Your name
CS 261, Spring 2021
Homework 8
Due Monday, May 24

1. A sequence of parentheses defines a valid ordered tree if the parentheses are nested. This can be checked by maintaining an integer counter of the nesting level (initially zero), and looping over the sequence one character at a time, adding one to the counter for an open-parenthesis character and subtracting one for a close-parenthesis counter. The sequence is valid if the counter is zero at the end without ever having become negative. Describe a similar counter-based algorithm for checking whether a sequence of 1's and 0's defines a valid binary tree, using the binary tree representation described in the notes.

■

2. If an array contains the numbers from 1 to n, it may be arranged into any of $n!$ different permutations. However, we have seen that the number of different Cartesian trees for an array of n items is less than 4^n, a much more slowly-growing function. Therefore, for large n, there must be many pairs of permutations that have the same Cartesian tree as each other. However, some permutations have a Cartesian tree that is unique, different from the Cartesian trees of all other permutations. For instance, if the array is in sorted order, it has a unique Cartesian tree: no other ordering but the sorted ordering can produce the same tree.

As a function of n, how many permutations have unique Cartesian trees?

(Hint: For a permutation to have a unique Cartesian tree, where must the minimum element be? Then apply induction to the remaining elements.)

■

3. In any tree T, a node x is an ancestor of another node y if and only if x occurs both before y in a preorder traversal of the tree, and after y in a postorder traversal. Therefore, it can be useful to store the nodes of a tree T in two list-ordering data structures, one for the preorder and one for the postorder. These two structures allow for constant time queries that answer the question: is x an ancestor of y?

Suppose that we have a tree T represented in two list-ordering structures in this way, and that we then delete a node x from both of these structures. What change in T corresponds to this change in the ordered lists?

(For instance, if x is a leaf, then removing x from the lists corresponds exactly to removing x from the tree, and leaving the rest of the tree unchanged. However, your answer should not assume that x is a leaf. You may assume that x is not the root.)

■

4. Suppose we have a rooted tree T in which each node is labeled by its depth, that is, its distance from the root. We wish to answer queries whose arguments are two nodes x and y and a number k, and that test whether the distance from x to y is less than or equal to k, returning True if it is and False otherwise.

(a) Describe how to answer these queries using a single lowest common ancestor query in T.

(b) Describe how to answer these queries using two level ancestor queries in T.

■