CSC 456 Operating Systems
Dr. Christer Karlsson
Benjamin Kaiser
Programming Assignment 1
Part 2 Sleeping TA

Description of Program

This program is a simulation of a TA helping his/her students. The requirements of the program are that we used POSIX threads to represent the students and the TA. We also are required to use both semaphores and mutexes.

Description of Algorithms and Libraries used

The two special libraries that are used are the pthread.h library and the sempahore.h library. The pthread.h library allows us to use POSIX threads and their mutex locks. The semaphore.h library allows us to create and utilize the semaphores.

A description of the algorithm is given in the section on functions and program structure.

Description of Functions and Program Structure

This function was a procedurally designed program with multiple different functions. The main function is the obviously the main function. The function calls an initialization function calls init and this function initializes the mutex lock and then three semaphores. The first semaphore is called sleeping and this tells whether or not the TA is sleeping or not. The next one is called freetogo and it talls the student threads to know when they can leave the office and start hanging out again. The final semaphore is whether or not the chair in the office is available. The init function also sets the student ids for each of the students. After this function is called, the create_ta function gets called. All it does is call pthread_create and then starts the thread into the ta_loop function. The next function that is called is the create_students function. All this function does is loop through the number of student and creates a thread for each one and then starts the the tread into the student loop. The TA loop function simply loops while true so pretty much infinitely. It starts out by computing a random time for the ta to “sleep” which is essentially the time he can help a student. He waits for a student to show up by going to sleep which is waiting on the sleeping semaphore. When he gets signaled by the students, he wakes up, locks around the students_waiting variable which tells how many students are in the waiting room and decrements. He then signals that the office chair is available or in other words, he is able to help a student. He decrements the number of students in the waiting room and then he goes into helping the student. The help_student function then prints out that he is helping a student, then sleeps the thread for the random times computer in the ta_loop. He then signals that the student is free to go by posting to the freetogo semaphore.

The student_loop starts out by calculating a random time to sleep or in other words for the student to “hang out”. Then he comes into the waiting room and locks the door to see if he can sit down. He tells the TA that he is there and sits down then unlocks the door so other people

can get in and increment the number of students in the waiting room. He then waits on the TA to open the office door and get the chair ready etc. Then they wait until the TA is done talking to them.

Description of Testing and Verification Process

The way I tested this program was to run it several different times and examine the output by hand. I did this several times to ensure that all of the data integrity is there. I did notice that the output statements were a little off but I think it was because the print is not 100% thread safe and then one thread is faster than the others.

Data

There was not really any data collected for this project. The only data that the program produces is its output which can be redirected to a file.

Analysis of Data

The only analysis of the data that I did was a manual observation of the data.

Description of Submission

The contents of the submission of this program are as follows:

- hangout.c -- This file contains the function that the student calls to “hang out and mess around in the lab”
- student.c -- This file contains the function which is a loop that each of the student threads calls and starts with.
- ta.c -- This file contains the function which is a loop that the TA calls to do its things.
- ta.h -- This file simply contains the declarations for the semaphores and mutexes and all the other variables that the ta and students
- main.c -- This file contains the main function and all the code to operate the ta and the student threads.