Name: Rushik Guduru
NetID: rxg200049
Section: CS 4348.501

Pseudocode for Doctors Office

```
/* program DoctorsOffice */

// Define the maximum capacities and initial semaphore values
semaphore maxClinicCapacity = new Semaphore(20);
semaphore waitingRoom = new Semaphore(15);
semaphore registrationDesk = new Semaphore(1);
semaphore doctorOffices[] = {new Semaphore(1), new Semaphore(1), new Semaphore(1)};
semaphore nurseAvailable[] = {new Semaphore(0), new Semaphore(0), new Semaphore(0)};
semaphore patientDone = new Semaphore(0);
int patientCount = 0; // counter for patients
int doctorCount = 3; // this could be dynamic based on input
int nurseCount = doctorCount; // one nurse per doctor
int[] patientId; // patient identifier array

void patient(int id) {
   semWait(maxClinicCapacity);
   enterClinic(id);
   semWait(registrationDesk);
   registerAtReceptionist(id);
   semSignal(registrationDesk);
   semWait(waitingRoom);
   sitInWaitingRoom(id);
   int assignedDoctor = getAssignedDoctor(); // function to assign a doctor randomly
   semWait(doctorOffices[assignedDoctor]);
   semSignal(waitingRoom);
   goToDoctorOffice(id, assignedDoctor);
   semSignal(nurseAvailable[assignedDoctor]);
   semWait(patientDone);
   leaveClinic(id);
   semSignal(maxClinicCapacity);
}

void doctor(int id) {
   while (true) {
      semWait(nurseAvailable[id]);
      seePatient(id);
      giveAdvice(id);
      semSignal(patientDone);
```

```
    }
}

void nurse(int id) {
    while (true) {
        semWait(nurseAvailable[id]);
        callPatient(id);
        directPatientToOffice(id);
        semSignal(doctorOffices[id]);
    }
}


// Receptionist thread
void receptionist() {
   while (true) {
       semWait(registrationDesk); // Wait for a patient to come to the registration desk
       int patientId = registerNextPatient(); // Method to register the next patient
       semSignal(registrationDesk); // Signal that the registration desk is available again
       if (patientId != -1) { // If a valid patient was registered
           semSignal(waitingRoom); // Signal that a patient is waiting in the waiting room
       }
   }
}


void main() {
    // Starting threads for patients, doctors, nurses, and receptionist based on the input
// Parse command-line arguments to set the number of doctors and patients
    doctorCount = Integer.parseInt(args[0]);
    nurseCount = doctorCount; // Assuming one nurse per doctor
    int patientNumber = Integer.parseInt(args[1]);

    // Initialize the patientId array
    patientId = new int[patientNumber];

    // Initialize semaphores for doctors' offices
    for (int i = 0; i < doctorCount; i++) {
        doctorOffices[i] = new Semaphore(1);
        nurseAvailable[i] = new Semaphore(0);
    }

    // Start the threads
    Thread receptionistThread = new Thread(new Receptionist());
    receptionistThread.start();

//Next create and start patient nurse receptionist and doctor threads

}
```