Minjae Chae

mxc210045

CS 4348.001

Project 2 – Design

(1) a list of every semaphore, its purpose, and its initial value

1. `mutex1`: Semaphore to count patient number one by one. Initial value of 1.

2. `mutex2`: Semaphore to ensure patient numbers are delivered one by one between patients and the receptionist. Initial value of 1.

3. `mutex3`: Semaphore to ensure patient numbers are delivered one by one between the receptionist and the nurse. Initial value of 1.

4. `mutex4`: Semaphore to ensure patient numbers are delivered one by one between the nurse and the doctor. Initial value of 1.

5. `mutex5`: Semaphore to ensure doctor numbers are delivered one by one between the nurse and the patient. Initial value of 1.

6. `receptionist_ready`: Semaphore to control the receptionist's readiness. Initial value of 1.

7. `receptionist_call`: Semaphore for the receptionist to wait for patient calls. Initial value of 0.

8. `nurse_call`: Semaphore for the nurse to wait for the receptionist's notification of patient waiting. Initial value of 0.

9. `patient_exit`: Semaphore for controlling patient exits. Initial value of 1.

10. `patient_waitingroom[MAX_CAPACITY]`: Array of semaphores, each controlling access to if patient is in patient_waitingroom. All initialized with a value of 0.

11. `patient_office[MAX_CAPACITY]`: Array of semaphores, each controlling access to if patient arrived to office. All initialized with a value of 0.

12. `receptionist_register[MAX_CAPACITY]`: Array of semaphores, each controlling the registration process of patients with the receptionist. All initialized with a value of 0.

13. `office[MAX_DOCTORS]`: Array of semaphores, each controlling the availability of the doctor's office. All initialized with a value of 1.

14. `nurse_ready[MAX_DOCTORS]`: Array of semaphores, each controlling the readiness of the nurses. All initialized with a value of 1.

15. `nurse_directs[MAX_CAPACITY]`: Array of semaphores, each controlling the direction of patients by nurses to the doctor's office. All initialized with a value of 0.

16. `doctor_ready[MAX_DOCTORS]`: Array of semaphores, each controlling the readiness of the doctors. All initialized with a value of 1.

17. `doctor_notified[MAX_DOCTORS]`: Array of semaphores, each controlling the notification of doctors about patients waiting. All initialized with a value of 0.

18. `doctor_advice[MAX_CAPACITY]`: Array of semaphores, each controlling the advice given by doctors to patients. All initialized with a value of 0.

(2) pseudocode for each function. The pseudocode should be similar to the pseudocode shown in the textbook for the barbershop problem. Every wait and signal call must be included in the pseudocode.

```
int patient_count = -1;

int doctor_count;

int total_patients;

Queue queue1, queue2, queue3, queue4;

Queue doctor_queues[MAX_DOCTORS];


Semaphore

mutex1 = 1, mutex2 = 1, mutex3 = 1, mutex4 = 1, mutex5 = 1;

receptionist_ready = 1, receptionist_call = 0, nurse_call = 0, patient_exit = 1;

patient_waitingroom[MAX_CAPACITY] = {0}, patient_office[MAX_CAPACITY] = {0};

receptionist_register[MAX_CAPACITY] = {0}, office[MAX_DOCTORS] = {1};

nurse_ready[MAX_DOCTORS] = {1}, nurse_directs[MAX_CAPACITY] = {0};

doctor_ready[MAX_DOCTORS] = {1}, doctor_notified[MAX_DOCTORS] = {0},
doctor_advice[MAX_CAPACITY] = {0};


function patient():

    int custnr, officenr;

    wait(mutex1)
    patient_count++
    custnr = patient_count
    signal(mutex1)
    enter();
    wait(mutex2)
    enqueue(queue1, custnr)
    signal(receptionist_call)
    signal(mutex2)
    wait(receptionist_register[custnr])
    enter_waitingroom();
    signal(patient_waitingroom[custnr])
    wait(nurse_directs[custnr])
    wait(mutex5)
    dequeue(queue4, officenr)
    signal(mutex5)
    enter_office();
    signal(patient_office[custnr])
    wait(doctor_advice[custnr])
    get_advice();
    signal(office[officenr])
    wait(patient_exit)
    exit();
    signal(patient_exit)

function receptionist():
    int i
    while true:
        wait(receptionist_ready)
        wait(receptionist_call)
        wait(mutex2)
        dequeue(queue1, i)
        signal(mutex2)
        signal(receptionist_register[i])
        wait(mutex3)
        enqueue(queue2, i)
        signal(nurse_call)
        signal(mutex3)
        signal(receptionist_ready)

function nurse(n_id):
    int j
    while true:
        wait(nurse_ready[n_id])
        wait(nurse_call)
        wait(mutex3)
        dequeue(queue2, j)
        signal(mutex3)
        wait(patient_waitingroom[j])
        wait(office[n_id])
        wait(mutex5)
        enqueue(queue4, n_id)
        direct();
        signal(nurse_directs[j])
        signal(mutex5)
        wait(mutex4)
        enqueue(doctor_queues[n_id], j)
        signal(doctor_notified[n_id])
        signal(mutex4)
        signal(nurse_ready[n_id])

function doctor(d_id):
    int k
    while true:
        wait(doctor_ready[d_id])
        wait(doctor_notified[d_id])
        wait(mutex4)
        dequeue(doctor_queues[d_id], k)
        signal(mutex4)
        wait(patient_office[k])
        listen_symptom();
        signal(doctor_advice[k])
        signal(doctor_ready[d_id])
```