Is This a Binary Search Tree?

by saikiran9194

Problem Submissions Leaderboard Discussions Editorial

For the purposes of this challenge, we define a binary tree to be a binary search tree with the following ordering requirements:

- The $data$ value of every node in a node's left subtree is *less than* the data value of that node.
- The $data$ value of every node in a node's right subtree is *greater than* the data value of that node.

Given the root node of a binary tree, can you determine if it's also a binary search tree?

Complete the function in your editor below, which has 1 parameter: a pointer to the root of a binary tree. It must return a *boolean* denoting whether or not the binary tree is a binary search tree. You may have to write one or more helper functions to complete this challenge.

Input Format

You are not responsible for reading any input from stdin. Hidden code stubs will assemble a binary tree and pass its root node to your function as an argument.

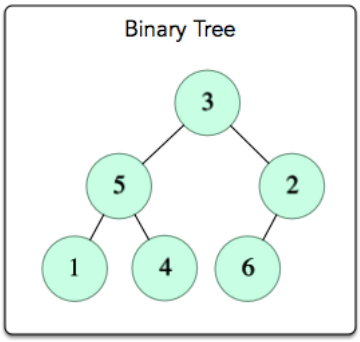
Constraints

- $0 \leq data \leq 10^4$

Output Format

You are not responsible for printing any output to stdout. Your function must return *true* if the tree is a binary search tree; otherwise, it must return *false*. Hidden code stubs will print this result as a *Yes* or *No* answer on a new line.

Sample Input



Sample Output

```
No
```

Submissions:16248

Max Score:30

Difficulty: Medium

Rate This Challenge:

More

Current Buffer (saved locally, editable) C++

```
/* Hidden stub code will pass a root argument to the function below. Complete the function to solve
the challenge. Hint: you may want to write one or more helper functions.

The Node struct is defined as follows:
   struct Node {
      int data;
      Node* left;
      Node* right;
   }


*/
/*
bool bstUtil1(Node* root, Node *l=NULL, Node* r=NULL)
{
    if(root==NULL)
        return true;

    if(l!= NULL and root->data < l->data)
        return false;
    if(r!= NULL and root->data > r->data)
        return false;

    return (bstUtil(root->left,l,root) and bstUtil(root->right, root,r) );
}

*/


bool checkBST(Node* root) {

      //return bstUtil(root);
        static Node *prev = NULL;



        if(root)
        {
            if(!checkBST(root->left))
                return false;

            if(prev!=NULL && prev->data >= root->data )
                return false;

            prev=root;

            if(!checkBST(root->right))
                return false;

        }


        return true;

   }

```

Line: 1 Col: 1

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