Check for BST

Easy | Accuracy: 25.37% | Submissions: 425K+ | Points: 2

Given the root of a binary tree. Check whether it is a BST or not.

Note: We are considering that BSTs can not contain duplicate Nodes.

A **BST** is defined as follows:

- The left subtree of a node contains only nodes with keys **less than** the node's key.
- The right subtree of a node contains only nodes with keys **greater than** the node's key.
- Both the left and right subtrees must also be binary search trees.

Example 1:

Input:

2

C++ (g++ 5.4) | Average Time: 30m | Start Timer

```
// } Driver Code Ends
class Solution
{
    public:
    //Function to check whether a Binary Tree is BST or not.
    bool check(Node* root, int max, int min){
        if(!root)
          return true;
        if(root->data <= min || root->data >= max)
          return false;

        return (check(root->left, root->data, min) && check(root->right, max, root->data));
    }
    bool isBST(Node* root)
    {
        // Your code here
        return check(root, INT_MAX, INT_MIN);
    }
};


// } Driver Code Ends
```

Custom Input | Compile & Run | Submit

k-th smallest element in BST

Medium | Accuracy: 43.53% | Submissions: 92K+ | Points: 4

Given a BST and an integer K. Find the Kth Smallest element in the BST using O(1) extra space.

Example 1:

Input:

K = 2

Output: 2

Explanation: 2 is the 2nd smallest element in the BST

Example 2:

C++ (g++ 5.4) | Average Time: 40m | Start Timer

```
// } Driver Code Ends

class Solution {
  public:
    // Return the Kth smallest element in the given BST
    int cnt = 0;
    int res;
    void inorder(Node *root, int K){
        if(root == NULL){
            return ;
        }
        inorder(root->left,K);
        if(cnt == K-1){
            res = root->data;
        }
        cnt++;
        inorder(root->right,K);
    }

    int KthSmallestElement(Node *root, int K) {
        // add code here.
        inorder(root,K);
        if(cnt < K){
            return -1;
        }
        return res;
    }
};
// } Driver Code Ends
```

Custom Input | Compile & Run | Submit