Practice(https://practice.geeksforgeeks.org/home/) 181 (/leaderboard)

Problem Editorial Submissions Doubt Support C++ (g++ 5.4) Test against custom input

Lowest Common Ancestor in a BST

Easy Accuracy: 50.22% Submissions: 86135 Points: 2

Given a Binary Search Tree (with all values unique) and two node values. Find the Lowest Common Ancestors of the two nodes in the BST.

Example 1:

```
Input:
              5
           /    \
         4       6
        /         \
       3           7
                    \
                     8
n1 = 7, n2 = 8
Output: 7
```

Example 2:

```
Input:
     2
   /   \
  1     3
n1 = 1, n2 = 3
Output: 2
```

Your Task:
You don't need to read input or print anything. Your task is to complete the function **LCA()** which takes the root Node of the BST and two integer values n1 and n2 as inputs and returns the Lowest Common Ancestor of the Nodes with values n1 and n2 in the given BST.

Expected Time Complexity: O(Height of the BST).
Expected Auxiliary Space: O(Height of the BST).

Constraints:
1 <= N <= 10^4

Output Window

View Bookmarked Problems (https://practice.geeksforgeeks.org/explore/?problemType=bookmark)

Problem Solved Successfully

Company Tags

You get marks only for the first correct submission if you solve the problem without viewing the full solution.

Topic Tags

Related Courses

Related Interview Experiences

Test Cases Passed: **110** / 110

Total Time Taken: **0.2**/1.38

We are replacing the old Disqus forum with the new Discussions section given below. Click here (https://practice.geeksforgeeks.org/comments/lowest-common-ancestor-in-a-bst/1/?rel=https://practice.geeksforgeeks.org/problems/lowest-common-ancestor-in-a-bst/1) to view old Disqus comments.

Correct Submission Count: **6**

Attempts No.: **8**

Report An Issue

If you are facing any issue on this page. Please let us know.

```
/*
    LCA = Stores node pointer
    ans = &LCA;

    ans means address of LCA
    *ans means value of LCA(LCA value is nothing but the address of Node)
    **ans means value pointed by
*/
void solve(Node* root, int n1, int n2, Node** ans){
    if(root == NULL)
        return;

    if((root->data) < n1 and (root->data) < n2)
        solve(root->right, n1, n2, ans);
    else if((root->data) > n1 and (root->data) > n2)
        solve(root->left, n1, n2, ans);
    else *ans = root;
}
Node* LCA(Node *root, int n1, int n2)
{
   //Your code here

```

Average Time: 20m
Your Time: 14m

Compile & Run Submit