DUE 12/1/2019 at 11:59PM

Late Penalty:

- 5% within 2 days
- 10% for each week till 12/15/2019

Description: in this project, you will implement an intelligent traffic light system where cars arrive at an intersection towards West, East, South, North, West-North (towards West but makes a turn to North), West-South, East-North, East-South, South-West, South-East, North-West, North-East directions. Your program should enforce the requirement that no two or more cars enter the intersection with "conflicting" directions. directions that conflict with each other include, for example, North and East, South and West, North-West and South. However, notice that North and South does not conflict. Similarly, North-West and South-East does not conflict. The goal of the solution, therefore, is to try to minimum the total time passing the intersection given any input of arriving cars.

Understanding that multi-threading concurrency has the potential of providing maximum potential simultaneous operation while ensuring mutual exclusion, we can model each car arriving as a thread and the intersection as the critical section. Upon the arrival of a car, we create a new thread and pass the direction as the argument. There should be four queues that manage all the four directions as requests to pass the critical section. All threads should be blocked while waiting for conditions to pass the critical section. An additional thread (could be just main thread) should check all Head-of-the-Line queued threads understand if it should signal threads in certain directions. There are two conditions to wait for a new car thread: (1) if it's not the head-of-line thread for its direction queue; (2) if it's not allowed to pass the intersection. A sleep time of 5 seconds should be used to simulate that a car takes time to pass the intersection.

Your program should display the following events on screen:

- Thread creation
- Direction ID # car arrived intersection (e.g., West #3 car arrived)
- Direction ID # car left intersection (e.g., West #3 car left intersection)

At the end, your program should print out the total time taken for all cars to pass the intersection.

Input File: three sample inputs (simple.txt  , medium.txt  , difficult.txt ) in the following format will be provided:

time, direction

0, N

2, W

2, E

...

Bonus (15%): your program allows N (configurable, 10 as default) cars in one direction to go. In this case, cars do not wait for the previous one to leave the intersection (5 seconds). Instead, the waiting time will be 1 second.

Testing: Your program should run forever until all cars pass the intersection.

Cygwin compilation: "g++ -std=c++11 sample.cpp -lpthread"

Project Report: a max of 2-page brief description of the results (total time) on three sample inputs. You should briefly justify the result based on your observation and analysis. If you implement the bonus part, you should present and justify your result in the report too.

Submission: One zip file containing the following should be submitted:

- All source code
- Project report (2-page max)
- README that "clearly" specify
    - Running environment: Cygwin, Linux, Mac
    - Compiler: C++11 (must be compatible)
    - commands that compile and run your program
    - Bonus part (if any) you have implemented