Your Day 22: Binary Search Trees submission got 30.00 points. Tweet

You are now 7 challenges away from the gold level for your 30 days of code badge.

Try the Next Challenge | Try a Random Challenge

Day 22: Binary Search Trees

by vatsalchanana

Problem Submissions Leaderboard Discussions Editorial Tutorial

Objective

Today, we're working with Binary Search Trees (BSTs). Check out the Tutorial tab for learning materials and an instructional video!

Task

The height of a binary search tree is the number of edges between the tree's root and its furthest leaf. You are given a pointer, , pointing to the root of a binary search tree. Complete

Submitted 38483 times
Max Score 30

Need Help?

the *getHeight* function provided in your editor so that it returns the height of the binary search tree.

Input Format

The locked stub code in your editor reads the following inputs and assembles them into a binary search tree:
The first line contains an integer, , denoting the number of nodes in the tree.
Each of the subsequent lines contains an integer, , denoting the value of an element that must be added to the BST.

Output Format

The locked stub code in your editor will print the integer returned by your *getHeight* function denoting the height of the BST.

Sample Input

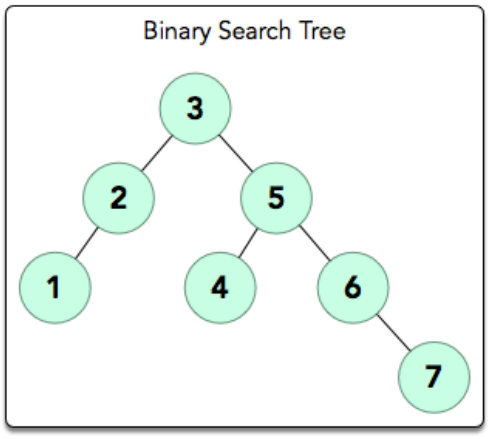
```
7
3
5
2
1
4
6
7
```

Sample Output

```
3
```

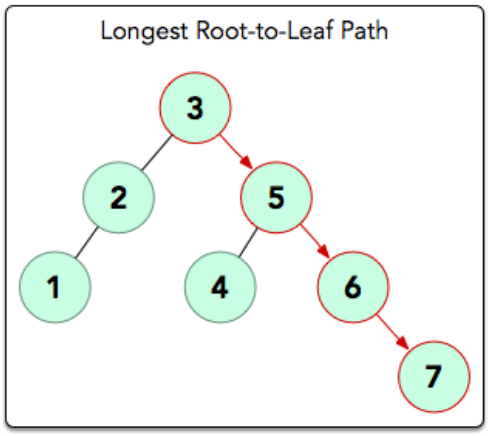
Explanation

The input forms the following BST:



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

View Tutorial
View Discussions
View Editorial Solution
View Top Submissions
Rate This Challenge:
Download problem statement
Download sample test cases
Suggest Edits



There are nodes in this path that are connected by edges, meaning our BST's . Thus, we print as our answer.

Current Buffer (saved locally, editable)

Python 3

```
class Node:
    def __init__(self,data):
        self.right=self.left=None
        self.data = data
class Solution:
    def insert(self,root,data):
        if root==None:
            return Node(data)
        else:
            if data<=root.data:
                cur=self.insert(root.left,data)
                root.left=cur
            else:
                cur=self.insert(root.right,data)
                root.right=cur
        return root
    def getHeight(self,root):
        #Write your code here
        if root == None:
            return 0
        elif root.left == None and root.right == None:
            return 0
        else:
            return 1 + max(self.getHeight(root.left), self.getHeight(root.right))

T=int(input())
myTree=Solution()
root=None
```

```
for i in range(T):
    data=int(input())
    root=myTree.insert(root,data)
height=myTree.getHeight(root)
print(height)
```

Line: 22 Col: 21

Test against custom input

Run Code | Submit Code

Congrats, you solved this challenge!

Challenge your friends:

Test Case #0 | Test Case #1 | Test Case #2

Next Challenge

You've earned 30.00 points. You are now 7 challenges away from the gold level for your 30 days of code badge.