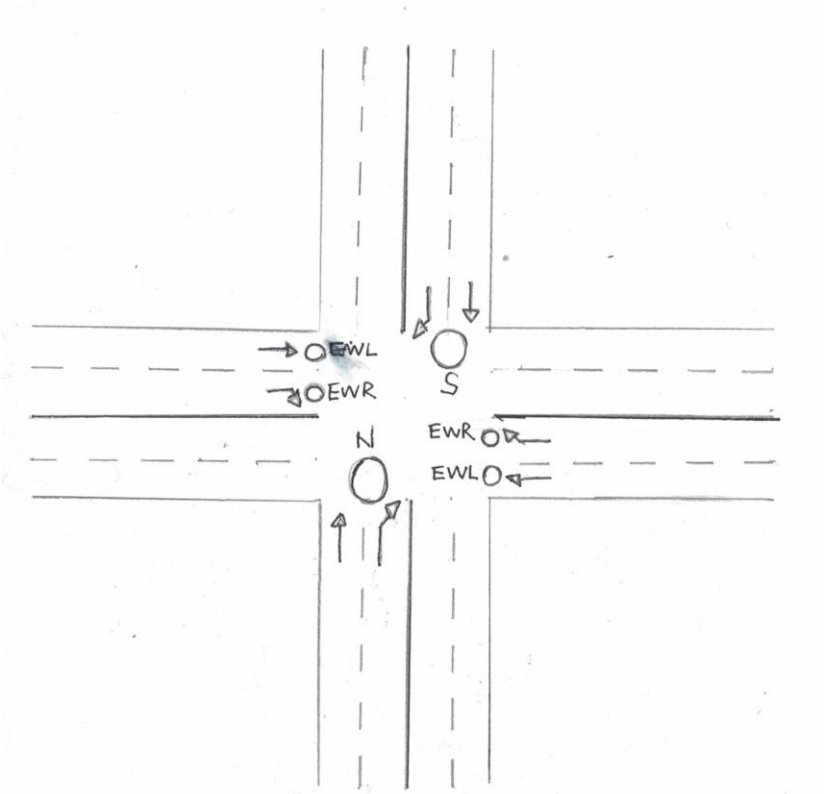
Title: Traffic Signal Controller (35%)



The above diagram is an illustration of a street intersection. Each circle stands for a set of traffic signals that all behave identically at all times. For example, both northbound traffic lanes get the same signal, but the left and right lanes of eastbound traffic get different signals. Northbound and southbound traffic get different signals. The signals are named: S, controlling southbound traffic (which may also make a right turn); N, controlling northbound traffic (which may also make a right turn); EWL controlling east- and west-bound through traffic in the east-west right lanes; and EWR controlling right turns from the east-west right lanes.

Compatibility of movement: no two signals may be green or yellow at the same time. If they are, crashes will result.

For each of the signals there is a sensor with the same name that indicates the presence or passing of a vehicle. The sensor sends a pulse when a vehicle arrives at it.

The behavior of the signals is dictated by what is detected by the sensors. If there is no traffic, or only EWL traffic, the EWL signal stays green. When traffic arrives for another direction the sensor indicates this to the controller. The controller stops any incompatible movements by first signalling yellow for 6 seconds and then changing to red. When all conflicting movements are

stopped, the controller changes the signal to green. Once green, a signal changes to yellow (and then red):

- no sooner than 6 seconds after turning green
- no later than 12 seconds after turning green if there is conflicting traffic
- in 2 seconds after the last signal from its sensor provided this is within the 6..12 second period
- In addition to these rules there is a nearby train track that has to be accommodated (see below).

If sensors indicate that there are conflicting requests they are to be served in round robin order, beginning with EWL, N, S, and finally EWR. Except for the EWL signals, lights should only turn green when sensors indicate traffic is present except as these rules are modified to handle the train.

The Train

Not shown on the diagram but nevertheless present, there are train tracks to the south of the intersection, parallelling the east-west roadway. A train-arriving sensor, named TA, and a train-departing sensor, named TD, are on the tracks. When an arriving train is detected the signals must allow northbound traffic to clear the tracks for a 10 second green period, regardless of the information from the N sensor and regardless of the current state. Of course, any current incompatible signals must first be changed immediately to yellow for 6 seconds and then to red. (Note that for train arrivals the current green phase may last less than 6 seconds). After that, only EL and WL traffic can proceed until the train departs. You may assume that each train departs before the next train arrives. After the train leaves, the signal should pick up where it left off, either restarting the interrupted phase if it has traffic, or continuing with the next phase in round robin order that does have traffic.

Designing a solution

You should first think about your design independently of the details of how we represent the signal lights and sensors in the program. What states have to be made mutually exclusive? How might you ensure mutual exclusion using the synchronization mechanisms?

Start with a simpler intersection that has only a single signal controlling north-south traffic and a single signal controlling east-west traffic. If you come up with good design principles you should be able to extend your solution to the more complex intersection in a straightforward way.

Decide what threads you are going to have and what conditions you need to signal between threads. Decide whether you are going to use monitors or semaphores. How will your program know when to take action based on the time specifications in the problem description?

Implementing signals and sensors

Your program will interact with the signal lights and sensors through its standard input and output.

Sensor input

Sensor input comes in the form of lines read from standard input. A line consists of the identity of the sensor (N, S, EWR, EWR, TA, TD) followed by a newline character (i.e. you can run your program in a terminal session and type sensor input to it). Example input:

N
S
EWR
EWL
TA
TD

Timing of the input clearly matters to the behavior of the signal. For the convenience in this project, you may add a timestamp before each sensor identity read from the standard input to represent the time at which the signal arrives.

Signal and sensor output

Output from your program will be a series of lines, each line containing a timestamp and either a light command to a single signal or an echo of a sensor input. Timestamps are integers representing the number of milliseconds since the start of the run. A light command line consists of 4 tokens: a timestamp, the letter L, the signal name (EWL, EWR, N, S) and a colour indicator: G, Y or R. The command indicates that the named signal is to show the indicated colour.

Sensor inputs must be echoed to the output as soon as they arrive. These lines consist of three tokens separated by whitespace: a timestamp, as above, the letter S, and the sensor name.

Example output

0 L N R
0 L S R
0 L EWL R
0 L EWR G
2250 S N
5500 S EWR
7500 L EWR Y
13500 L EWR R

13500 L N G
19500 L N Y
25500 L N R
25500 L EWR G
...

Bonus marks

Use the output to drive a graphical display of the state of the intersection.

Assignment details

You need to complete the project in a group of 4-5 members and evaluation will be done in two phases.

Phase 1: 15% (Due Friday 6 April at 11.59 p.m.)

Deliverables (By group):

- Preliminary design document which lays out your general approach to the problem, including what you've determined to be the basic synchronization requirements, areas that you see as problematic, what threads you are going to use, what different iterations of your design are, your data structures, etc. (10%)

- Preliminary code which illustrates that you have mastered the input and output requirements, the synchronization primitives. Working code for a 2-way intersection with sensors would represent good progress. Show sample outputs of running the program with different input files – attach the input files too. (5%)

Phase 2: 15% (Due Friday 11 May at 11.59 p.m.)

Deliverables (by group):

- A report containing code and a revised design document describing in detail what you built. Explain what the most important things to look at in your design and code are, what changes between the preliminary and final designs, how well it works, does it meet the specification, etc.? Softcopy of the report source code, input files and executable must be submitted through Spectrum and hardcopy of the report must be placed into my letter box located at 3rd Floor of Block A by the due date.

In class presentation: 5% (Monday 14 May)

Explain your design approach to cope with the synchronization requirements, important parts of your code like data structures, and demonstrate your program in not more than 15 minutes per group.

Peer evaluation (Individual)

A peer evaluation mechanism will be used to determine each group member's contribution factor. The contribution factor will be multiplied with the marks obtained for both phases to produce your final marks. Every group member must submit his/her peer evaluation and evaluate every member in the group, including himself/herself. Name the peer evaluation form "WIF3003 Peer Evaluation Form (***)-Phase #", where *** is your student ID and # is either 1 (for phase 1) or 2 (for phase 2). Submit the form through Spectrum by the due date for each phase. Two submission links will be created for each phase – one for group submission (see above), another for individual peer evaluation form.

The contribution factor will be computed for each group member based on the formula stipulated below:

Peer Evaluation Score	Contribution Factor
0-4	0
5-8	0.2
9-12	0.4
13-16	0.6
17-20	0.8
21-24	1.0

Final contribution factor for each student will be the mode of all contribution factors given by his/her peers. Critical disputes in assigning peer evaluation scores will be investigated by the course instructor.

The marks for each student in each phase will be computed using the following formula:

*Group's marks in that phase * contribution factor*

For example, if a group obtains 13 marks for phase 1 evaluation, and the final contribution factor for a student is 0.8, the student's marks for phase 1 evaluation will be:

*13 * 0.8 = 10.4*

Late submission

If for some good reason it is impossible to get the assignment in by the deadline, email the lecturer to negotiate a late submission. This should be done prior to the deadline. Late assignments without negotiation will have marks deducted (2% per day overdue).

Penalties

If it is determined that plagiarism has occurred, one or more of the following penalties may be imposed:

- loss of all or part marks for the assessment item;
- downgrading the final grade in the course;
- imposing a grade of fail in the course.