COMP3520 Operating Systems Internals Assignment 1 – Synchronization

General Instructions

This assignment is about synchronization. It consists of two **compulsory** tasks:

1. write a multithreaded program to solve the synchronization problem as described in the section "The Problem: Traffic Light Synchronization"; and
2. answer the assigned discussion document questions (which are provided in a separate document).

This assignment is an individual assignment. Whilst you are permitted to discuss this assignment with other students, the work that you submit must be your own work. You should not incorporate the work of other students (past or present) into your source code or discussion document.

You will be required to submit your discussion document to *Turnitin* for similarity checking as part of assignment submission. The examiner may use other similarity checking tools in addition to *Turnitin*. Your source code may also be checked.

Submit your source code and report to the appropriate submission inboxes in the COMP3520 Canvas website.

The Problem: Traffic Light Synchronization

Consider a traffic light-controlled four-way intersection where a minor road perpendicularly intersects with a trunk road. The trunk road is in a north-south orientation. To control vehicle movements, separate sets of traffic lights including right turn arrows (but not left turn arrows) are provided on the trunk road for forward-moving vehicles and right-turning vehicles (left-turning vehicles obey the traffic lights for forward-moving vehicles). Also, a single set of traffic lights without turn arrows is provided on the minor road to allow vehicles to turn left onto the trunk road or to proceed straight ahead. Vehicles are not permitted to turn right from the minor road.

The trunk road consists of one through lane for each direction. In addition to through traffic lanes, there are dedicated lanes for vehicles turning right from the trunk road.

The minor road consists of two lanes, one for each direction.

For this problem, assume that there are no pedestrian or cyclist crossings. Also, assume that each traffic light is either red or green (i.e. there are no yellow lights), and that vehicles instantly accelerate to normal driving speed or instantly stop whenever required.

The operation of the traffic lights is described below.

By default, the traffic lights on the minor road display red. The traffic lights on the trunk road display red right-turn arrows for both directions but green for other traffic.

X seconds after the traffic light system has entered the default state, the traffic lights on the trunk road immediately change to red; the right-turn arrows stay red. **Two** seconds

after the trunk road traffic lights change to red, the traffic lights for the minor road change to green.

Y seconds after the traffic lights on the minor road have changed to green, these traffic lights immediately change to red. **Two** seconds after these lights have changed to red, the right-turn arrows for controlling right-turn movements from the trunk road to the minor road change to green; the traffic lights for other traffic stay red.

Z seconds after the right-turn arrows have changed to green, these arrows immediately change to red. **Two** seconds after these arrows have changed to red, the traffic lights on the trunk road for forward-moving (and left-turning) vehicles change to green; the traffic lights on the minor road and right-turn arrows stay red. **This is the default state of the traffic light system.**

The traffic light cycle continues until there are no more vehicles to serve.

All vehicles that wish to turn right from the trunk road must comply with the right-turn arrows. All other vehicles must comply with the regular traffic lights. Green means "go" while red means "stop" (left-turning vehicles must not turn on red).

In this problem there are **six** traffic lights to control vehicle travelling in the following directions:

a. n2s (from north to south on the trunk road);
b. s2n (from south to north on the trunk road);
c. e2w (from east to west on the minor road);
d. w2e (from west to east on the minor road);
e. n2w (from north to west right turn on the trunk road); and
f. s2e (from south to ease right turn on the trunk road).

Both traffic lights in the (n2s, s2n) pair will change to green or red at the same time; the same behaviour applies for the other pairs (e2w, w2e), and (n2w, s2e). To make the problem more challenging and more interesting, in this assignment you are asked to use **THREE** mini-controllers. Each mini-controller controls a pair of traffic lights and they coordinate to make the traffic light system function as described above.

NOTE: You **MUST NOT** only use a single controller to control these six traffic lights (**If you do this, you will be awarded a failing mark for your source code!**).

To implement synchronization between the various threads, you may use mutexes, condition variables, semaphores or a combination thereof. **However, you must not use busy waiting or any other type of synchronization mechanism.**

When the main thread starts, it asks the user to enter the following parameters:

- Total number of vehicles to dispatch;
- Vehicle arrival rate to the intersection (in seconds);
- The minimum time interval *t* (in seconds) between any two consecutive vehicles to pass through the intersection;
- Number of seconds *X* the traffic lights on the trunk road for forward-moving vehicles stay green;
- Number of seconds *Y* the traffic lights on the minor road stay green;
- Number of seconds *Z* the right-turn traffic light arrows on the trunk road stay green.

The time interval between consecutive vehicle thread creations is randomly determined (with a uniform distribution); however, there must be a gap of at least one second

between consecutive vehicle thread creations if and only if the corresponding directions of travel are the same (if the directions of travel are different, then two consecutive vehicle thread creations may occur simultaneously).

Also, the directions of travel of the vehicles are randomly generated.

Each vehicle thread is given

1) an *id*, which is a natural number starting from 0 according to its arriving order in one direction and
2) a *direction*, which indicates its travel direction (n2s, s2n, e2w, w2e, n2w, s2e).

For example, a vehicle thread with *id* = 3 and *direction* = n2s is the 4th vehicle to pass through the intersection from north to south.

To check whether your program functions properly, **each vehicle thread** must print the following status messages wherever appropriate:

- "Vehicle *id direction* has arrived at the intersection."
- "Vehicle *id direction* is proceeding through the intersection."

The traffic light mini-controller threads must print the following status messages wherever appropriate:

- "Traffic light mini-controller *directions*: Initialization complete. I am ready."
- "The traffic lights *directions* will change to red now."
- "The traffic lights *directions* have changed to green."

where *directions* are one of traffic light pairs (n2s, s2n), (e2w, w2e) and (n2w, s2e).

When there are no more vehicles to serve, the main thread must print the following status message and then terminate:

- "Main thread: There are no more vehicles to serve. The simulation will end now."

Additional Requirements

Source Code

The source code template for this assignment is available on the COMP3520 Canvas website. **You must use this template to implement your solutions.**

Your solution must be implemented in the C language. C++ features are not permitted except those that are part of an official C standard.

Your source code needs to be properly commented and appropriately structured to allow another programmer who has a working knowledge of C to understand and easily maintain your code.

You need to include a well-structured and properly commented *makefile* that allows for the compilation of your source code using the *make* command on the School of Computer Science servers.

Testing and Debugging

You are responsible for testing and debugging your source code.

It is crucial that you ensure that the source code you submit compiles correctly on the School of Computer Science servers and that the resulting binary functions as intended. If you submit source code that cannot be compiled on the School servers or if you fail to make a serious attempt, you may be **DISQUALIFIED** from this assignment.

Discussion Document

You are required to answer all assigned questions in a separate written document. The questions and requirements specific to the discussion document will be provided in a separate document.

Other Matters

Do not engage in academic dishonesty. Malpractice will not be tolerated and may result in **DISQUALIFICATION** from this assignment.

Marking criteria for the source code and discussion document will be provided separately.

To maximize your chances of realizing your full potential in COMP3520, **please start work on this assignment promptly.**