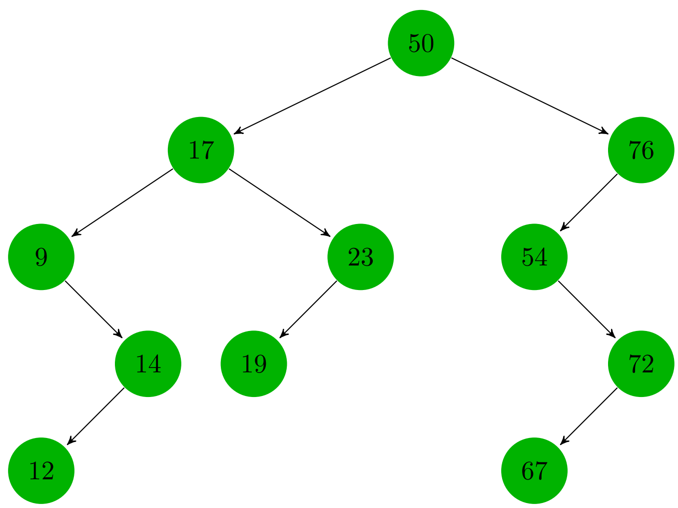
Weekly Assignment 12: Search Trees

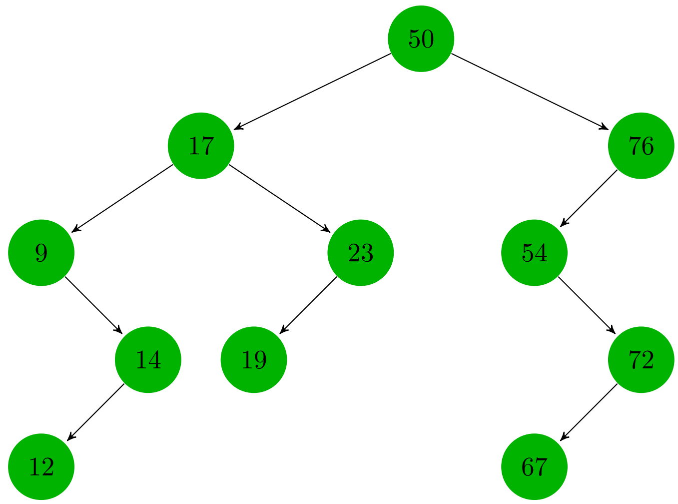
1. (a) How many binary search trees can you build with the following elements: $\{3, 5, 8, 12\}$? Draw them.
   (b) How many AVL trees are there with the same elements $\{3, 5, 8, 12\}$? Draw them (or better, just point them out in your answer to the previous question).

2. Consider the following binary search tree:



   Delete the nodes with keys 9, 17 and 50, in this order, using the procedure explained in the lecture/textbook. Draw the binary search trees obtained after deleting the first, second and third node.

3. Consider the following binary search tree:



Show how this tree can be turned into an AVL tree using four rotations. Draw the binary search tree obtained after each rotation.

4. Describe an efficient algorithm which takes as input a binary search tree T_1 (with n nodes) and another binary search tree T_2 (with m nodes) over the integers, and combines them into a new binary search tree $T_1 + T_2$ which contains all the elements from both T_1 and T_2. Different nodes with the same key should not be merged, so that $T_1 + T_2$ has $n + m$ nodes. Moreover, the new tree $T_1 + T_2$ should be balanced, that is, height $\mathcal{O}(\lg(n + m))$.

   Explain your answer, and include a correctness and complexity analysis.

5. Consider a binary search tree implementation that maintains an attribute *size* such that, for each node x, $x.size$ gives the number of nodes in the subtree with root x. We define $NIL.size = 0$. Give pseudocode for an algorithm with time complexity $\mathcal{O}(height)$ that computes, for any node x and key k, RANK(x, k): the number of nodes in the subtree of x with key at most k.

   Explain you algorithm and discuss its complexity.