cs201c: Programming Evaluation 1
Instructor: Apurva Mudgal
Due date: Thursday, September 12, 2019 by midnight

Instructions. Note the following points carefully:

1. For your data structure to be correct, *we only require that the return values of all of the following function calls are correct.*
2. Your data structure should support all of the operations below in $O(\log_2 n)$ worst-case time, where n is the total number of cars currently on the highway.
3. You cannot use any in-built libraries (including standard template library). The data structure should be implemented in C++ from scratch.
4. You should use the templates feature of C++ for implementation (see Practice Lab 2).
5. **Collaboration is not permitted on this assignment. Your submitted code should be completely your own.**

   See section titled "Honor Code" in course outline already shared with you.

Topic: Cars on a National Highway

Consider a national highway H, which goes in a straight line and has two-way traffic. Further, vehicles can enter or leave the highway through the various side roads joining it.

The location of a vehicle on the highway at any given instant of time is given by a unique real number in the range $(-\infty, \infty)$. Further, the current time *curr* takes increasing, positive real values starting from initial value of 0. All vehicles on the highway travel at the uniform speed of 1 unit distance per 1 unit time.

Each vehicle has a unique registration number, which we take to be a non-negative integer value for the purposes of this assignment.

You have to implement a data structure, which maintains the state of the highway in a suitable format, and answers queries about its traffic flow. Your data structure should support the following operations (see attached figures for example):

1. `int insert(int r, float x, float t, int d)`:

   Assume $t > curr$.

   At (future) time t ($t > 0$), a new vehicle with registration number r ($r > 0$) enters the highway from a side road at location x. Further, the vehicle is traveling in direction d, where d takes only values 0 and 1. If d is 1, the vehicle is traveling from left to right on the highway, and if d is 0, the vehicle is traveling from right to left on the highway.

After this operation, we set the current time, *curr*, to t and return 1.

If a car with registration number r was already on the highway, insert is unsuccessful and we return 0.

2. `int delete(int r, float t)`:

   Assume $t > curr$.

   The vehicle with registration number r, currently on the highway, leaves the highway at (future) time t through a side road.

   Set current time, *curr*, to t, after this operation and return 1.

   If there was no car with registration number r on the highway, delete is unsuccessful and we return 0.

3. `int find_immediate_left(int r, int t)`:

   Consider the state of the highway at future time t, assuming that no car enters or leaves the highway between current time *curr* and future time t.

   Return the registration number of the car to the immediate left of car with registration r at time t.

4. `int find_immediate_right(int r, int t)`:

   Symmetric to above, with left replaced by right.

5. `int count_left(int r, int t)`:

   Consider the state of the highway at future time t, assuming that no car enters or leaves the highway between current time and future time t.

   Return the number of cars with x-coordinate strictly less than the x-coordinate of car with registration number r in this state, counting cars traveling in both directions. In other words, we return the total number of cars to the left of car r at time t.

6. `int count_right(int r, int t)`:

   Same as above, except left is replaced by right.

7. `int number_of_crossings(int r, int t)`:

   Assume t is greater than current time, and no car enters or leaves the highway between current time and future time t.

   Return the total number of cars which (i) are traveling in the opposite direction to the car with registration number r, and (ii) cross the car with registration number r between current time and future time t.