounded buffer. The number of producers and consumers will be available to the program as command-line arguments (the first argument will be P and the second argument will be C); those values will not exceed 10, and will default to 1. Note that command-line arguments are arrays of type "char" in C/C++; the function "strtol" can be used to convert a character string into an integer value.

2. The buffer will be circular and will consist of 20 records, where each record contains a thread ID number and an integer sequence number.

3. Each producer will generate the first 100 integers, starting at zero. For each of those integer values, the producer will build a record and insert it into the circular buffer. The record will contain the producer’s thread ID and the integer value (the sequence number).

4. Each consumer will repeatedly retrieve one record from the buffer and display its contents on the standard output stream in a readable format. All consumers will terminate after all of the records have been generated and retrieved.

5. The critical sections of the producer(s) and consumer(s) will be kept as small as possible.

6. The program will function correctly, even though the producer(s) and/or consumer(s) are interrupted inside and/or outside their critical sections. Your program will use function "sched_yield" and random numbers to simulate interruptions. A process will be interrupted approximately 12.5% of the time inside its critical section and approximately 25% of the time outside its critical section.

7. The program will use POSIX semaphores to coordinate the producers and consumers, as well as to guard critical sections.

8. The program will include appropriate error-handling.