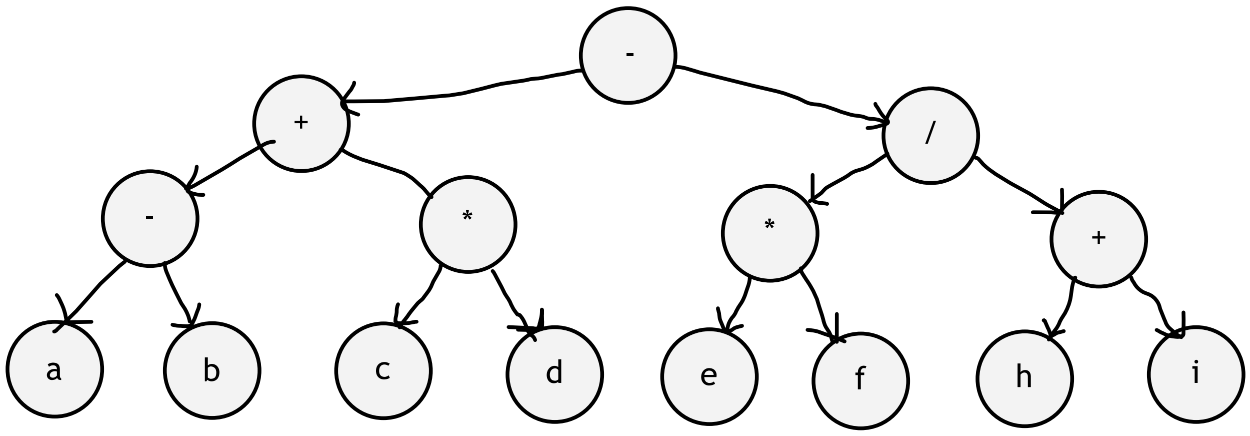
CptS 223 – Advanced Data Structures in C++

Individual Written Homework Assignment 3: Binary Trees, BSTs, and AVL Trees

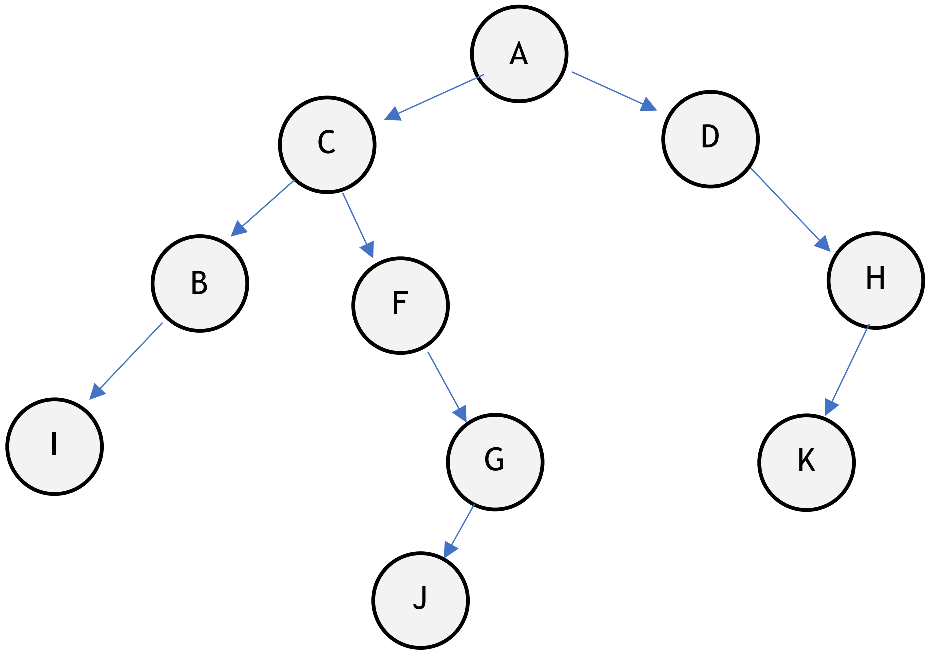
I. Problem Set:

1. **(15 pts)** Given the following infix expression: (a – b + c * d) – e * f / (h + i). Produce a binary expression tree. Recall, leaves of the tree are *operands*, and other internal nodes are the *operators*.

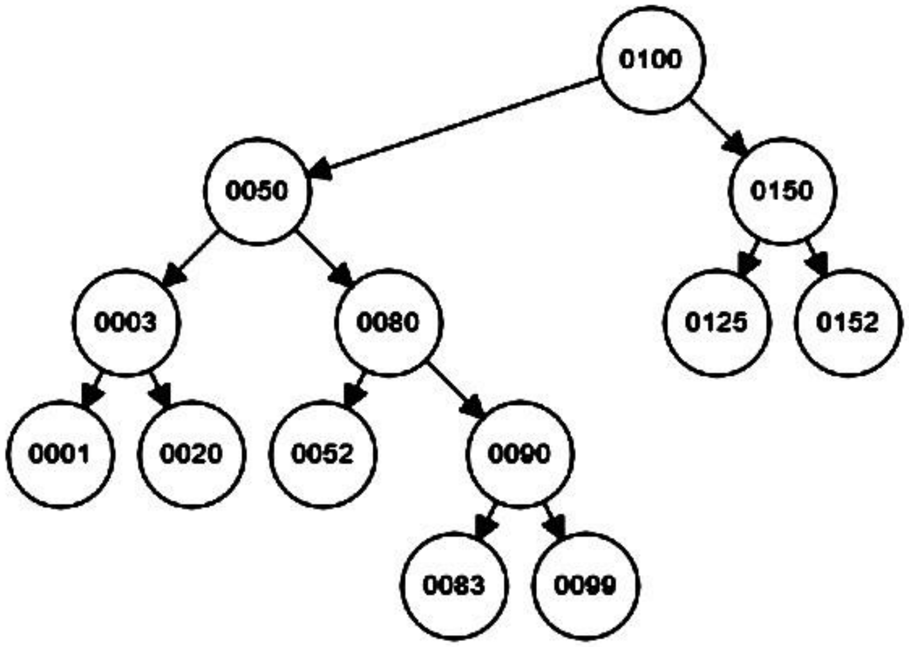


2. (15 pts) Given the following pre-order and in-order traversals, reconstruct the appropriate binary tree. **NOTE: You must draw a single tree that works for both traversals.**

Pre-order: A, C, B, I, F, G, J, D, H, K
In-order: I, B, C, F, J, G, A, D, K, H



3. **(30 pts)** Given the following binary tree (where nullptr height == -1):



a. **(3 pts)** What is the *height* of the tree?

The height of the tree is 4.

b. **(3 pts)** What is the *depth* of the *root* node?

The depth of the root node **is 0.** `

c. **(3 pts)** At which level is the *root* node?

The root node is on **level 0.**

d. **(3 pts)** What is the *depth* of node 0020?

The depth of node 0020 **is 3.**

e. **(3 pts)** List the values of all leaf nodes.

The leaf node values are **0001, 0020, 0052, 0083, 0099, 0125, 0152.**

f. **(3 pts)** What is the *height* of node 0020?

The height of node 0020 **is 0.**

g. **(12 pts – 4 pts/traversal)** Give the pre-order, in-order, and post-order traversals of this tree.

Pre-order: 0100, 0050, 0003, 0001, 0020, 0080, 0052, 0090, 0083, 0099, 0150, 0125, 0152
In-order: 0001, 0003, 0020, 0050, 0052, 0080, 0083, 0090, 0099, 0100, 0125, 0150, 0152
Post-order: 0001, 0020, 0003, 0052, 0083, 0099, 0090, 0080, 0050, 0125, 0152, 0150, 100

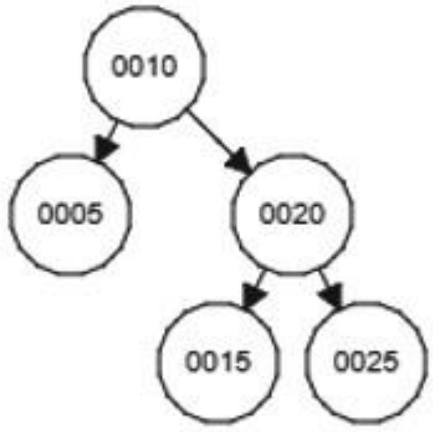
4. **a. (5 pts)** What is an AVL tree? Explain.

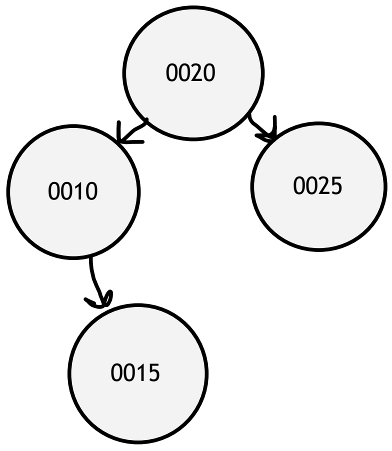
   An AVL tree is a binary search tree that is always balanced such that for every node in the BST the heights of the node's left and right subtrees differ by at most 1. This is maintained by forceful rotation, so whenever a new element is added or deleted from the tree, the tree rotates so it is balanced.

   b. (5 pts) What is the purpose of an AVL tree? Explain.

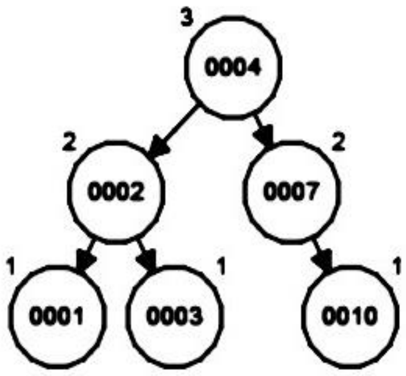
   The purpose of an AVL tree is to keep the worst-case depth of the BST to always be O(lg N). By keeping the tree balanced, the time complexity of tree functions is decreased since the worst-case time complexity is no longer O(n), so it is overall more efficient.

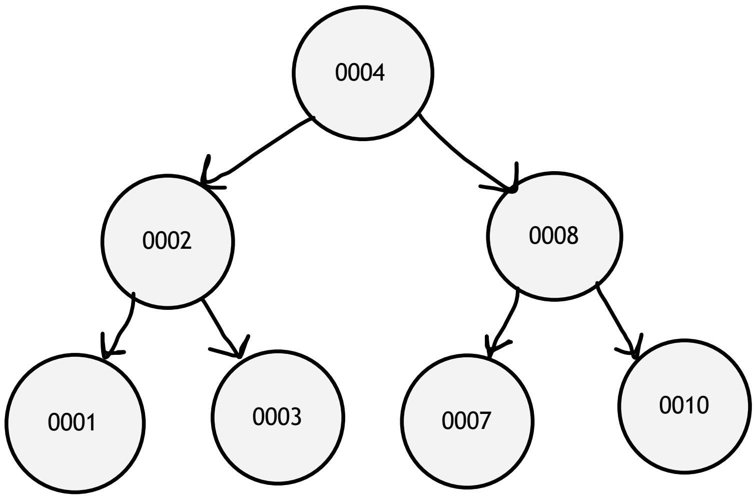
5. **(10 pts)** Remove 0005 from the following AVL tree; draw the resulting tree:



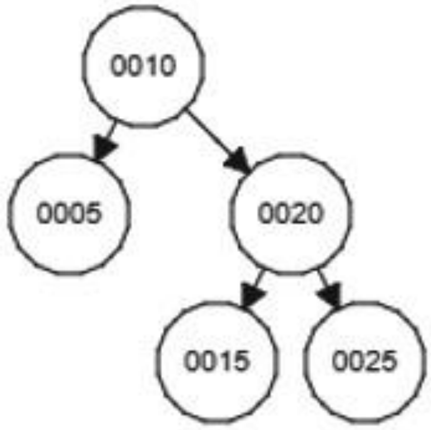


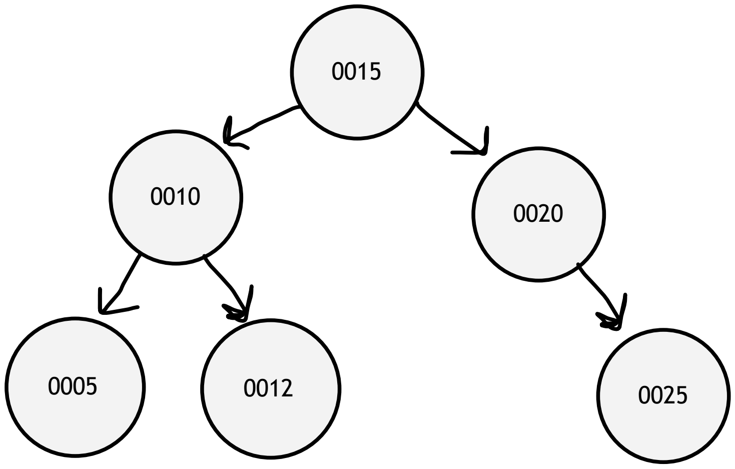
6. **(10 pts)** Insert the value 0008 into the following AVL tree; draw the resulting tree:





7. **(10 pts)** Insert the value 0012 into the following AVL tree; draw the resulting tree:





II. Submitting Written Homework Assignments:

1. On your local file system, create a new directory called HW3. Move your HW3.pdf file in to the directory. In your local Git repo, create a new branch called HW3. Add your HW3 directory to the branch, commit, and push to your private GitHub repo created in PA1.
2. Do not push new commits to the branch after you submit your link to Canvas otherwise it might be considered as late submission.
3. Submission: You must submit a URL link of the branch of your private GitHub repository to Canvas.

III. Grading Guidelines:

This assignment is worth 100 points. We will grade according to the following criteria:

- See above problems for individual point totals.