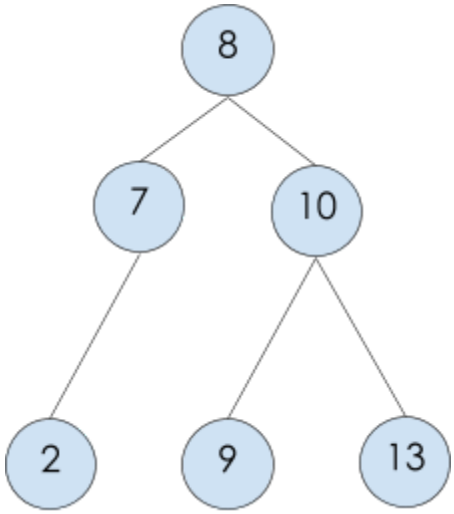
Sum of the K-smallest elements inside a Binary Search Tree

You are asked to find the sum of the k-smallest elements inside a Binary Search Tree (BST) *without* any change in the BST node structure. Further discuss the computational complexity (Big-O) of the solution.

Reference: https://www.geeksforgeeks.org

Outline of the problem: Consider the following example:



with K = 3

The K-th smallest element is "8" and thus the sum of all elements of this tree that are smaller or equal to that is 2 + 7 + 8.

Coding Task: Considering the code subset provided below, implement the `ksmallestElementSum(struct Node *root, int k)` method.

Provided Code:

```
// c++ program to find Sum Of All Elements smaller
// than or equal to Kth Smallest Element In BST
#include <bits/stdc++.h>
using namespace std;

/* Binary tree Node */
struct Node
{
    int data;
```

```
    Node* left, * right;
};

// utility function new Node of BST
struct Node *createNode(int data)
{
    Node * new_Node = new Node;
    new_Node->left = NULL;
    new_Node->right = NULL;
    new_Node->data = data;
    return new_Node;
}

// A utility function to insert a new Node
//  with given key in BST and also maintain lcount ,Sum
struct Node * insert(Node *root, int key)
{
    // If the tree is empty, return a new Node
    if (root == NULL)
        return createNode(key);

    // Otherwise, recur down the tree
    if (root->data > key)
        root->left = insert(root->left, key);

    else if (root->data < key)
        root->right = insert(root->right, key);

    // return the (unchanged) Node pointer
    return root;
}


/*
```

HERE GOES YOUR IMPLEMENTATION

```
*/

/* Driver program to test above functions */
int main()
{

    Node *root = NULL;
    root = insert(root, 20);
```

```
    root = insert(root, 8);
    root = insert(root, 4);
    root = insert(root, 12);
    root = insert(root, 10);
    root = insert(root, 14);
    root = insert(root, 22);

    int k = 3;
    cout <<  ksmallestElementSum(root, k) << endl;
    return 0;
}
```