CSE 110 - Honors Contract

Instructor: Tyler Baron

Overview

The goal of an honors contract is to provide students credit for completing a small project that stretches the bounds of what is taught in the class. This means that those who undertake this contract will need to demonstrate mastery of some of the class's more advanced topics, while extending that domain a little beyond what is covered in class.

This honors contract focuses on the topic of classes and objects, as covered in class and extends this into the domain of basic class inheritance and polymorphism, which are not covered in this course.

Project Description

Design and build a Java program that simulates a four way traffic intersection. Each direction will maintain a list of vehicles waiting to cross. Each vehicle has a specific type, such as a car or truck. Only one pair of directions (N and S, or E and W) will be able to move at a time. Vehicles are removed from each list, as long as they have time to go. The light will change after a set amount of time, and then the other direction may go. If there are any vehicles remaining in the intersection after the light turns red, a new random vehicle is added to one of the directions. Emergency vehicles are given priority (moved to the front, and their direction is allowed to go). The simulation ends when there are no vehicles left.

You will print out the state of your intersection each time a light changes, or an emergency vehicle is given priority. It is up to you how it looks, but it should look something like the one below.

```
NS: Green                    |      |
EW: Red                      |      |
                             |  T   |
                             |  C   |
---------------------------------|  E   |---------------------------------

                C       T       T       C       C       C

---------------------------------|  S   |---------------------------------
                             |  C   |
                             |      |
```

Requirements

To get credit for this contract you must email me a zip folder of your Java project containing all the code needed to run your project no later than 11:59 PM 12/03/2017.

Your project must meet the following requirements:

Have an abstract class called Vehicle

- One int variable called time which represents the time it takes for this vehicle to get through the intersection.
- One boolean variable called isEmergency which represents whether or not this is an emergency vehicle.
- A default constructor which sets time to 0 and isEmergency to false..
- A regular constructor which sets the time and isEmergency to their parameters.
- Two abstract methods which are a get for time and isEmergency.

Have four classes called Car, Truck, Semi, and Emergency

- Each of these must extend Vehicle.
- Each only has a single default constructor.
    - This constructor must use super as appropriate.
- Only emergency sets isEmergency to true, all else set it to false.
- It is up to you to set time for each how you want, but they should follow the pattern that car.time < truck.time < semi.time and emergency can fit however you like.
- Must implement the abstract methods from Vehicle.

Have a class called TrafficLight

- Four arrays, one for each direction of traffic., which can hold up to five vehicles each.
- A boolean variable called running which indicates whether the simulation is running or not.
- Two boolean variables for whether the NS and EW directions have a green light or not.
- An constant int called time that indicates how long the light stays green for one direction or the other.
    - Will need to be greater than or equal to the longest time you gave to any vehicle type.
- Only one, default, constructor which places 1 to 5 vehicles of a random type in each direction and sets running to true.
    - Do not add any emergency vehicles yet.
- A method to print the state of the intersection.
- A method to add a Vehicle to a direction indicated by a parameter..
    - Should take another boolean parameter which indicates whether or not emergency is a type of vehicle that can be added.
- A boolean method that returns true if there are no vehicles left in any direction.

- A boolean method that returns true if there are any emergency vehicles in any direction.
- A boolean method that returns true if there is an emergency vehicle in a direction specified in the parameter.
- A method that moves finds which direction has an emergency vehicle in it, moves that vehicle to the front of that direction, and turns the light green for that direction, even if it was green last time.
- A method which "runs" movement during a green light.
    - For the two directions (NS or EW) that have a green light, remove the front vehicle from the list and move the other cars forward as needed.
        - Think about how you might do this where moving all the subsequent cars is not needed.
        - Obviously, do not remove a car if there are no cars to remove.
    - Keep track of how much time is left.
        - Start with the time constant.
        - When you remove a vehicle subtract its time from your running total.
            - If you remove 2 vehicles at once (because there's one in both active directions), only subtract the higher value one.
        - If a vehicle's time is higher than the remaining time, do not remove it.
            - If both directions have a vehicle and only one has enough time, remove only that one.
    - When time is 0, or there are no vehicles in the active directions, or there is not enough time for the remaining vehicles, stop and switch the active directions.
        - If there are any vehicles left, add a new random one of any type.
        - If there are not, set running to false.

Have a class called TrafficSimulation which serves as your main class

- Put your main method here.
- Have a method which runs the simulation.
    - Create a new TrafficLight object.
    - Print the starting state of the intersection.
    - Enter the main program loop that lasts as long as the simulation is running.
        - Make one "run" of the intersection and then print it out
        - Wait until the user hits enter and then repeat

Consequences for Failing to Fulfill this Contract

Honors contracts are intended to be a binding agreement between you the student and me the instructor. They are not something you start and then give up on if you decide you don't like it anymore. You are supposed to only pursue contracts you feel you can fulfill and once taken you are to see them to the end. Contracts should be fun, but you must take them seriously.

Barrett Honors College gives instructors the right to define in-class penalties for failing to fulfill an accepted contract. I define the penalty for failure to complete an accepted contract as:

Failure of this contract means the student receives an automatic score of 0 on the Final Exam.

This automatic 0 is still eligible for being dropped as your lowest exam score as normal. So, failure of this contract does not automatically constitute failure of CSE 110. However, it does rob you of a potential chance to increase your grade and means you are stuck with the other 3 exam scores, even if one of them is also a 0.

You will fail this contract if you fail to submit the required deliverables by the provided date, if you frequently miss our regular meetings, if the program you submit was not your own, if the program falls way short of expectations laid out in the requirements section, or at my discretion should any notable circumstances come to light that I feel violates any ASU Academic Integrity or Ethical Policies.