#define CHAIRS 5 /\* # chairs for waiting students \*/

typedef int semaphore; /\* declare semaphores\*/

semaphore studentAvailable = 0; /\* # of student waiting for help\*/

semaphore taReady = 0; /\* ta waiting for student\*/

Semaphore waitingRoom = 1; /\* for mutual exclusion of waiting room access\*/

int studentsInWaitingRoom = 0; /\* number of students waiting\*/

void taRoutine(void){

while (TRUE){

wait (&studentAvalable); /\* go to sleep if # of students is 0 \*/

wait (&waitingRoom); /\* acquire access to "waiting room \*/

studentsInWaitingRoom = studentsInWaitingRoom -1;

/\* decrement count of waiting students \*/

/\*ta is helping student\*/

signal (&taReady); /\* ta is ready for next student\*/

signal (&waitingRoom); /\* release access to 'waiting’ room \*/

}

}

void studentRoutine(void){

wait (&waitingRoom); /\* enter critical region i.e access waiting room \*/

if (waiting < CHAIRS) { /\* if there are no free chairs, leave \*/

studentsInWaitingRoom = studentsInWaitingRoom + 1;

/\* increment count of waiting students\*/

signal (&studentAvailable); /\* wake up ta if necessary \*/

signal (&waitingRoom); /\* release access to 'waiting' \*/

wait (&taReady); /\* go to sleep if # of students waiting is 0 \*/

}

else{

signal (&waitingRoom); /\* room is full; do not wait \*/

}

}