CS 360 Special Topics: Computer Networking

Spring 2007 **DUE: April 6, 2007**
Assignment #4-1

Building a Thread Safe Queue data structure

You will need to implement a thread-safe queue which will be used by Project 4 to allow each component of the system to pass messages to other components. A thread-safe queue should allow multiple threads to add and remove data correctly. The code for the queue must take care of all the thread-safety issues (mutexes and the like). The application that uses the queue should not need to use a mutex directly to protect access to the queue.

The interface you must write to is in Queue_TS.h. You MAY NOT alter this header file in any way. Refer to the comments for each function prototype to determine exactly what each function is to do. You do need to augment the data structure that represents the queue (**`struct Queue,`** defined in Queue_TSData.h) with mutex variables to make this a thread safe queue.

You will need to build TWO test cases to show that your queue is thread safe. These test cases should be multithreaded applications that add and remove data from a queue. As indicated above, your test cases should NOT use any mutexes to protect access to the queue. The thread-safety should be built into the queue. At least one of these test cases must store a struct with two or more data fields (of any type) in the queue.

What and How to Submit

You must submit your thread safe queue (Queue_TS.c, Queue_TS.h, Queue_TSData,h, Makefile, QueueTest1.c, QueueTest2.c) electronically in **`PUNetID_cs360s07_PA4_1.tar.gz.`**

The Makefile should build your test cases as `tsqTest1` and `tsqTest2`. The names of the targets for these two files should be tsqTest1 and tsqTest2, respectively. The Makefile should also build both test cases by default if no command line arguments are given to the `(g)make` command. The test cases should not need any command line argument to run.

Both the electronic and paper submissions must be made by 6pm, April 6th. *Please do not create a directory in your .tar.gz file!*

Hints

- In a complex project like this, you should commit to Subversion often! Name your project **PUNetID_cs360s07_PA4** in your Subversion repository so that I will be able to look in your repository and review your commits and commit messages. **This will be a graded portion of the assignment**.

- queueDequeue() takes a parameter *timeout* which controls if the function blocks when the queue is empty (and for how long) or immediately returns NULL. We will discuss the relevant pthreads constructs to implement this in class.