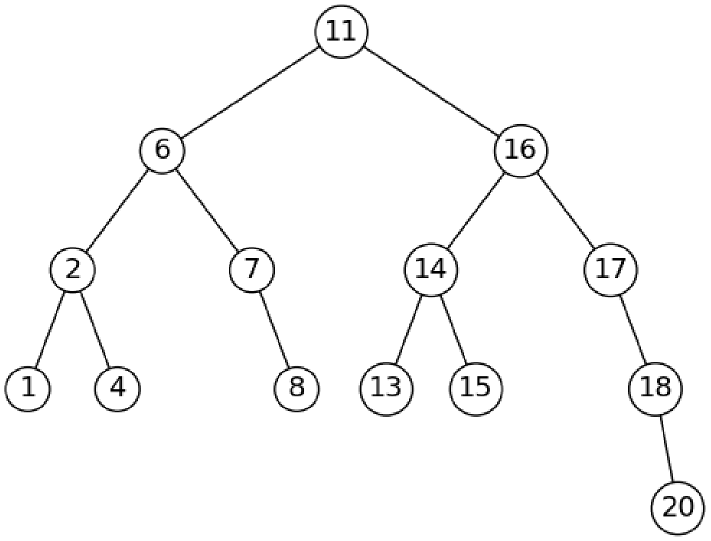
CS2302 - Data Structures

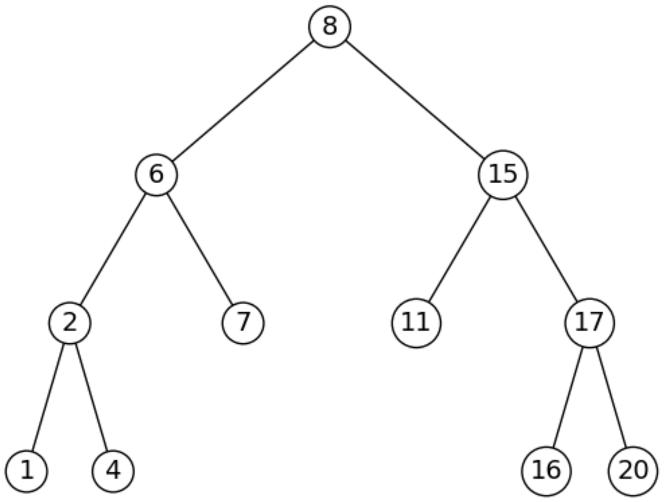
Spring 2020

Exam # 2 - Binary Search Trees

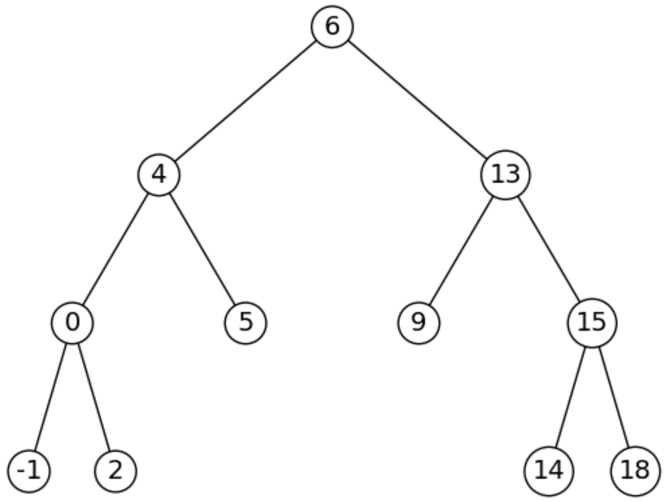
1. The function *count_odd(T)* is supposed to receive a binary search tree T and return the number of items in T that are odd. For example, if T is the tree in the figure, *count_odd(T)* should return 6. However, the function does not work. Fix it so it produces the right results.



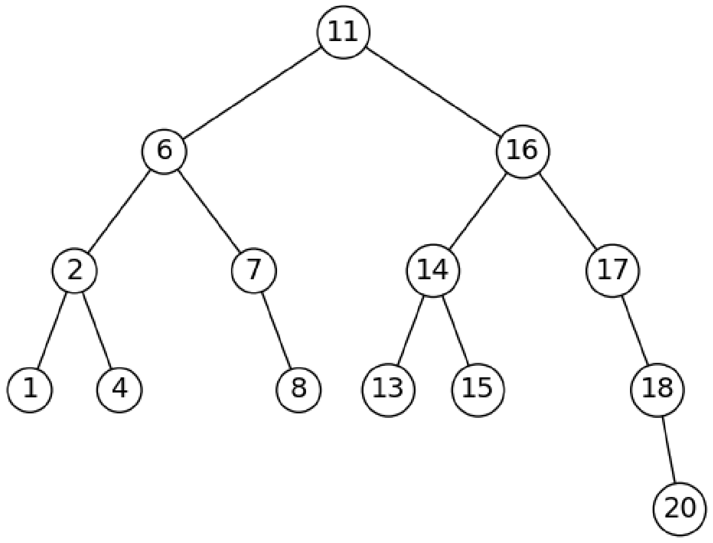
2. A binary search tree is full if all of its nodes have zero or two children. For example, the tree in the figure above is not full, since nodes 7 and 17 have 1 child, while the tree in the figure is below is full, since all nodes have either 0 children (nodes 2, 7, 14 and 17) or 2 children (nodes 6, 11, and 16). Write the function *is_full(T)* that receives a binary search tree T and determines if it is full.



3. Write the function *subtract_n(T,n)* that receives a binary search tree T and an integer n and subtracts n to every data item in the tree. For example, if T is the tree in the figure above, after executing *subtract_n(T,n)* T should be the tree below.



4. A path from the root to a node in a binary search tree can be encoded by a string of characters 'L' and 'R', where 'L' means 'go left' and 'R' means 'go right'. For example, the path from the root to node 14 in the tree below would be encoded by the string 'RL' (go right (16), go left (14)), the path to 1 would be encoded by the string 'LLL', and the path to the root would be encoded by '' (the empty string). Write the function *follow_path(T,s)* that receives a binary search tree T and a string s and returns the item stored in the node that you would reach by following the path encoded in s. You may assume that s contains only valid characters ('L' or 'R'). If the path encoded does not lead to any node (for example 'LLLLL' in the tree below) your function should return None.


11
6
16
2
7
14
17
1
4
8
13
15
18
20