1) (10 pts) DSN (Binary Trees)

Consider using the following struct definition for a node of a binary search tree:

```
typedef struct node {
    int data;
    int height;
    struct node* left;
    struct node* right;
} node;
```

Assume that a binary search tree has been built with the data values in each struct filled in, but the heights are uninitialized. Write a **void recursive** function, `assignHeights`, **with no helper** functions, which takes in a pointer, `root`, to the root of a binary search tree, and assigns the height component of each node in the subtree pointed to by `root` to its correct height in the tree. Recall that the height of a leaf node is 0, and that more generally, the height of a node is the maximum number of links (left or right) to follow from that node to any leaf node in that subtree. (If root is NULL, then no action should be taken.)

```
void assignHeights(node* root) {

    if (root != NULL) {
        assignHeights(root->left);
        assignHeights(root->right);
        int big = -1;
        if (root->left != NULL) big = root->left->height;
        if (root->right!= NULL && root->right->height > big)
            big = root->right->height;
        root->height = big + 1;
    }
}
```

Grading: 2 pts recursive call left before root height assigned (1 pt if after)
2 pts recursive call right before root height assigned (1 pt if after)
2 pts – attempting to determine max of left and right
2 pts – avoiding NULL ptr error while getting max
2 pts – setting root height to max of left and right plus 1