Assignment 4, MTL458

Junction Management System

In this assignment, you are required to implement a Junction management system. There are 4 lanes which intersect with each other. Trains can enter a lane from the directions as shown in the figure. North is top of the figure. Each lane can allow only one train to pass through it at a time. Each train is represented by an independent thread and it performs the following 3 functions, in order:

- Arrive Lane - `arriveLane()`: The front of the train arrives at the first intersection point in the lane.
- Cross Lane - `crossLane()`: The train crosses the lane.
- Exit Lane - `exitLane()`: The complete train exits the lane.

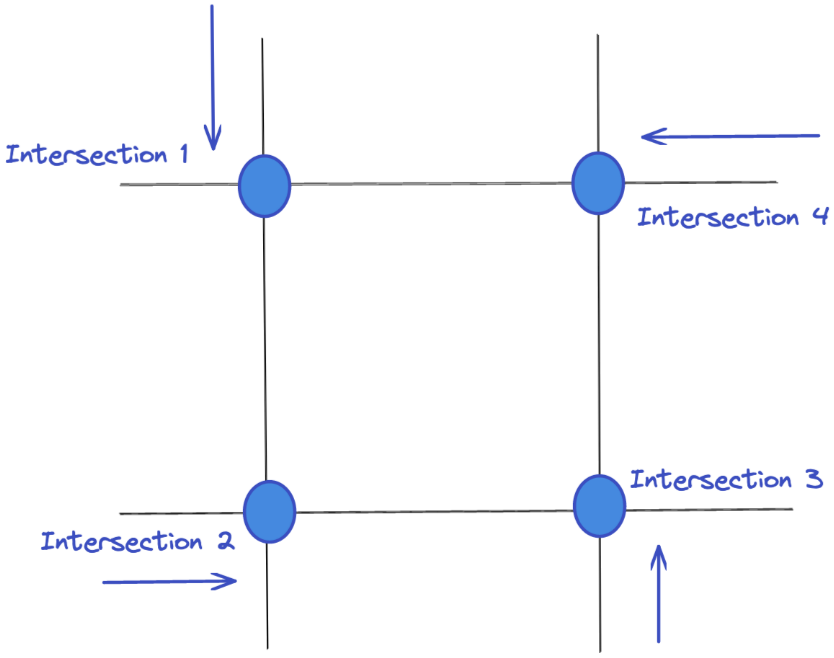

Figure 1: Junctions at the Subway

To prevent crashes and multiple trains entering a lane at the same time, the trains do the following:

- To arrive at a lane, a train blocks the first intersection on the lane in advance.
- To cross a lane it must block the next intersection on the lane.
- Since the size of train is large, it is safe to unblock the intersection points it has blocked earlier only after the train has exited the lane.

Note that following the above conditions can cause a deadlock. You are required to detect and resolve any deadlock. For this, a separate thread will be running, which periodically checks if there is a deadlock and signals one of the trains to go. You are free to use any synchronization primitives for implementing controllers, other critical sections/signalling in your code.

You submission will be judged on 3 properties:

- **Safety:** No crash can occur, and only one train can use a lane a time.
- **Liveness:** The system is not stuck forever.
- **Starvation:** No train should be starved from entering the lane.

Submission Details

You are given a single file, main.c which takes a single command line argument like: ”NSEWEEWSNSENNWSEW”, which represents the directions from which trains are entering the lane. The main function should launch a background deadlock detector thread. It should then launch a thread for each train. You are required to complete all the functions(including main) in the given file. In addition you can use any global variables/functions, or split your code into multiple files.

Submit a zip file named <Entry_No>_A4.zip (eg: 2019MT60648_A4.zip). Upon unzipping your submission, we should get a folder named <Entry_No>_A4 containing main.c and other code files.