York University EECS 4101/5101 November 11, 2022

Homework Assignment #7
Due: November 18, 2022 at 11:59 p.m.

1. We want to design a data structure that stores a value and allows some historical queries on the values that were stored in the past. Initially, at time 0, the value stored is 0. An UPDATE(t, v) operation changes the value to v at time $t > 0$. We say the value of the object at time t is the value of the UPDATE with the largest time parameter that is strictly less than t. A GREATEST(t_1, t_2) query returns the maximum value the object had at any time during the interval of time from t_1 to t_2 (inclusive), where $0 \leq t_1 \leq t_2$. (If $t_1 = t_2$, then the query should simply return the value that the object had at time t_1.) You may assume that the time parameter t of an UPDATE operation is strictly greater than the time parameters given to all previous operations.

An example of a sequence of operations would be:
UPDATE$(3, 12)$
UPDATE$(5, 7)$
GREATEST$(6, 6)$ returns 7
GREATEST$(5, 6)$ returns 12
UPDATE$(8, 9)$
GREATEST$(6, 11)$ returns 9

The object has value 0 in the time interval $[0, 3]$, the value 12 in the time interval $(3, 5]$, the value 7 in the time interval $(5, 8]$ and the value 9 after time 8.

[3] **(a)** We can use an internal red-black tree (RBT) to represent the updates. What fields would you store in each node of the RBT? Explain what each field represents. Which field is used as the key for searches of the RBT?

[2] **(b)** In order to preserve the running time of RBT algorithms, we need to be able to compute the values of any field of a node that might change during a rotation in constant time by looking at information stored that node and its children. Explain how to do this for the fields you have defined above.

[2] **(c)** Briefly describe how you can implement UPDATE(t, v).

[5] **(d)** Give pseudocode that implements GREATEST(t_1, t_2) efficiently. Comment your pseudocode to make it easy to read.

Hint: this problem is well-suited to a recursive solution and if you use this approach, it might simplify your code to generalize the problem a little so that t_1 is allowed to be $-\infty$ and t_2 is allowed to be ∞.

[2] **(e)** Give good upper bounds on the running time of UPDATE and GREATEST. Define the variable that you are using to state the running time. (That is, if you say the running time is $O(n^3)$, you should also say what n represents.)