**Programming Assignment 5**

**Thread Synchronization**

# Objective

The objective of this assignment is to use semaphores to protect the critical section between two competing threads.

# Assignment: Using Threads and Mutex/Counting Semaphores

The idea is to write a C/C++ program that creates two threads. The first thread is the consumer thread that consumes the data written to a shared memory buffer. The second thread is the producer thread that “produces” the data for the shared memory buffer. In order to prevent a race condition (e.g. the consumer reading before the producer writing) use a mutex semaphore and counting semaphores to coordinate when each thread can safely write or read to/from a common shared memory region.

**Prodcon Program Implementation**

The producer/consumer program (*prodcon.c*) that takes one argument from the command line (no prompting the user from within the program).

1. To start the *prodcon* program

*./****prodcon*** *<nitems>* where the argument *<nitems>* indicates the number of items in the shared buffer. Each item consists of a sequence number (integer), a timestamp (unsigned integer, you can use the ‘time’ command), a checksum (unsigned 16-bit integer, use the internet checksum algorithm) and 22 bytes of random data for a total of a 32-byte item.

1. The main process is to create the shared memory region using POSIX shared memory with size *nitems*, initialize the mutex the counting semaphores and create both the producer and consumer threads.

1. The producer thread is to create the data *item(s)* which includes the item number, timestamp, checksum and random data.
2. The consumer thread is to read the shared memory buffer of *item(s)*, validate the item number (ensure it’s in sequence) then calculate the checksum and compare that with the value stored in the shared memory buffer to ensure that the data did not get corrupted.

**Error Handling**

Perform the necessary error checking to ensure the correct number of command-line parameters. Limit *memsize* to 64K and ensure *ntimes* is a positive integer. If the consumer detects a mismatched checksum it is to report the error along with the expected checksum and the calculated checksum and exit the program.

# Grading

The program will be graded on the basic functionality, error handling and how well the implementation description was followed. Be sure to name your program **prodcon.c** (no extra characters, capitals) Note that documentation and style are worth 10% of the assignment's grade!

# Submission

The source code for program should be available on Canvas along with a README/Output file that provides any documentation and sample output.