Noel Romero

NXR 170030

CS 4347.001

Dr. Ozbirn

Project 2

The purpose of this project is to put us students to utilize the knowledge gained from lectures about semaphores. In this project we use semaphores to enforce mutual exclusion and keep concurrency amongst action being performed to simulate the event of a patient going to the Doctor to receive consultation. There are four key characters in this model Receptionist, Doctor, Nurse and Patient. In our Doctor simulation each character is brought to life using a thread. First in this program's Main class we prompt the user to enter the number of patients (not exceeding 30) and the number of doctors (not exceeding 3) that this office will have. In addition to this, in the main class we start all the Patient, Doctor and Nurse threads and the receptionist thread and declare all needed data structures such as queues to represent the line of patients that wish to meet the doctor or the map to represent the linking of the nurse and the doctor and here in the main class we also initiate all the threads . After creating the needed infrastructure in the main class, the flow of this doctor's office begins with the patient class. The patient start off by entering the waiting room and getting its object reference stored in a queue which the receptionist thread will get the patient object and signal it through the use of semaphores to proceed to sit in the waiting room. A nurse then gets this registered patient object and signals the patient object to proceed and insert its reference into a map with the key matching the nurse object. Once this is completed the patient then signals to the doctor thread that its available the Doctor thread now accesses the mutual exclusive map to access the patient and at the same time remove it from the map and then perform its services on the patient. Once the doctor is

complete the doctor thread signals the patient thread to proceed with the use of semaphores. The patient outputs to the console that it received the doctors consultation and proceeds to leave the office. That specific thread is now completed and it is joined in the main class.

In this project the most difficult part was to figure out a way to keep the nurse thread and doctor thread linked and I found using a maplike data structure would be the perfect tool for this citation. I learned exactly what kind of behavior occurs when buffers are not mutually exclusive and when they are through trial and error. This was an interesting project to see how semaphores can be used to block multiple threads to create a form of synchronization that mimics a real world set of actions such as a doctor's office. Through effort and time I was able to successfully complete the project and create concurrency amongs four threads all while maintaining mutual exclusion.

Enter the amount of Doctors

3

Enter the quantity of patience you want

8

Run with 8 patients, 3 nurses, 3 doctor

Patient 0 enters waiting room, waits for receptionist

Patient 1 enters waiting room, waits for receptionist

Patient 2 enters waiting room, waits for receptionist

Patient 3 enters waiting room, waits for receptionist

Patient 4 enters waiting room, waits for receptionist

Patient 5 enters waiting room, waits for receptionist

Patient 7 enters waiting room, waits for receptionist

Patient 6 enters waiting room, waits for receptionist

Receptionist registers patient 0

Patient 0 leaves receptionist and sits in waiting room

Receptionist registers patient 1

Nurse 0 takes patient 0 to the doctors office

Receptionist registers patient 2

Patient 1 leaves receptionist and sits in waiting room

Patient 2 leaves receptionist and sits in waiting room

Patient 0 enters doctor 0's office

Receptionist registers patient 3

Nurse 1 takes patient 1 to the doctors office

Receptionist registers patient 4

Patient 3 leaves receptionist and sits in waiting room

Doctor 0 listens to Symptoms from patient 0

Nurse 2 takes patient 2 to the doctors office

Patient 0 receives advice from doctor 0

Nurse 0 takes patient 3 to the doctors office

Receptionist registers patient 5

Patient 4 leaves receptionist and sits in waiting room

Patient 1 enters doctor 1's office

Nurse 1 takes patient 4 to the doctors office

Receptionist registers patient 7

Patient 5 leaves receptionist and sits in waiting room

Patient 3 enters doctor 0's office

Patient 0 leaves

Patient 2 enters doctor 2's office

Doctor 1 listens to Symptoms from patient 1

Doctor 2 listens to Symptoms from patient 2

Patient 1 receives advice from doctor 1

Patient 1 leaves

Doctor 0 listens to Symptoms from patient 3

Receptionist registers patient 6

Patient 4 enters doctor 1's office

Nurse 2 takes patient 5 to the doctors office

Patient 7 leaves receptionist and sits in waiting room

Patient 5 enters doctor 2's office

Doctor 1 listens to Symptoms from patient 4

Patient 6 leaves receptionist and sits in waiting room

Patient 3 receives advice from doctor 0

Patient 3 leaves

Patient 2 receives advice from doctor 2

Patient 2 leaves

Nurse 1 takes patient 6 to the doctors office

Patient 4 receives advice from doctor 1

Patient 4 leaves

Doctor 2 listens to Symptoms from patient 5

Nurse 0 takes patient 7 to the doctors office

Patient 5 receives advice from doctor 2

Patient 5 leaves

Patient 6 enters doctor 1's office

Patient 7 enters doctor 0's office

Doctor 0 listens to Symptoms from patient 7

Doctor 1 listens to Symptoms from patient 6

Patient 7 receives advice from doctor 0

Patient 7 leaves

Patient 6 receives advice from doctor 1

Patient 6 leaves

Process finished with exit code 0