PW SKILLS

Assignments Solution

Binary Search Tree 1

PW SKILLS

1. **Find the nodes with minimum and maximum value in a Binary Search Tree.**

Solution :

```
void maxAndmin(TreeNode* root) {
        int mn = root->val, mx = root->val;
        TreeNode* temp = root;
        while(temp->left) {
            temp = temp->left;
            mn = temp->val;
        }
        while(root->right) {
            root = root->right;
            mx = root->val;
        }
        cout << mn << " " << mx << "\n";
}
```

2. **kth Smallest element in a BST** **[Leetcode 230]**

Solution :

```
class Solution {
public:
    // bst ka inorder sorted hota hai, use that property
    int k, ans = -1;
    void helper(TreeNode* root) {
        if(!root) return;
        helper(root->left);
        k--;
        if(k == 0) ans = root->val;
        helper(root->right);
    }
    int kthSmallest(TreeNode* root, int k) {
        this->k = k;
        helper(root);
        return ans;
    }
};
```

3. **Given the root of a binary search tree, return a balanced BST with the same node values.** **[Leetcode 1382]**

Solution :

```
class Solution {
public:
    void inorder(TreeNode* root,vector<TreeNode*> &v){
        if(root==NULL) return;
        if(root->left) inorder(root->left,v);
        v.push_back(root);
        if(root->right) inorder(root->right,v);
    }
    TreeNode* solve(int low,int high,vector<TreeNode*> &v){
        if(low>high)
        return NULL;
        int m=(low+high)/2;
        v[m]->left=solve(low,m-1,v);
        v[m]->right=solve(m+1,high,v);
        return v[m];
    }
    TreeNode* balanceBST(TreeNode* root) {
     vector<TreeNode*> v;
     inorder(root,v);
     return solve(0,v.size()-1,v);
    }
};
```

4. **Given the root node of a binary search tree and two integers low and high, return the sum of values of all nodes with a value in the inclusive range [low, high].**

[Leetcode 938]

Solution :

```
class Solution {
public:
    // just use property that inorder is sorted
    int ans = 0;
    void helper(TreeNode* root, int l, int r) {
        if(!root) return;
        helper(root->left, l, r);
        if(l <= root->val && root->val <= r) ans += root->val;
        helper(root->right, l, r);
    }
    int rangeSumBST(TreeNode* root, int low, int high) {
        helper(root, low, high);
        return ans;
    }
};
```

Note:- Please try to invest time doing the assignments which are necessary to build a strong foundation. Do not directly Copy Paste using Google or ChatGPT. Please use your brain.