DEU CAFETERIA

MAKE SURE YOU READ THE DOCUMENT COMPLETELY PRIOR TO SUBMITTING YOUR ASSIGNMET!!!

Due:
29th October, 11:59 pm

Goal

In this assignment you are expected to simulate and solve DEU Cafeteria synchronization problem, which its details are given. You know there are long waiting lines in front of the cafeteria during lunch times. With your simulation and solution with mutex and semaphores students hopefully will no longer wait so long for their lunch.

Scenario

In DEU Cafeteria there is one cook during lunch times that put the food in the trays and puts the trays on the conveyor (a moving band that carries the filled trays out of the kitchen so students can fetch them and eat). Conveyor can take at most 8 trays. If the conveyor is full cook sleeps until a tray is fetched. Students come in random times and fetch their trays. If the there is no tray available they wait until cook fills a tray and puts it on the conveyor. For the sake of simplicity you don't have to worry about the travel time of the tray on the conveyor.

Requirements

- Simulation has to start with a full conveyor. (With 8 trays ready to be fetched)
- There is only ONE (1) cook. (This means you will need a thread for the cook)
- If the conveyor is full cook sleeps, if not fills trays.
- Filling a tray takes random amount of time between 2-5 seconds.
- Each student has to be a different thread and has to be created at random times. (Creating constant amount of student threads prior to simulation is not ACCEPTED!!!)
- A student comes to cafeteria in random intervals that changes between 1-4 seconds.
- A student to fetch a ready tray takes exactly 1 second.
- One student can fetch a tray at a time (as it is in the real cafeteria).
- Order of the students while fetching a tray is NOT important!!!
- Your simulation is to be continuous (should not end unless it is stopped or killed)
- Your simulation needs to keep track of the
    - The time cook started to sleep
    - The time cook awake
    - The time he started and finished to fill the n^{th} tray
    - The time m^{th} student arrived
    - The time m^{th} student fetched his tray
- Your simulation must keep the record of n (total number of trays filled), m (total students that came to cafeteria) that are given above and total number of students ate already (fetched tray).
- Each tread itself must increment or decrement those global variables. (Not the thread creating code)
- Your simulation must give correct and detailed output of the times listed above, m and n at each state change as well as an illustration like the one given below. (30 points)
- Your simulation must have another "monitor" thread for printing out the illustration below at each state change. For printing the times, each thread (student or cook) must print out their own times.

Sample Output **(Note that times are not given, but your implementation should output times)**

```
                                CONVEYOR
                          ____________________
                          TRAYS READY      :5
                          AVAILABLE PLACE  :3                WATING LINE
          COOK                                            ________________
  ________________                                               3
                                                         Student(s) is/are
  Working                 [ ] [ ] [ ] [ ] [ ]                 waiting

  Filling 12th tray


                          CAFETERIA STATISTICS
                       ____________________________
                       TOTAL TRAYS FILLED       :11
                       TOTAL STUDENTS CAME      :9
                       TOTAL STUDENTS FETCHED   :6
```

Submission

Make sure your submission fully covers the following requirements; otherwise it will be graded as 0 (ZERO).

Submission will be via Google Classroom.

- Put only your (**make sure no other files exist in the folder**):
  - make file (you must have a make file) (Name it as: "Makefile")
  - your source file (Only one c file named as "cafeteria.c")

  in a folder named exactly as your ***StudentID_Name_Surname*** (Ex:2008510900_Mete_Akdogan)
- Tar your folder and name it exactly as your ***StudentID_Name_Surname.tar*** (Ex:2008510900_Mete_Akdogan.tar)
- Submit only once your zip folder. You will lose 10 points for each re-submission. So make sure every checkpoint is accomplished before you submit.

  -10 POINT PENALTY APPLY TO EACH DAY OF LATE SUBMISSIONS.

Honesty

Your submissions will be scanned among each other as well as the Internet repository. Any assignments that are over the similarity threshold of MOSS will get 0. We strongly encourage you not to submit your assignment rather than a dishonest submission.

For Questions

For any questions about the assignment please comment under the assignment in Classroom. I will try to answer any of your questions as soon as possible, except the ones "Hocam my code does not work, can you fix it" or "I have implemented it but it does not work, can you look at it". Search engines are far more suitable options.