# Homework 2 for CS153 (Winter 2021)

## *Due: Monday 2/8/2021*

**Instructions:**

I. Write synchronization code to simulate each of the following scenarios:

a. (3 points) A bakery where threads of two types representing two ingredients cake mix and icing arrive. Whenever we have one of each, we make a cake. You may assume that there is a baker thread or have just threads for the ingredients such that either the cake mix or the icing makes the cake when both ingredients are there.

b. (3 points) The recipe in part a has changed – now we need two portions of cake mix to arrive (in addition to icing) before we can make cake. Update your implementation.

II. You are writing code for the voting machines for an upcoming election. You use shared counters, one for each candidate to keep track of the votes as they come from the different voting machines. You can think of each vote machine as a thread: every time it receives a vote, it increments a centralized counter for that candidate.

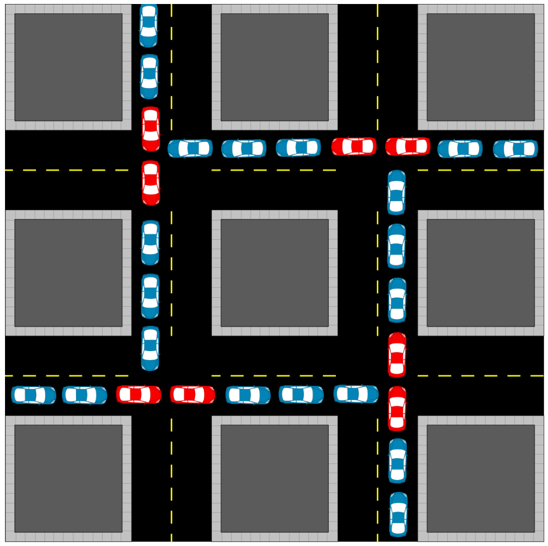
(a) (2 point) Explain what could go wrong with this implementation if we do not use synchronization

(b) (2 points) Use Sempahores to solve this problem without changing the logic of the code other than adding the lock operations. You can show pseudo code or describe you changes clearly.

(c) (2 point) Consider the following improvement to the implementation suggested by a cs153 veteran: for each thread, maintain a local count of the votes, and then update the global count periodically. Do we still need synchronization?

(d) (2 point) Compare the implementation to your implementation in b. Which performs better? Are there any drawbacks to the faster implementation?

III. (5 points) Traffic in Manhattan goes around a block as shown in the figure below.



Picture from: Wikipedia (https://upload.wikimedia.org/wikipedia/commons/thumb/d/d9/Gridlock.svg/440px-Gridlock.svg.png)

Having studied concurrency, you recognize that even though we call this gridlock, this situation may be a case of deadlock. Use our criteria for deadlock to show whether this is indeed deadlock or not.

If this is deadlock, discuss and compare two solutions to prevent it from happening.

IV. An OS uses a multiple level feedback scheduler with 3 round-robin levels. The quantum for the two levels are 2, 4 and 6 respectively. Assume that we have 5 jobs that arrive at intervals of 5 msec starting at time 0. Assume that once a quantum starts, it runs until it ends (or the process finishes); it is not interrupted by a newly arriving process. The processes have a burst lengths 20, 5, 2, 4, and 12.

(a) (4 points) Show the scheduling timeline (which process runs at what times) for the processes until the end.

(b) (2 points) Compute the normalized turnaround time for each process ((finish time- arrival time)/running time)