https://course.acciojob.com/idle?question=a429b9c2-b596-426b-a197-72aeb9a5bb23

- MEDIUM
- Max Score: 30 Points

Sum of range in BST

You are given a pointer to the root of binary search tree. You are also given two integers `L` and `R`. You have to find the sum of values of nodes which lie in the range from [L, R] inclusive.

NOTE: You need to complete the given function `rangeSum` which receives the root, `L` and `R` as the parameters and returns the required sum. The input and printing of output will be handled by the driver code.

Input Format:

The first line contains the number of test cases.

For each test case: You are given a pointer to the root of the binary tree and the values of 'L' and 'R'.

Output Format:

For each test case print the sum.

Example 1:

Input:

```
1
2 4

   2
 /   \
1     3
```

Output:

```
5
```

Explanation:

'2' and '3' lie within the range [2, 4]. 2 + 3 = 5.

Example 2:

Input:

```
1
2 4

   2
 /   \
1     4
     /
    3
```

Output:

```
9
```

Explanation:

'2', '3', and '4' lie within the range [2, 4]. 2 + 3 + 4 = 9.

Constraints:

1 <= T <= 10

1 <= N <= 10000

1 <= L <= R <= 1000000

1 <= A[i] <= 1000000

Topic Tags

- BST

My code

```
// n java
import java.util.*;
class Node {
    int data;
    Node left, right;
    public Node(int item)
    {
        data = item;
        left = right = null;
    }
}

class BinarySearchTree
{
    Node constructBST(int[]arr,int start,int end,Node root)
    {
        if(start>end)
            return null;
        int mid=(start+end)/2;

        if(root==null)
            root=new Node(arr[mid]);

        root.left=constructBST(arr,start,mid-1, root.left);
        root.right=constructBST(arr,mid+1,end, root.right);
```

```
        return root;

    }
}

public class Main {
    public static void main(String[] args) throws Throwable {
        Scanner sc = new Scanner(System.in);
        int t = sc.nextInt();
        while(t-->0){
            int n = sc.nextInt();
            int l = sc.nextInt();
            int r = sc.nextInt();
            int arr[]=new int[n];
            for (int i = 0; i < n; i++)
            {
                arr[i] = sc.nextInt();
            }

            Arrays.sort(arr);
            Node root=null;
            BinarySearchTree bst=new BinarySearchTree();
            root=bst.constructBST(arr,0,n-1,root);

            Solution A = new Solution();
            long ans=A.rangeSum(root,l,r);
            System.out.println(ans);
        }
    }
```

```
}

class Solution
{
    long rangeSum(Node root, int l, int r){
        if(root==null)
            return 0;
        long a=0L;
        if(root.data<=r && root.data>=l)
            a=root.data;
        long b=rangeSum( root.left,  l, r);
        long c=rangeSum( root.right,  l, r);
        return a+b+c;

    }
}
```