Ali Hilal

Saturday April 30, 2022

Project 2 Design and Summary

**Design**

static Semaphore mutexBOAQueue = new Semaphore(1);

* Mutual exclusion queue for the box office agent

static Semaphore mutexTTQueue = new Semaphore(1);

* Mutual exclusion queue for the ticket taker

static Semaphore mutexCSWQueue = new Semaphore(1);

* Mutual exclusion queue for the concession stand worker

static Semaphore mutexMovie = new Semaphore(1);

* Used to store movies and the number of seats

static Semaphore readyBOACustomer = new Semaphore(0);

* The line for the box office agent

static Semaphore readyTTCustomer = new Semaphore(0);

* The line for the ticket taker

static Semaphore readyCSWCustomer = new Semaphore(0);

* The line for the concession stand

public static class Movie{

Simple Movie class that will hold the name, ID number, and number of seats available for a single movie. Methods will include simple getName, getID, and getSeatsAvailable, as well as a removeSeat method the decrements the number of seats by 1.

}

public class Customer extends Thread{

Customer class that extends thread. Has ID, movieID, and order variables. Initialized with ID and a random movie ID number.

@Override

public void run(){

Takes thread and adds it to queue. If the movie is available, customer thread is added to ticket taker queue. If customer is to visit concession stand, their order is taken, and customer is added to consession stand queue. After these processes, the customer enters the th

private void orderCST(){

randomly chooses between ordering a popcorn, soda, or both

}

private boolean visitCS(){

randomly chooses if the customer will go to the concession stand or not

}

public class BoxOfficeAgent extends Thread{

Box office agent class that holds ID, the customers ID, and the name of the movie

@Override

public void run(){

when a customer is ready, they are added to the box office agent queue. The process then begins, waiting 1.5 seconds for the box office agent to sell the ticket from the customer.

private void beginBOAProcess(){

sleeps for 1.5 seconds before selling ticket to customer

private void removeBOA(){

removes customer from the box office line.

}

private void movieAvailable(){

if there are seats available, assign seat to customer, call remove seat function from the movie, and return true, else return false.

}

public class TicketTaker extends Thread{

creates ticket taker thread, with customerID being passed when object initialized.

@Override

public void run(){

when a customer is ready, they are added to the ticket taker agent queue. The process then begins, waiting .25 seconds for the ticket taker to take the ticket from the customer.

private void beginProcess(){

when the process starts, process sleeps for .25 seconds when the ticket taker takes the ticket from the customer.

private void removeTT(){

remove customer from ticket taker line.

}

}

public class ConcessionStandWorker extends Thread{

creates concession stand thread, with customerID and order being passed when object initialized.

@Override

public void run(){

when a customer is ready, they are added to the concession stand agent queue. The process then begins, waiting 3 seconds for the concession stand worker to take the order from the customer and them the food.

}

private void beginProcess(){

when the process starts, process sleeps for 3 seconds when the concession stand worker takes the order from the customer and gives the food.

private void removeCSW(){

remove customer from concession stand line.

}

public static void main(String[] args) throws FileNotFoundException {

The function starts with taking the first argument from the terminal. That argument is the file from which the movie lists are. Read in the movies and the number of seats available and add to arraylist of movie objects.

File movieInput = new File(args[0]);

Scanner movieSCN = new Scanner(movieInput);

int count = 0;

create 2 box office agnet threads and start each of them.

Make array threads of 50 customers, add each customer thread to array, start each process.

Create ticket taker thread and start process.

Create concession stand worker thread and start process.

At the end, join all threads and exit the program.

**Summary**

This project was very fun to work on. I learned a lot of interesting concepts. I found the use of semaphore and thread essential.