CSCI 4060U – Laboratory #6
Programming with Pthreads in C
Lab Due: 11:00pm Monday, March 10, 2025 (Canvas)

Introduction

The main purpose of this lab is to create a program that simulates airline ticket agents. Your program will be written in C and use the `pthread.h` library (including a `mutex` to limit access to shared data).

Activity #1

In this activity, you will create a ticket agent simulator program in C. Your program should accept three command-line arguments:

- Number of agents (e.g., 5)
- Number of seats on airplane (e.g., 100)
- Percentage of overselling allowed (e.g., 10%)

Your program should be run at the command-line as:

```
./ticket-simulator 5 100 10
```

First, your program should calculate the total number of tickets available for sale (`tickets_available`) by adding the allowed number of over sale tickets to the number of airplane seats. Your program should also have a shared variable called `tickets_sold` which is initialized to zero.

Ticket agents will be simulated in your program using threads. Your program should create an array of ticket agent threads that will sell tickets before joining back to the main thread. Each thread will execute a routine called `ticket_agent` that will accept as a parameter a structure containing the thread/agent id and the total number of available tickets for sale. Each thread's `ticket_agent` routine should complete ticket sale transactions until `tickets_sold == tickets_available`. Each transaction should:

- Successfully complete a transaction 30% of the time for odd-numbered threads and 45% of the time for even-numbered threads
- Each successful transaction will randomly sell 1 to 4 tickets and update the shared variable `tickets_sold` (remember to use a `mutex` to ensure only one thread accesses this variable at a time)
- Once a transaction has completed, the thread should print either:

  ```
  Ticket agent <X>: Unsuccessful transaction
  ```

  ```
  Ticket agent <X>: Successful transaction – <Y> tickets sold
  ```

Once all of the tickets are sold, the main threads should print out a summary of the final ticket sales.

Activity #2

In the second activity you should focus your attention on the program's performance. Specifically, you need to assess how long (in seconds) it takes to run the program with the following parameters:

```
./ticket-simulator 5 100 10
./ticket-simulator 5 2500 25
./ticket-simulator 50 2500 25
```

Specifically, you should run the program with each set of parameters using the `time` command to determine both the real time and CPU time. Record these times in a comment at the top of your source file.

Marking Scheme

Activity #1:

• Creation of `tickets_sold` global variable and mutex	2 marks
• Creation of threads	1 marks
• Thread join and summary data	2 marks
• `ticket_agent` routine	3 marks

Activity #2:

• Timing information	2 marks
TOTAL	10 MARKS

Submission

You should submit your commented *source file* through the lab drop box in Canvas.