CSE3321.3
Assignment 2.

Using Threads.

Due Date: Mon. Nov. 7 12:00noon.

1. POSIX threads

In all the following questions include a short answer and on which man page you found the information.

(1) What is the naming conventions for pthread function calls?

(2) Enumerate the synchronization mechanisms for POSIX threads?

(3) Why is it a good programming practice to call `pthread_cond_wait` from within a loop?

(4) Which thread executes after a thread is woken up from a condition variable.

(5) What happens if a thread blocks on a condition variable while holding two mutexes.

(6) What is a detacched thread.

2. Probabilistic Simulation of a Client-Server system

Write a program that does a probabilistic simulation of a client server system. The clients generate job packages at random times and place them on the global queue. Every package takes a random amount of time to execute and if the servers are busy the package stays on the queue. At every time interval a the client and the server are woken up to simulate work done during this interval.

The servers need one parameter, the completion rate μ (mu), also called death rate, of the currently executing process. At every activatin of the server, if there is a job executing, the server generates a random number between zero and one and compares it to μ. If the random number is smaller than μ the job execution is terminated. If there is no job executing, the server gets a job from the global queue if available. The state of the server can be either FREE or BUSY.

The clients need one parameter as well, the birth rate λ (lambda) of the jobs. Again, at every activation the client generates a random number between zero and one and compares it to λ. If the random number is smaller than λ a new job is generated and placed on the global queue.

The program has to have a simple queue (really simple, it just keeps track of the number of items in it and nothing else). It also needs to keep a few statistics (like number of jobs generated, total number of jobs in the queue, number of clock ticks so far, etc) from which to compute useful statistics.

The program prints the following job statistics, one per line, properly annotated:

(1) Average waiting time (AWT)

(2) Average execution time (AXT)

(3) Average turnaround time (ATA)

(4) Average queue length (AQL)

(5) Average interarrival time. (AIA)

The command line options to the program are the --lambda [0.005], --mu [0.01], --servers[2], --clients[2].

Yout program will use pthreads. Every server, and every client will be separate threads. In addition there is one thread for the clock. At every tick of the clock thread all other threads get activated, do whatever they need to do and then wait for the next tick. Whenever the clock sees that all the threads have completed their work for the current tick, it emits the next tick. The program runs for 1000 ticks.