CSC263H1: Problem Set 3

Due Tuesday October 8 before 10pm

General instructions

Please read the following instructions carefully before starting the problem set. They contain important information about general problem set expectations, problem set submission instructions, and reminders of course policies.

- Your problem sets are graded on both correctness and clarity of communication. Solutions that are technically correct but poorly written will not receive full marks. Please read over your solutions carefully before submitting them.
- Each problem set may be completed in groups of up to three. If you are working in a group for this problem set, please consult the articles on collaboration and plagiarism on posted on quercus.
- Solutions must be typeset electronically, and submitted as a PDF with the correct filename. **Hand-written submissions will receive a grade of ZERO.**

  The required filename for this problem set is **problem_set3.pdf**.
- Problem sets must be submitted online through CrowdMark. If you haven't used CrowdMark before, give yourself plenty of time to figure it out, and ask for help if you need it! If you are working with a partner, you must form a group on CrowdMark, and make one submission per group. "I didn't know how to use CrowdMark" is not a valid excuse for submitting late work.
- Your submitted file(s) should not be larger than 9MB. You might exceed this limit if you use a word processor like Microsoft Word to create a PDF; if it does, you should look into PDF compression tools to make your PDF smaller, although please make sure that your PDF is still legible before submitting!
- Submissions must be made *before* the due date on CrowdMark.
- The work you submit must be that of your group; you may not use or copy from the work of other groups, or external sources like websites or textbooks.

Additional instructions

- If you are working in a group, remember to form your group before you hand in the problem set.
- Ensure that your last submission before the deadline is the submission you want graded - CrowdMark does not allow for late resubmissions.
- Please limit the length of your pseudocode to 2 pages.

1. **[6 marks] Idealy-Balanced vs Height-Balanced Trees**

   (a) Draw a minimal (smallest possible) example of a *height-balanced tree* that is *not ideally-balanced.* To receive (any) credit, you *must* write the balance factor next to every node, and also clearly indicate why the tree is not ideally balanced. Full points given for a *minimal* example.

   Page allowance: Please neatly present your solution using a single page. Any additional pages will not be marked.

2. **[4 marks] AVL Operations**

   (a) Let T be an empty AVL tree. Consider inserting these keys using the AVL insertion algorithm, in this order: 3, 1, 4, 15, 9. *Draw the resulting tree.* Do not draw the intermediate trees.

   (b) Indicate whether any rotations were required, and, if so, *how many* of *each kind.*

   (c) Next, *delete* the key 1 using the AVL deletion algorithm. *Draw the resulting tree.* Do not draw the intermediate trees.

   (d) Indicate whether any rotations were required, and, if so, *how many* of *each kind.*

   Page allowance: Please neatly present your solution using a single page. Any additional pages will not be marked.

3. **[10 marks] Augmentation.** Assume that T is a balanced binary search tree data structure (AVL tree). Explain how to augment T to support the operation $FindAverage(T)$ which finds (in constant time) the average of the keys stored in the subtree rooted at T, while preserving the worst-case running time of the operations $Rotation$, $BSTInsert$ and $BSTDelete$.

   (a) Briefly explain what additional information needs to be stored, and where it will be stored.

   (b) Justify that the operations $BSTInsert$, and $BSTDelete$ can be modified to maintain the newly stored information from part (a) while preserving their worst-case running time. You may explain this with a brief logical argument including references to theorems in the textbook; alternatively, you may present pseudocode.

   (c) Write pseudocode for the operation $FindAverage(T)$.

   Page allowance: Please neatly present your solution using *one or two* pages. Any additional pages will not be marked. To save space, do not re-describe or copy/paste pseudocode for standard AVL tree operations, unless your pseudocode is explicitly modifying those operations.