Programming Assignment 3
Spring, 2016

Purpose:
The goals of this assignment are (1) to learn how to use the pthreads library to manage threads, (2) to understand how to create a multi-threaded server, (3) to understand how and where to use basic thread synchronization, and (4) to gain practice with client software that can issue multiple requests over a socket.

Part 1: Assignment Description

You are to develop a multithreaded server so that it creates a thread pool with N threads. When a new client request arrives, the server should add it to a queue of connections. The threads should remove one connection from the queue and handle the request. If the connection queue does not have a connection available, the thread should “sleep” until a connection is ready. The connection queue is shared among all threads and thus requires synchronization to protect access to the connection queue.

The processing to be done on behalf of the client is to take a number sent from the client, multiply it by 10 and return the result back to the client.

Your program should not use busy waiting.

Programming Language: C

Server Command Line Input: Port number that will be used to listen on, number of threads, size of connection array

Client Command Line Input: Server’s host name, port number and the number to be multiplied

Part 2: Hints

Do not assume that just because you run your program a few times that it is actually correct. In order to see race conditions, you will need to ensure that your threads run at roughly the same time. Sometimes if you don’t do anything special, it’s possible for a thread to complete all operations on shared data before being context switched out.

One tool you can use is a barrier. You may want to consider using pthread_barrier_init(....) and pthread_barrier_wait(....) to test your program. Once testing is done take these out. You want to use these in parts of code that have operations on shared data.