Binary Search Tree & Heap

1. https://leetcode.com/problems/range-sum-of-bst/

```
class Solution {
public:
    int rangeSumBST(TreeNode* root, int low, int
high) {
        if(root == NULL) return 0;

        if(root->val < low){
            return rangeSumBST(root->right, low,
high);
        }
        if(root->val > high) return
rangeSumBST(root->left, low, high);
        // jodi low < root->val < high tokhn sum
korbo
        return root->val + rangeSumBST(root->left,
low, high) + rangeSumBST(root->right, low, high);
    }
};
```

2. https://leetcode.com/problems/search-in-a-binary-search-tree/
3. https://leetcode.com/problems/two-sum-iv-input-is-a-bst/description/

```
class Solution {
public:
    void inorder(TreeNode* root, vector<int> &v){
        if(root==NULL) return;
        inorder(root->left, v);
        v.push_back(root->val);
        inorder(root->right, v);
    }
```

```
    bool findTarget(TreeNode* root, int k) {
        vector<int> list;
        inorder(root, list);
        // target = 6
        // 2 3 4 5 6 7
        // l   r
        int l=0, r = list.size()-1;
        while(l<r){
            int sum = list[l] + list[r];
            if(sum == k) return true;
            if(sum < k){
                l++;
            }
            if(sum > k) r--;
        }
        return false;
    }
};
```

4. https://leetcode.com/problems/minimum-absolute-difference-in-bst/description/

```
class Solution {
public:
    int result = INT_MAX;
    int history = INT_MAX;

    void inorder(TreeNode *root){
        if(root == NULL) return;
        inorder(root->left);
        // root->val = 2
        result = min(result, abs(root->val -
history));  // INT_MAX
```

```
        // history = 2
        history = root->val; // immediately je value
niye kaj korlam seta store kortechi
        inorder(root->right);
    }
    int getMinimumDifference(TreeNode* root) {
           inorder(root);
           return result;
    }

};
```