Tree: Height of a Binary Tree

by vatsalchanana

The height of a binary tree is the number of edges between the tree's root and its furthest leaf. This means that a tree containing a single node has a height of 0.

Complete the *getHeight* function provided in your editor so that it returns the height of a binary tree. This function has a parameter, ***root***, which is a pointer to the root node of a binary tree.
Note -The Height of binary tree with single node is taken as zero.

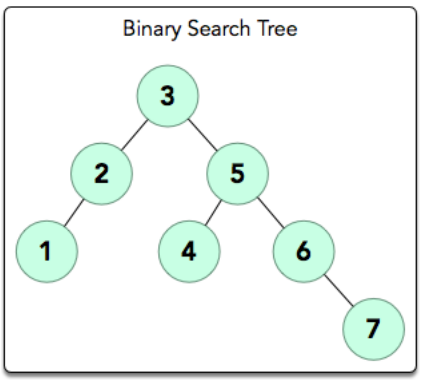
Input Format

You do not need to read any input from stdin. Our grader will pass the root node of a binary tree to your *getHeight* function.

Output Format

Your function should return a single integer denoting the height of the binary tree.

Sample Input



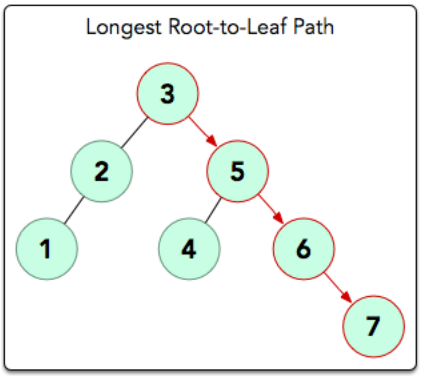
Note: A *binary search tree* is a binary tree in which the value of each parent node's left child is less than the value the parent node, and the value of the parent node is less than the value of its right child.

Sample Output

```
3
```

Explanation

The longest root-to-leaf path is shown below:



There are 4 nodes in this path that are connected by 3 edges, meaning our binary tree's $height = 3$. Thus, we print 3 as our answer.

Submissions:67722

Max Score:10

Difficulty: Easy

Rate This Challenge:

More

Current Buffer (saved locally, editable) C++

```
#include ↔

using namespace std;

class Node{
    public:
        int data;
        Node *left;
        Node *right;
        Node(int d){
            data = d;
            left = NULL;
            right = NULL;
        }
};

class Solution {
    public:
        Node* insert(Node* root, int data) {
            if(root == NULL) {
                return new Node(data);
            }
            else {
                Node* cur;
                if(data <= root->data){
                    cur = insert(root->left, data);
                    root->left = cur;
                }
                else{
                    cur = insert(root->right, data);
                    root->right = cur;
                }

                return root;
            }
        }

/*The tree node has data, left child and right child
class Node {
    int data;
```

```
    Node* left;
    Node* right;
};

*/
    int height(Node* root) {
        // Write your code here.
    }


}; //End of Solution

int main() {
    Solution myTree;
    Node* root = NULL;
    int t;
    int data;

    cin >> t;

    while(t-- > 0){
        cin >> data;
        root = myTree.insert(root, data);
    }
    int height = myTree.height(root);
    cout << height;

    return 0;
}

```

Line: 15 Col: 1

Upload Code as File | Test against custom input | Run Code | Submit Code

Join us on IRC at #hackerrank on freenode for hugs or bugs.

Contest Calendar | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature