P1

Main idea:

You are given a binary tree T with n many nodes. Your task is to find if T is balanced or not.

Describe a brute force algorithm to solve this problem. What is its running time in terms

of n?

Solution: in my opinion brute force algorithm for this problem will look like this:

You will need exactly 3 loops, which first of them will resposible to take root element from tree, after that second loop will engaged to take one of the sides (subtrees: left&right), next loop will look for consisting of leafs in both sides until function will return false or otherwise ending of tree after that you will get return true.

Running time of such algorithm will roughly O(n^3) or more

P2

In divide and conquer we recurse down to the bottom of the tree to check if the tree is balanced and also check for differences in subtree height. In the root (if the tree isn't rooted, pick any node).  
In each step, try to descend into the child node that has the largest subtree (the number of nodes “below” it is the biggest).If doing that would make the number of nodes “above” bigger than n/2, stop, otherwise continue with that child. The extra complexity comes as we go higher back up the tree towards the root, the get\_height still recurses down the tree to compute the height. Therefore, traveling down the tree once more --> O(n^2).This algorithm should be O(log n) if the tree is reasonably balanced and we have sizes of subtrees precomputed for each node. If one of those conditions doesn't apply, it would be O(n)

P3

def is\_balanced2(root):

if root is None:

return True

else:

return (abs(get\_height(root.left) - get\_height(root.right)) < 2) and is\_balanced(root.right) and is\_balanced(root.left)

def get\_height(root):

if root is None:

return 0

else:

return max(get\_height(root.left), get\_height(root.right)) + 1

i recommend to see these links about binary tree which will really help

<https://medium.com/the-renaissance-developer/learning-tree-data-structure-27c6bb363051>

https://www.laurentluce.com/posts/binary-search-tree-library-in-python/