Devin Wiley  
dxw112230

Project 2

November 11, 2013

Design

semaphore infoDeskLine = 10;

//ensures only 10 customers are in the info. Desk line

semaphore agentLine = 10;

//ensures only 10 customers are in the line to the agents

semaphore custInAgentLine = 0;

//number of customers in agent line, agent cant run when line is empty

semaphore custInInfoLine = 0;

//number of customers in info desk line, info desk cant run when line is empty

semaphore agentQueueMutex = 1;

//ensures that only one thread can access the agent line queue at a time

semaphore infoQueueMutex = 1;

//ensures that only one thread can access the info desk line queue at a time

semaphore servicedMutix = 1;

//ensures that only one thread can access the serviced variable at once

semaphore assignedMutix = 1;

//ensures only one thread can access the assigned variable at once

semaphore announce = 0;

//signals that a number has been assigned, keep announcer from announcing

//numbers that havn’t been assigned to customers yet

semaphore servicedMutex = 1;

//ensures that only one thread can access the serviced variable

Semaphore assigned = 0;

//Customer waits at information desk until assigned a number

semaphore customerWait[100] = {0};

//semaphore for each customer to cause individuals to wait

semaphore agentWait[4] = {1};

//semaphore for each agent to cause individuals to wait

public Customer(){

created();

wait(infoDeskLine);

enter();

wait(infoQueueMutex); //mutex on queue

enqueue(customer);

signal(infoQueueMutex);

signal(custInInfoLine); //signal line has occupant

wait(assigned); //wait until assigned a number

signal(infoDeskLine); //signal open spot in line

wait(customerWait[]); //wait until number is called

wait(agentQueueMutex); //mutex on queue

enqueue(customer);

signal(agentQueueMutex);

signal(custInAgentLine); //signal that agent line has occupant

wait(customerWait[]); //wait for agent to service customer

announceAgent();

signal(agentWait[]); //signal agents turn to issue photo and eye exam

wait(customerWait[]);

takeExam();

signal(agentWait[]); //signal agents turn to issue licenses

wait(customerWait[]);

getLicense();

signal(agentWait[]); //signal that customer has left

}

public DMV{

//create all the threads

initAgents();

initInfoDesk();

initAnnouncer();

initCustomers();

//join all the threads

joinCustomer();

joinInfoDesk();

joinAnnouncer();

joinAgents();

finished();

}

public Announcer {

wait(announce); //wait until a number has been assigned

wait(agentLine); //ensure that less than 10 people are in line

announceNumber();

signal(customerWait[]); //signal that customer has been called

}

public InformationDesk{

wait(assignedMutex);

while (++assigned <= CUSTOMERS){

signal(assignedMutex);

infoDeskTask();

wait(assignedMutex);

}

signal(assignedMutex); //everyone has been assigned

infoDeskTask(){

wait(custInInfoLine); //wait for occupant in line

wait(infoQueueMutex); //mutex queue

dequeue(customer);

assignNumber();

signal(assigned); //signal customer of assigned value

signal(infoQueueMutex);

signal(announce)

}

}

public agentTask{

wait(custInAgentLine); //wait for occupant in line

wait(agentQueueMutex); //mutex queue

dequeue(customer);

signal(agentQueueMutex);

signal(agentLine); //get customer from the line

announceServing();

signal(customerWait[]); //signal customer to be serviced

wait(agentWait[]);

issueExam();

signal(customerWait[]) //signal customer to take exam

wait(agentWait[])

issueLicense();

signal(customerWait[]); //signal customer to leave

wait(agentWait[]); //signal customer left

}