Name: _____

Solutions

1. (1 pt.) Write your name in the space provided above.

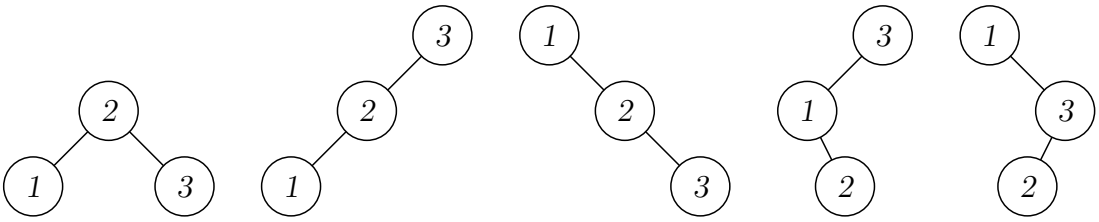
2. (6 pts.) Using the usual order on letters ($a < b < c < \cdots < y < z$), sort the following strings in **lexicographic order**. (List the strings in sorted order.)

abba	abatement	abaxial	abate	abaxile
abave	abatis	abature	abatua	abator

   ✍ *abate abatement abatis abator abatua abature abave abaxial abaxile abba*

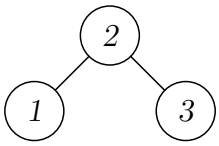
3. (8 pts.) Depict **all** the **binary search trees** containing exactly three keys: 1, 2, 3.

   ✍



4. (5 pts.) Depict **all** the **AVL trees** containing exactly three keys: 1, 2, 3.

   ✍



5. (5 pts.) What is the number of **binary trees** (**not necessarily** binary search trees) containing exactly three keys: 1, 2, 3? *Justify your answer.* (You do not need to depict all the trees.)

   ✍ *Let $b(n)$ denote the number of binary trees with n nodes, $n \geq 0$. There is exactly one empty binary tree and exactly one binary tree with one node, so $b(0) = b(1) = 1$. A binary tree with $n > 0$ nodes consists of a root, a left subtree with l nodes (for some $0 \leq l \leq n-1$) and a right subtree with $n-1-l$ nodes. There are n choices for the root node, $\binom{n-1}{l}$ choices for the nodes in the left subtree, $b(l)$ choices for the left subtree with these nodes, and $b(r)$ choices for the right subtree with the remaining nodes, giving:*

$$b(n) = n \sum_{l=0}^{n-1} \binom{n-1}{l} \cdot b(l) \cdot b(r)$$

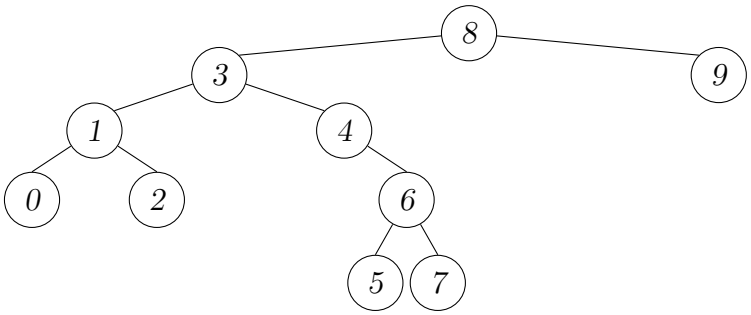
Using the base cases for $b(0)$ and $b(1)$, we may evaluate $b(2)$ and $b(3)$ as follows:

$$\begin{aligned} b(2) &= 2\left(\binom{1}{0} \cdot b(0) \cdot b(1) + \binom{1}{1} \cdot b(1) \cdot b(0)\right) \\ &= 2(1 \cdot 1 \cdot 1 + 1 \cdot 1 \cdot 1) \\ &= 4 \\ b(3) &= 3\left(\binom{2}{0} \cdot b(0) \cdot b(2) + \binom{2}{1} \cdot b(1) \cdot b(1) + \binom{2}{2} \cdot b(2) \cdot b(0)\right) \\ &= 3(1 \cdot 1 \cdot 4 + 2 \cdot 1 \cdot 1 + 1 \cdot 4 \cdot 1) \\ &= 30 \end{aligned}$$

6. (5 pts.) Depict the **binary search tree** resulting from the insertion of the following keys, in the order presented (into an initially empty tree). (You do not need to depict intermediate states of the tree.)

8 3 4 6 9 1 2 0 5 7

A

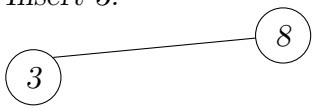


7. (10 pts.) Determine the **AVL tree** resulting from the insertion of the keys of Question 6, in the order presented there. **Depict the state of the tree after each insertion** and clearly **mark any rotations** that occur (as single or double rotations).

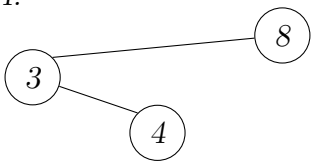
A

Insert 8:

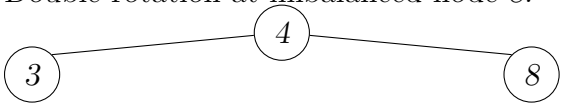
Insert 3:



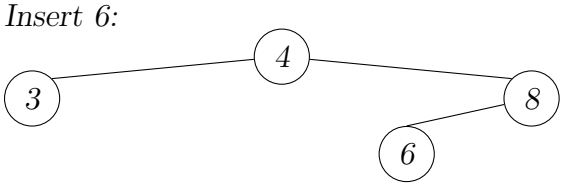
Insert 4:



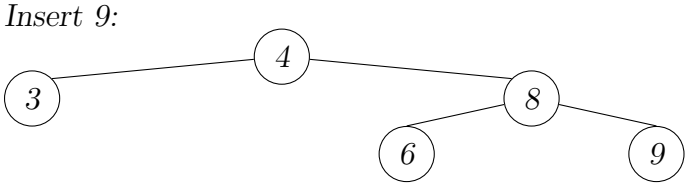
Double rotation at imbalanced node 8:



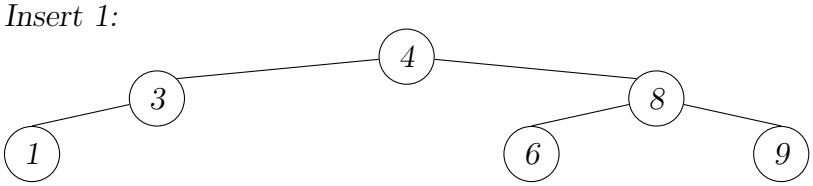
Insert 6:



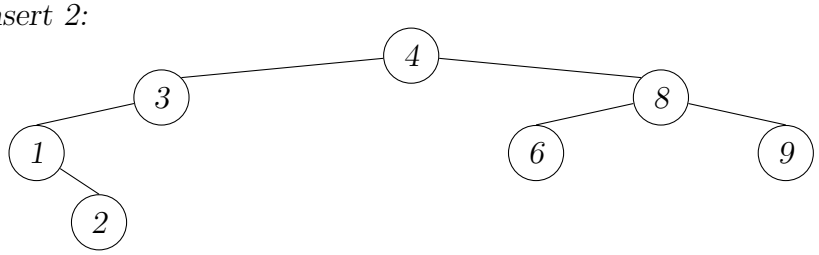
Insert 9:



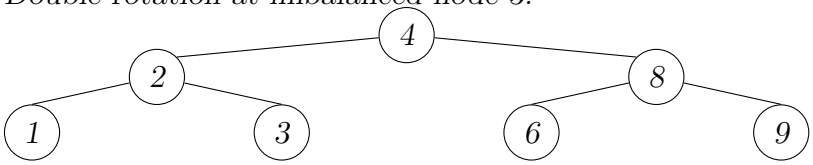
Insert 1:



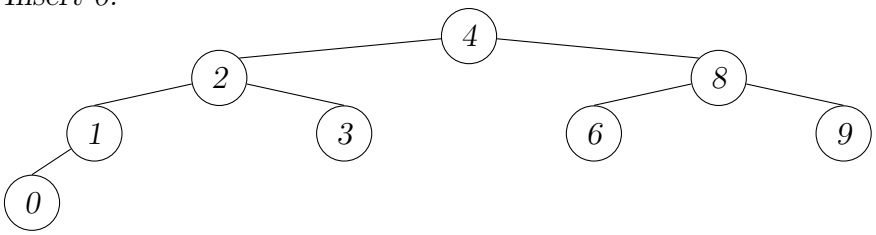
Insert 2:



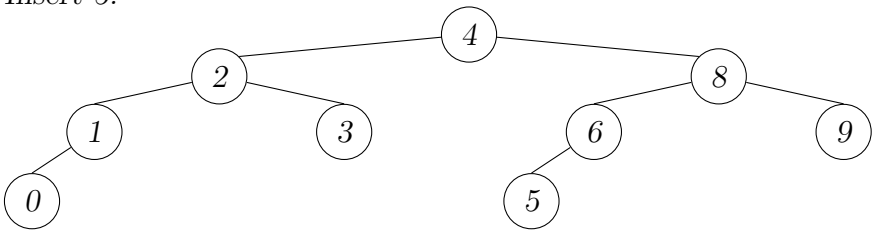
Double rotation at imbalanced node 3:



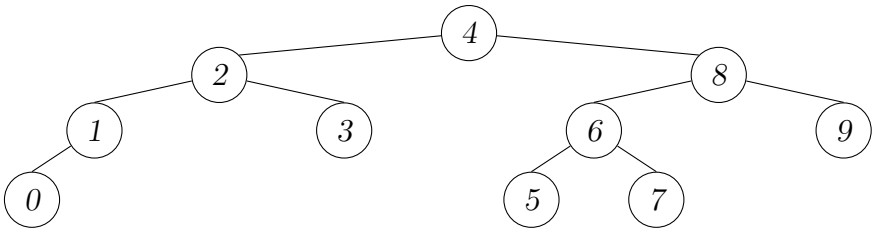
Insert 0:



Insert 5:



Insert 7:



8. (5 pts.) Determine which nodes in the tree of Question 6 are **AVL unbalanced** (do not satisfy the AVL balance condition). For each unbalanced node, list its **key** and the **heights of its left and right subtrees** below.

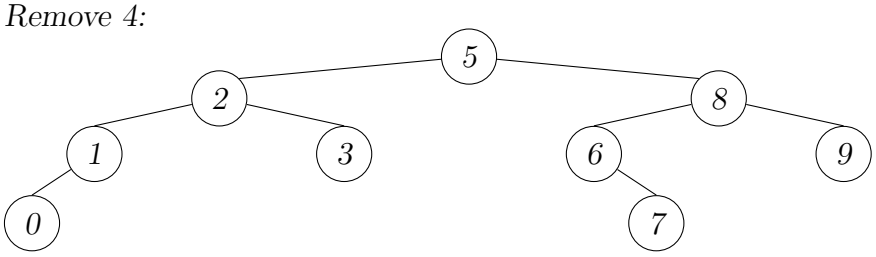
   ~~A~~

key	*left s.t. ht.*	*right s.t. ht.*
4	*-1*	*1*
8	*3*	*0*

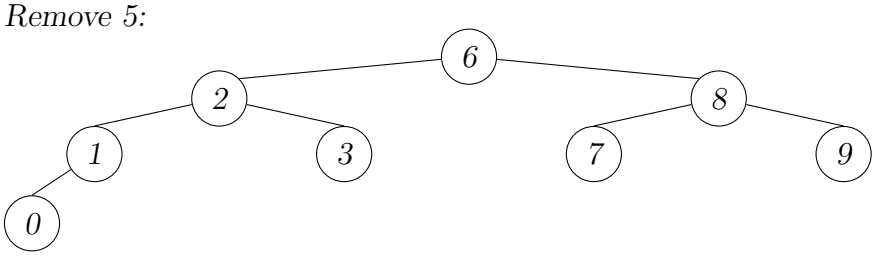
9. (5 pts.) Depict the result of removing the keys 4 and 5 (in that order) from the AVL tree of Question 7. **Depict the state of the tree after each removal** and clearly **mark any rotations** that occur (as single or double rotations).

   ~~A~~ *We will follow the convention of removing a key that is in an interior node with two children by replacing it with the smallest key in its right subtree. (A key in a leaf is removed by deleting the leaf, and a key in one-child node is removed by replacing it with the child's key and deleting/short-circuiting the child.)*

   Remove 4:



   Remove 5:



10. (10 pts.) ⋆ Derive an expression for the number of binary search trees containing exactly n keys (with all keys distinct). An answer in closed form (free of summations or other iterative operators) is preferred. There will be no credit for answers without clear explanations, and very little partial credit (for this question).