WEEK 5 – LAB 4 - UE15CS305 – Introduction to Operating System (IOS) Laboratory Problem Statements with Solutions

Background:

Chapter 3: Read Shared Memory Systems and the producer consumer problem.

Chapter 4: Read the section on pthreads and how to create threads.

Refer man pages for pthread.

Attached tutorial on Mutex

Problem Statement 5.1.

Write a program to implement Producer Consumer problem using Mutex.

Implement a main program that creates two threads: producer and consumer threads which executes producer and consumer functions respectively. The producer should produce an item and update the buffer. The consumer should consume an item and update the buffer. You need to use mutex and enclose the critical sections in both producer and consumer so that only one of them can update the buffer at a time. Also, consumer should wait if buffer is empty and producer should signal when the buffer has at least one item. You can assume it is unbounded buffer so that producer does not have to wait for buffer availability. Both the producer and consumer threads can be infinite loops and each should also randomly sleep to let the other proceed. The output should be the number of items in buffer along with the consumer/producer that is updating the buffer.

Basically, the producer produces goods while the consumer consumes the goods and typically does something with them. In our case our producer will produce an item and place it in a bound-buffer for the consumer. Then the consumer will remove the item from the buffer and print it to the screen. At this point you may be asking yourself what exactly is a “bound-buffer”? Well a buffer is a container of sorts; a bound-buffer is a container with a limit. We have to be very careful in our case that we don’t over fill the buffer or remove something that isn’t there; in c this will produce a segmentation fault.

The Producer/Consumer Problem

This problem is one of the small collections of standard, well-known problems in concurrent programming: a finite-size buffer and two classes of threads, producers and consumers, put items into the buffer (producers) and take items out of the buffer (consumers). A producer must wait until the buffer has space before it can put something in, and a consumer must wait until something is in the buffer before it can take something out. A condition variable represents a queue of threads waiting for some condition to be signalled.