Alex Tavares, Leo Gonsalves

CS3013 – Operating Systems

2/14/2018

Project 3 Write-Up

Using the program:

First make your way to the directory where the project is stored.

Once there run the command: make all

This builds the executable: bathroomSim

To run the program simply run the command:

./bathroomSim nUsers meanLoopCount meanArrival meanStay (optional)seed

Where:

- **nUsers** is the number of people in the simulation
- **meanLoopCount** is the average number of times that a user repeats the bathroom usage loop
- **meanArrival** is the mean of the arrival intervals
- **meanStay** is the average length of time that a person stays in the bathroom, once he or she has entered
- **seed** is an optional argument that sets the seed for the random number generator

Recommended set of parameters: ./bathroomSim 50 10 500 400 3

Controlling Access:

To ensure synchronization among threads we used a monitor with the pthread library. When a person calls the *Enter* function they grab the bathroom lock, check if the bathroom is occupied by a member of the opposite gender, and if it is, they wait on the conditional variable *vacant*. If the bathroom is vacant or occupied by a member of the same gender the person can go into the bathroom and if they are the first person in the bathroom they set the gender of the bathroom. Then when the last person leaves the bathroom they call *pthread_cond_broadcast()* which unblocks all threads currently blocked on the conditional variable *vacant*, which consists of all threads currently in the queue. Threads in the queue are woken up by the broadcast, grab the bathroom lock, decrement the queue length, and then recheck if the bathroom is occupied by a member of the opposite gender. If it is, the are then

added back to the queue, but if it's not then they enter the bathroom. To ensure that members of the opposite gender do not enter the bathroom when it is occupied, we make sure that every individual grabs the lock before they check the if the bathroom is occupied by a member of the opposite gender and only the first person to enter and last person to leave set the current gender of the bathroom. Also, when a thread is woken up it always has to check the condition before it can enter the bathroom and we did this by wrapping the *pthread_cond_wait()* function in a while loop.

Multithreaded Test Program:

The multithreaded test program works by first taking the arguments from the command line. Then it calls the function *Initialize()*, which initializes the values of the bathroom object bathroom(which is a global variable). Then it creates two threads, one that keeps track of the amount of time the bathroom is occupied and the amount of time it is vacant and one that keeps track of the amount of time that there is a queue (we use this to calculate the average queue length). Then using a for-loop we create *nUsers* number of threads which execute the function Individual. This function executes the *Enter()* and *Leave()* functions which simulate enter and leaving the bathroom. When the desired number of loops are run, the thread must then grab the *print_lock* before it can print its statistics, ensuring that no two threads print at the same time. The main function waits for all threads to finish by using a for-loop that calls *pthread_join* for all the thread's thread ID's. Then it grabs the bathroom lock, sets the flag inside of the bathroom object to 0 and signals both the timer thread and the queue thread, which causes those threads to exit. Then the main function calls pthread_join on both of those threads. Finally, the function *Finalize()* is called and the statistics for the bathroom are printed and the program exits.

Program Files:

Makefile:

Contains instructions to build program.

main.c:

Contains the code which takes user input, creates all the threads, and prints the statistics.

bathroom.h:

Contains the bathroom_Object struct and the gender data type. It also contains the function headers for the Enter, Leave, Initialize, Finalize, Time_Keeper, and Queue_Time_Kepper functions.

bathroom.c:

Contains the code fore the functions Enter, Leave, Initialize, Finalize, Time_Keeper, and Queue_Time_Kepper.

person.h:

Contains the Person_object struct and the function headers for genTime, genLoops, genGender, and Individual.

person.c:

Contains the code for rhe functions genTime, genLoops, genGender, and Individual.