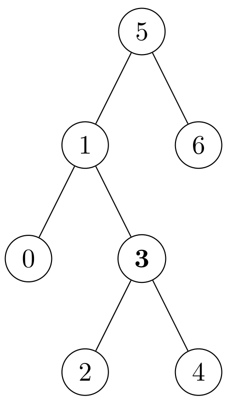
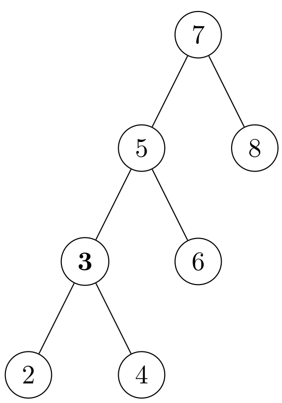
CS 161 Fall 2017: Section 3

Tree Rotations

Modify the following trees via a sequence of rotations so that 3 is at the root of each tree. Draw out the intermediate stages (result after each rotation step).



(a)



(b)

Randomly Built BSTs

In this problem, we prove that the average depth of a node in a randomly built binary search tree with n nodes is $O(\log n)$. A *randomly built binary search tree* with n nodes is one that arises from inserting the n keys in random order into an initially empty tree, where each of the $n!$ permutations of the input keys is equally likely.

Let $d(x,T)$ be the depth of node x in a binary tree T (the depth of the root is 0). Then, the average depth of a node in a binary tree T with n nodes is

$$\frac{1}{n}\sum_{x\in T} d(x,T) \ .$$

(a) Let the *total path length* $P(T)$ of a binary tree T be defined as the sum of the depths of all nodes in T, so the average depth of a node in T with n nodes is equal to $\frac{1}{n}P(T)$. Show that $P(T) = P(T_L) + P(T_R) + n - 1$, where T_L and T_R are the left and right subtrees of T, respectively.

(b) Let $P(n)$ be the expected total path length of a randomly built binary search tree with n nodes. Show that $P(n) = \dfrac{1}{n}\displaystyle\sum_{i=0}^{n-1}(P(i) + P(n-i-1) + n - 1)$.

(c) Show that $P(n) = O(n \log n)$. You may cite a result previously proven in the context of other topics covered in class.

(d) Design a sorting algorithm based on randomly building a binary search tree. Show that its (expected) running time is $O(n \log n)$. Assume that a random permutation of n keys can be generated in time $O(n)$