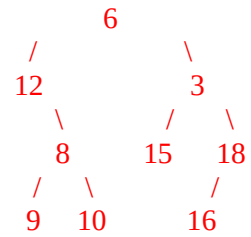
COP 3502 Section 4 Exam 2B – Binary Trees, AVL Trees, Binary Heaps Solutions
(Thursday 11/16/2023)

1) (10 pts) The depth of a node in a binary tree is the number of links one needs to follow from the root to get to that node. Write a recursive function that determines the sum of the depths of all of the leaf nodes in the tree. The function will take in both a pointer to the node of the tree upon which it's being called as well as an integer, depth, representing the current depth that node is in the tree. The initial call to the function is given below, as well as the binary tree node struct to use for the problem:

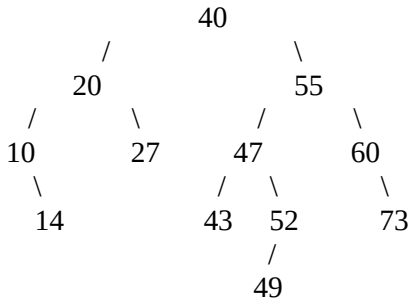
```
typedef struct treenode {
    int data;
    struct treenode *left;
    struct treenode *right;
} treenode;

// Call: int res = sumleafdepths(tree, 0);
// where tree is pointing to the root of a binary tree.

int sumleafdepths(treenode* root, int depth) {

    if (root == NULL) return 0;                                  // 2 pts

    if (root->left == NULL && root->right == NULL)               // 2 pts
        return depth;                                            // 1 pt


    return sumleafdepths(root->left, depth+1) +                  // 1 pt return
           sumleafdepths(root->right, depth+1);                  // 2 pts each
                                                                 // rec call
}
```

2) (5 pts) The preorder traversal of a binary tree is **6, 12, 8, 9, 10, 3, 15, 18, 16.**
The inorder traversal of that same tree is **12, 9, 8, 10, 6, 15, 3, 16, 18.** Draw the tree below. In order to get full credit, your left and right links must be unambiguous.

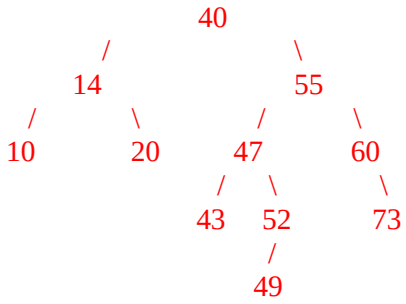


Grading: 1 pt 6 root,
1 pt to have 12 in the correct spot,
1 pt to have 3 in the correct spot,
2 pts total for the rest

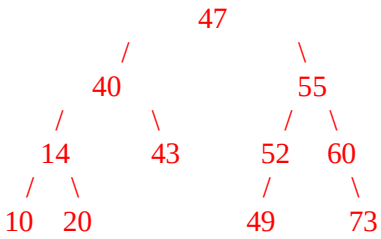
3) (10 pts) Show the result of deleting 27 from the AVL Tree below. Please draw a box around the temporary result after the first rebalance and a second box around your final answer (after the second rebalance).



After first rebalance:



After final rebalance:



Grading: 4 pts first picture (give 1 pt if they have removed 27)
6 pts second picture (1 pt 47 root, 1 pt 40 left, 1 pt 55 right, 3 pts rest give partial as needed award only a whole number of points)

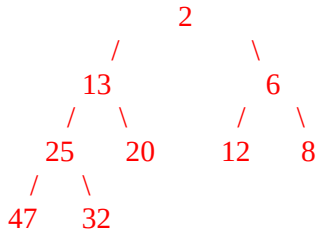
4) (9 pts) Arrays of integers (of the same length) can be sorted in lexicographical order. To compare two arrays arr1 and arr2 of the same length in this order, find the lowest index i such that arr1[i] ≠ arr2[i]. If arr1[i] < arr2[i], then arr1 comes first. Otherwise, arr2 comes first. Consider creating a valid binary (minimum) heap with the following integer values:

2, 6, 8, 12, 13, 20, 25, 32, 47

Recall that in code, binary heaps are stored in arrays, with the root of the tree being stored in index 1, and index 0 being unused. Of all possible valid binary heaps that could be created with the numbers above, draw the one that **comes last in lexicographical order**, when comparing heaps by their array representations. Draw both the tree version of the heap AND show the array version of this particular heap.

Index	1	2	3	4	5	6	7	8	9
Value	2	13	6	25	20	12	8	47	32

Drawing below:



Grading: Automatic 0/9 if the drawing is not a valid min heap
If drawing and array aren't consistent, max 3 pts out of 9
Otherwise, 1 pt per correct value in the array.
So, if the drawing is consistent with the array, just grade the array.
If it is not consistent, assign the grade minimum(3, arraygrade).

5) (1 pt) The UCF President lives in the Burnett House. What is the last name of the couple that donated money that was used to get the house built?

Burnett (Give to all)