CS4323-Spring2022 GroupD

GitHub

Members:

- Lucas Stott lstott@okstate.edu
- Nathan Bales nathan.bales@okstate.edu
- Drew Nguyen drew.nguyen@okstate.edu
- Daniel Albrecht daniel.albrecht@okstate.edu

Project Purpose:

1. Learn Mutex/Semaphore Synchonization and Scheduling
2. Build ThreadPooling Methods
3. Collorate within Team
4. Understand Deadlock/Starvation Problems for (Consumer/Producer)

Running the Program

> gcc main.c -o main -lpthread && ./main 2 100 3 3 10 100 Arguments (2, 100, 3, 3, 10 , 100)
>
> 1. Number Medical Proffesions (Nm)
> 2. Number of Patients (Np)
> 3. Waiting Room Capacity (Nw)
> 4. Number of Sofas (Ns)
> 5. Maximum Arrival Time Between Patients (ms)
> 6. Check-up Duration (ms)

Thread Pools

- [x] PatientQueue
- [x] MedicalProfessionalsQueue

Functions Patient

- [x] enterWaitingRoom()
- [x] sitOnSofa()
- [x] getMedicalCheckup()
- [x] makePayment()
- [x] leaveClinic()

Functions Doctor

- [x] waitForPatients()
- [x] performMedicalCheckup()
- [x] acceptPayment()

Tasks Lucas

- [x] Create Main Function for Args()
- [x] Create Independent Functions
- [x] Start Performing Synchronization

Tasks Nathan

- [x] Concurrency Threading
- [x] Performing Analysis
- [x] Syncing Global Variables with Threads

Tasks Daniel

- [x] Semaphore/Mutex Lock Implementing
- [x] Sync Doctor/Patients
- [x] Structuring Queue's

Tasks Drew

- [x] Validating Threads Generation
- [x] Gracefully Exit Threads
- [x] Testing Shared Memory Exposure

Potential Errors

1. Passing Patient Information to Doctor
2. Possible Variable Overwriting on Globals

Fixed

1. Patient and Doctor Threads Exited