CSC2044 Concurrent Programming

Lab 7: Synchronizers

In this lab, we will try to simulate or solve different issues by using the 4 different synchronizers that we have covered in the lecture class, (i) blocking queue, (ii) countdown latch, (iii) semaphore, and (iv) barrier. In fact, synchronizer is just an object that can be used to coordinate the threads based on certain conditions. This also means that we can try to use them to control the executions of multiple threads. To start with, you may refer to (or try to run) the sample code provided in the lecture slides.

A. Blocking Queue

Exercise 1

- Create a blocking queue with 10 slots.
- Create 5 staffs that each will put a door gift into the queue each round for 10 rounds. A staff might take in between 1 – 2 seconds (randomly) to prepare a door gift, before putting it into the queue.
- Create 5 customers that each will take a door gift from the queue each round for 10 rounds. If there is no door gift available in the queue, the customer will need to wait.

B. Countdown Latch

Exercise 2

- Modify your code from exercise 1 to include a countdown latch. The purpose of the countdown latch is to ensure that all the customers (threads) can only start to get a door gift from the queue after we have invoked the start() method for all of them. Otherwise, certain customers (threads) would be able to get a head start over the others.

C. Semaphore

Exercise 3

- When the local convenient store is open, 10 customers are there waiting to enter the store. But due to the pandemic, the store must limit the number of customers that can enter the store to 4.
- Simulate the scenario given above using semaphore. A customer (or thread) is only required to print a message to indicate that he or she has entering or exiting the store. You should be using the semaphore to control if a customer (or thread) is allowed to enter.

D. Barrier

Exercise 4

- An online multi-player game can only start after all the 4 players have pressed the start button.
- Create 4 players (threads) that each can delay in between 1 – 2 seconds (randomly) before pressing the start button (arrived at the barrier).
- Print appropriate message(s) to show the progress and when the game starts.