**SOLUTION**

/\*\*

\* Definition for a binary tree node.

\* struct TreeNode {

\* int val;

\* TreeNode \*left;

\* TreeNode \*right;

\* TreeNode() : val(0), left(nullptr), right(nullptr) {}

\* TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}

\* TreeNode(int x, TreeNode \*left, TreeNode \*right) : val(x), left(left), right(right) {}

\* };

\*/

class Solution {

public:

Solution(){

ios::sync\_with\_stdio(false);

std::cin.tie(nullptr);

std::cout.tie(nullptr);

}

bool dfs(TreeNode\* root, vector<int>& arr, int index, int& n){

if(root==NULL)

return false;

if(index==n-1 && root->left==NULL && root->right==NULL && root->val==arr[index])

return true;

else if(index<n && root->val==arr[index])

return dfs(root->left,arr,index+1,n) || dfs(root->right,arr,index+1,n);

else

return false;

}

bool isValidSequence(TreeNode\* root, vector<int>& arr) {

int n=arr.size();

int ans=dfs(root,arr,0,n);

return ans;

}

};

**TIME COMPLEXITY: O(N)**

**SPACE COMPLEXITY: O(1)**