Group Project 3: All about deadlock

CECS 326 – Operating Systems

You should submit the required deliverable materials on BeachBoard by **11:59pm, April 4th (Tuesday), 2023.**

1. Problem Description

Consider that has this kind of scenario: two villages (Eastvillage and Westvillage) only have a single-lane road for the connection. People from these two villages only can use this road for exchange or share their produce. The road can be deadlocked if a people from either East or West on the road simultaneously. To solve this problem to avoid deadlock, please design an algorithm that uses semaphores and/or mutex locks. There have no concerns for the starvation cases.

Implement your solution using synchronization tools. In particular, represent the people at Eastvillage and Westvillage as separate threads (*east_village.java* and *west_village.java*). Once a people is on the road, the associated thread will sleep for a random period of time, representing traveling across the road. You should design a new action for each people when they get into the road, such as eat a donut, to wait for some time. Design your program so that you can create several threads representing the two villages' people without deadlock in the road.

You can flexibly design the algorithm, the test should have no deadlock in the multiple execution cases.

2. The Required Deliverable Materials

(1) A README file, which describes how we can compile and run your code.
(2) Your source code, should submit in the required format.
(3) Your short report, which discusses the design of your program.
(4) A recorded video shows the output and runtime

3. Submission Requirements

You need to strictly follow the instructions listed below:

1) This is a **group project**, please submit a .zip/.rar file that contains all files, only one submission from one group.

2) Make a **video** to record your code execution and outputs. The video should present your name or time as identification (You are suggested to upload the video to YouTube and put the link into your report).

3) The submission should include your **source code** and **project report**. Do not submit your binary code. Project report should contain your groupmates name and ID.

4) Your code must **be able to compile**; otherwise, you will receive a grade of zero.

5) Your code should not produce anything else other than the required information in the output file.

7) If you code is **partially completed**, please explain the details in the report what has been completed and the status of the missing parts, we will grade it based on the entire performance.

8) Provide **sufficient comments** in your code to help the TA understand your code. This is important for you to get at least partial credit in case your submitted code does not work properly.

Grading criteria:

Details	Points
Have a README file shows how to compile and test your submission	5 pts
Submitted code has proper comments to show the design	15 pts
Screen a ***video*** to record code execution and outputs	10 pts
Have a **report** (pdf or word) file explains the details of your entire design	20 pts
Report contains clearly individual contributions of your group mates	5 pts
Code can be compiled and shows correct outputs	45 pts

4. Policies

1) Late submissions will be graded based on our policy discussed in the course syllabus.
2) Code-level discussion is **prohibited**. We will use anti-plagiarism tools to detect violations of this policy.