cs201c: Programming Evaluation 2

Instructor: Apurva Mudgal

Due date: Sunday, September 22, 2019 by midnight

Instructions. Note the following points carefully:

1. For your data structure to be correct, *we only require that the return values and printed output of all of the following function calls are correct.*
2. You can reuse code submitted by you as part of Programming Evaluation 1.
3. Your data structure should support:
   (a) `insert` in worst-case $O(\log_2(n))$ time,
   (b) `remove` in worst-case $O(\log_2(n))$ time, and
   (c) `next_crossing` in worst-case $O(k \log_2(n))$ time, where k is the number of cars which appear in the *printed output* of this function call (see description below).

   In the above, n denotes the total number of cars currently on the highway.
4. You cannot use any in-built libraries (including standard template library). The data structure should be implemented in C++ from scratch.
5. You should use the templates feature of C++ for implementation (see Practice Lab 2).
6. **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 One-way National Highway with Variable Speeds

Consider a national highway H, which goes in a straight line and has *one-way traffic (all cars go from left to right)*. 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.

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

The cars on the highway now have variable speeds. The speed at which a particular car is traveling is specified when it enters the highway.

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 v)`:

   At *current* time *curr*, a new vehicle with registration number r ($r > 0$) enters the highway from a side road at location x. Further, the vehicle is traveling with uniform speed of v units of distance per unit time.

   We return 1, if the operation is successful. On the other hand, if a car with registration number r was already on the highway, insert is unsuccessful and we return 0.

2. `int remove(int r)`:

   The vehicle with registration number r, currently on the highway, leaves the highway at current time *curr* through a side road.

   If remove is successful, we return 1. On the other hand, if there was no car with registration number r on the highway, remove is unsuccessful and we return 0.

   Note that the current time curr does not change after insert or remove operations.

3. `int next_crossing()`:

   This function finds the *earliest* (future) time t ($t > curr$) such that there are at least two cars r_1 and r_2 currently on the highway which (i) have different x-coordinates at current time *curr*, but (ii) occupy the same x-coordinate at (future) time t.

   After this operation, current time is set to t.

   The **printed output** of this function call consists of a sequence of lines, where each line consists of an x-coordinate w followed by registration numbers (in increasing order) of cars which occupy (same) x-coordinate w at (future) time t. Further, the x-coordinates w in successive lines (if the printed output consists of more than one line) are strictly increasing.

   For this operation, we assume that no car enters or leaves the highway between current time and future time t.

   Finally, the function returns 1 if such a future time exists, otherwise it return 0.

Input and Output Format

Input format. The first line contains m, the number of function calls. This is followed by m lines, with each function call given on a line by itself. The first number is the function code (with code 1 for `insert`, code 2 for `remove`, and code 3 for `next_crossing`) followed by values of function parameters:

```
1 r x t (insert)
2 r (delete)
3 (next_crossing)
```

It is assumed that at the start of input, $curr = 0$ and the highway is empty.

Output format. This consists *only* of the printed output of successive calls to `next_crossing`. (No return values are printed.)

Submission instructions

Submit a single file "(YOUR ROLL NO)-prog-eval2.cpp" on submission link on Google Classroom.