Code | Accepted ×

All Submissions

```c
int kthSmallest(struct TreeNode* root, int k) {
    // Stack for iterative traversal
    struct TreeNode* stack[100];
    int top = -1;

    // Current node
    struct TreeNode* curr = root;

    // Variable to keep track of visited nodes
    int count = 0;

    // Traverse the tree until the current node is NULL
    while (curr != NULL || top != -1) {
        // Move to the leftmost node
        while (curr != NULL) {
            stack[++top] = curr;
            curr = curr->left;
        }

        // Pop the top node from the stack
        curr = stack[top--];

        // Increment the count
        count++;

        // If count equals k, return the value of the current node
        if (count == k) {
            return curr->val;
        }

        // Move to the right of the current node
        curr = curr->right;
    }

    // If k is greater than the number of nodes in the tree
    return -1; // or any other appropriate error code
}
```

Code | Accepted ×

All Submissions

Accepted

nid_hii submitted at May 09, 2024 09:55

Editorial | Solution

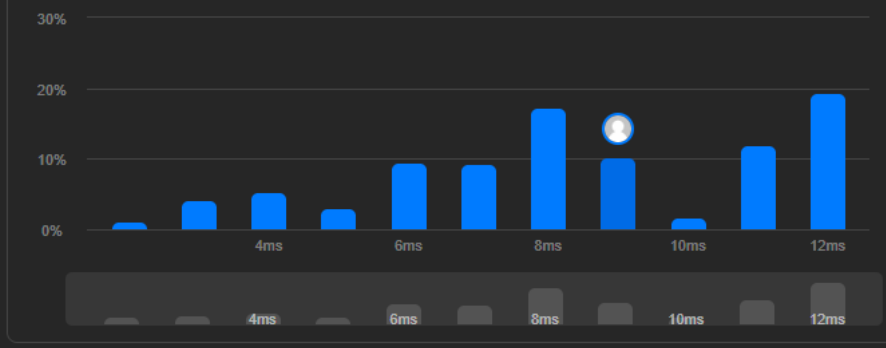
Runtime

9 ms | Beats 50.60%

Analyze Complexity

Memory

10.36 MB | Beats 24.25%



Code | C

Testcase | Test Result

• Case 1 • Case 2

Input

root =

[3,1,4,null,2]

k =

1

Output

1

Expected

1

Contribute a testcase

Testcase | Test Result

• Case 1 • Case 2

Input

root =

[5,3,6,2,4,null,null,1]

k =

3

Output

3

Expected

3

Contribute a testcase