Patrick El-Hage Assignment 4

Assume an n × n matrix A is given, containing only 1's and 0's, such that, in each row, all 1's come before all 0's. Give an O(n log n) algorithm to count all 1's in A.

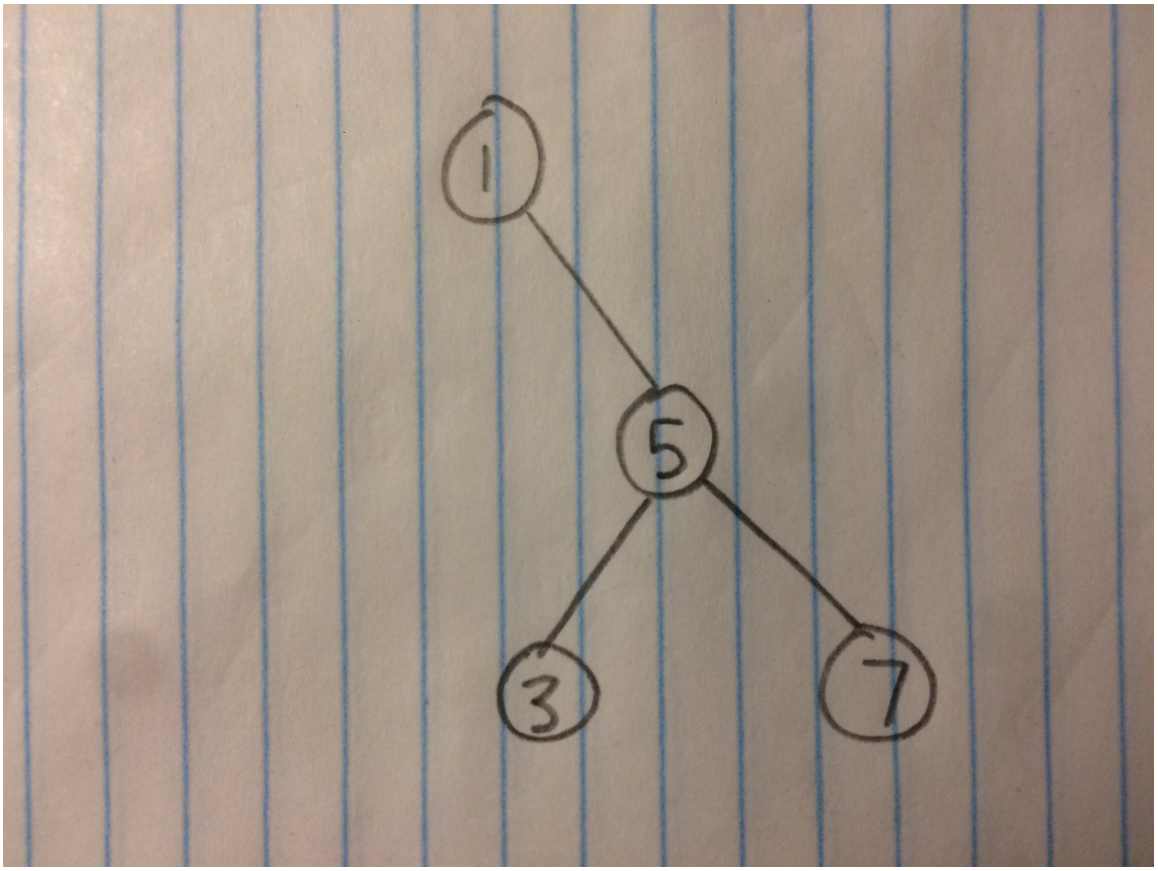
To be able to count all 1s in the matrix, perform a binary search for each row, searching for the last 1 in the row. By keeping track of the indexes while searching, since we know all 1's come before all 0's, we can derive the total number of 1's in the row via the last 1's index. Performing a logn operation for each row would make for a nlogn algorithm

Let A and B be two sequences of n integers each. Given an integer m, describe an O(n log n) algorithm to determine if there is an element a of A and an element b of B such that m = a + b.

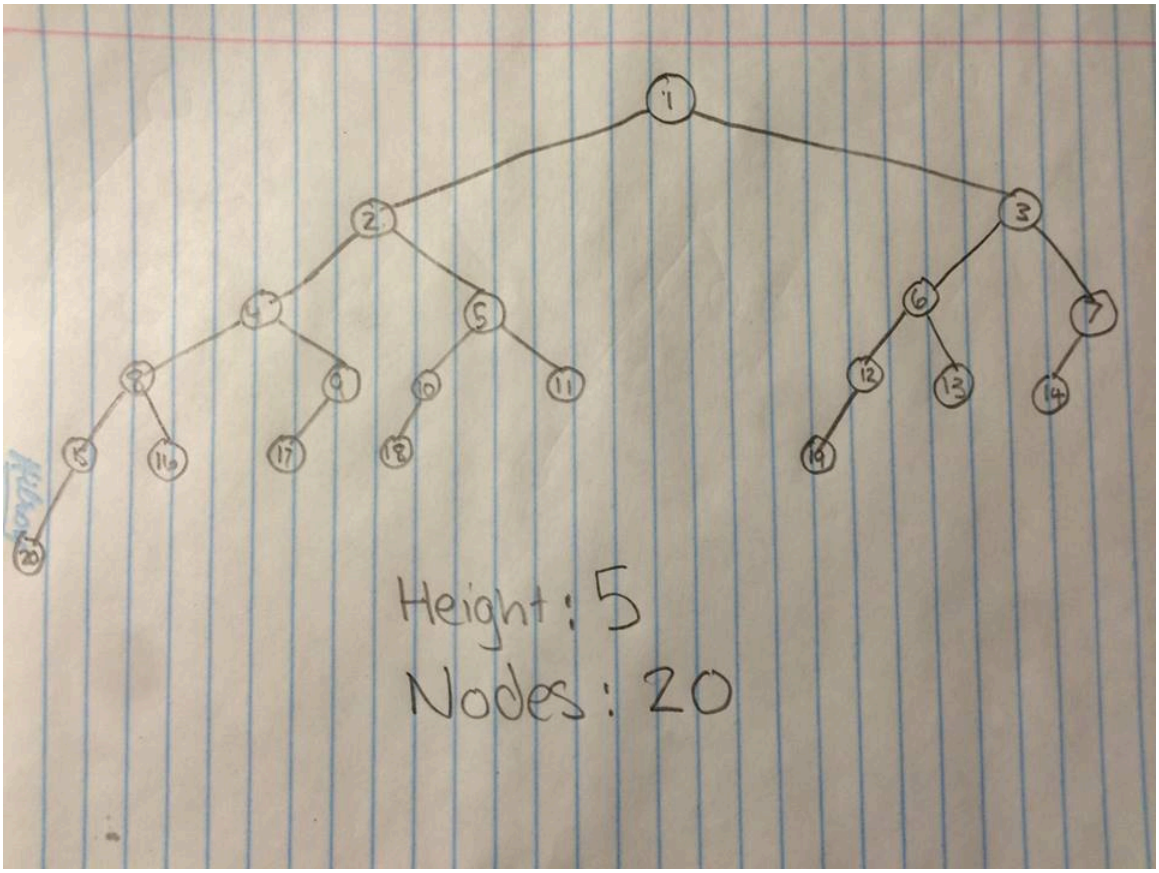
Do a merge sort on sequenceA, and then go through sequenceB, item by item (i) and then perform a binary search on sequenceA to see if there exists a value in A such that m – sequenceB[i] = a

Professor X thinks he has discovered a remarkable property of binary search trees. Assume that the search for a key k ends up in a leaf. Consider the sets A, containing the keys to the left of the search path, B, the key on the search path, and C, the keys to the right of the search path. Professor X claims that for any $a \in A, b \in B, c \in C$, we have $a \leq b \leq c$. Give a smallest possible counterexample to this claim.

If we were searching for the key 7, the path to key 7, and members of set B, would be: 1, 5, 7.
Key 3, part of set A, is to the left of the search path, but is not less than all members of set B. $3 > 1$, so therefore Professor X's claim is incorrect.



What is the minimum number of nodes an AVL tree of height 5 can have? Draw one such AVL tree. Explain why it has the minimum possible number of nodes.
The minimum number of nodes an AVL tree of height of 5 can have is 20 nodes.



In order to make it to height 5, each sub tree's left and right trees had to only have a difference of height = 1. As you can see, if one were to remove any single node in the above tree, the tree would be unbalanced. For example, if I removed node 18, node 4 and node 5 would have a difference of height = 2. If I removed node 19, then nodes 2 and 3 would have difference of height = 2.

Every node in the tree must be there.

Write a Python program compareSort to compare several sorting algorithms

My results:
quickSort: 0.1121177
mergeSort: 0.0045908
bubbleSort: 0.0449889

My quick sort is the slowest… I must have bugs but I'm running out of time!