0. Reminder: Midterms is scheduled for October 27, Monday in class.

1. **On AVL Trees.**
   a. Two binary trees that contain the same keys can look completely different. For example, one can be long and thin, while the other can be short and fat. What affects their topology (i.e., their “shape”) is the sequence in which the items are inserted. Now, we want to know if this is still the case for AVL trees. After all, AVL trees can’t be long and thin anymore.

   Now, we want to know if this is still the case for AVL trees. After all, AVL trees can’t be long and thin anymore.

   a1. First, insert the following sequence of keys into an initially empty AVL tree: 1, 2, 3, 4, 5, 6, 7.

   a2. Now, address the problem – if you had started off with a different sequence of the same set of keys, will the topology of the tree stay the same or different? Why?

   b. Another thing we noted about AVL trees is that a single `removeElement` step (unlike a single `insertItem` step) may require several rotations in order for the height balance property to be fixed. Present an AVL tree where the removal a single item causes at least three rotations for the height balance property to be restored. Indicate what item has to be removed and then show the three fixes that will fix the problem.

2. **R-3.15.** For a, b, and c, do the first 5 keys only. In part b, make use of the tree produced at the end of part a. In part c, make use of the tree produced at the end of part b.

3. **C-3.11.** For this problem, assume each node v has the field `v.size`.

4. As in the previous homework, let S and T be two ordered dictionaries each containing n items. Both are implemented via sorted arrays and do not have any keys in common. Assume that `FindMedians(S, T)` is an algorithm that outputs the lower as well as the upper medians in the union of S and T. You are now interested in finding the kth smallest key in the union of S and T. Describe an algorithm that solves this problem by making use of `FindMedians(S, T)` (Keep in mind that k can be larger than n.) If `FindMedians(S, T)` runs on $O(\log n)$ time, what is the running time of your new algorithm?