Binary Search Trees
(Assignment Solutions)

Question 1 :

```
int rangeSumBST(TreeNode* root, int low, int high) {
        if(root->val >=low && root->val <= high){
            return root->val
            + rangeSumBST(root->left,low,high)
            + rangeSumBST(root->right,low,high);
        }
        if(root->val < low) {
            return rangeSumBST(root->right,low,high);
        }

        return rangeSumBST(root->left,low,high) ;
}
```

Question 2 :

```
void maxDiffUtil(struct Node *ptr, int k, int &min_diff,
                                        int &min_diff_key) {
    if (ptr == NULL)
        return ;

    if (ptr->key == k)
    {
        min_diff_key = 0;
        return;
    }

    if (min_diff > abs(ptr->key - k))
    {
        min_diff = abs(ptr->key - k);
        min_diff_key = ptr->key;
    }

    if (k < ptr->key)
        maxDiffUtil(ptr->left, k, min_diff, min_diff_key);
```

```
    else
        maxDiffUtil(ptr->right, k, min_diff, min_diff_key);
}

int maxDiff(Node *root, int k) {
    int min_diff = INT_MAX, min_diff_key = -1;
    maxDiffUtil(root, k, min_diff, min_diff_key);
    return min_diff_key;
}
```

Question 3 :

```
int search(TreeNode* root, int& K) {
        if (root == NULL) {
            return -1;
        }

        int leftVal = search(root->left, K);
        if(leftVal != -1) {
            return leftVal;
        }

        K--;
        if(K == 0) {
            return root->val;
        }

        return search(root->right, K);
    }

    int kthSmallest(TreeNode* root, int k) {
        return search(root, k);
    }
```

Question 4 :

```
bool isPairPresent(Node* root, Node* temp, int target)
    {
```

```
        if (temp == NULL)
            return false;
        return search(root, temp, target - temp->data)
               || isPairPresent(root, temp->left, target)
               || isPairPresent(root, temp->right, target);
    }
    bool search(Node* root, Node* temp, int k)
    {
        if (root == NULL)
            return false;
        Node* c = root;
        bool flag = false;
        while (c != NULL && flag != true) {
            if (c->data == k && temp != c) {
                flag = true;
                cout << "Pair Found: " << c->data << " + "
                     << temp->data;
                return true;
            }
            else if (k < c->data)
                c = c->left;
            else
                c = c->right;
        }
        return false;
    }
```

Question 5 :

```
class nodeValue{
    public:
    int min;
    int max;
    int sum;
    nodeValue(int min,int max,int sum)
    {
        this->min=min;
```

```
        this->max=max;
        this->sum=sum;
    }
};

class Solution {
public:
   int ans=0;
   int maxSumBST(TreeNode* root) {
       helper(root);
       return ans;
   }
   nodeValue helper(TreeNode* root) {
       if(root==NULL)
       return nodeValue(INT_MAX,INT_MIN,0);
       nodeValue l=helper(root->left);
       nodeValue r=helper(root->right);
       if(root->val>l.max && root->val<r.min)
       {
           ans=max(ans,root->val+l.sum+r.sum);
                                                                    return
nodeValue(min(root->val,l.min),max(root->val,r.max),l.sum+root->val+r.sum);
       }
       return nodeValue(INT_MIN,INT_MAX,max(l.sum,r.sum));
   }
};
```

https://telegram.me/+QGUyTripaP4zNTcx

https://telegram.me/+nEKeBr_yhXtmY2Yx

Say Hi to TG for Further Updates:
https://t.me/RepublicDayBot

#TGSFamily