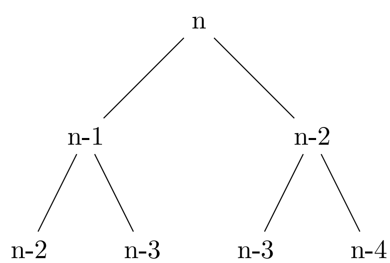
Caio Brighenti
COSC 302 - Analysis of Algorithms – Spring 2019
Lab 5

3. Recursion tree method.
   The recurrence is $T(n) = T(n-1) + T(n-2)$. Thus, the recursion tree is:



   Note that this tree is not balanced, but we will assume it is since we are showing an upper bound with $O(f(n))$ notation. We can see that the slowest decreasing element of the tree, the leftmost node, decreases by 1 on each level. We will call the total number of levels i. We assume the function recurses until 0 – therefore since n decreases by 1 each level, we will have n levels, thus $i = n$.
   Next, we find the number of nodes per level. We can see that the first level has 2^0 nodes, the second 2^1 nodes, and the third 2^2. Thus, if we let j represent the current level, then the number of nodes for each level is equal to $2^{(j-1)}$. Thus, we can represent the total amount of nodes (or calls) as, since we know we have i levels:
   $$2^0 + 2^1 + 2^2 + \cdots + 2^{i-1}$$
   This can be simplified to $2^i - 1 = 2^n - 1$. Thus, we solution to the recurrence is:
   $$T(n) = 2^n - 1 = O(2^n)$$