Prompt | Scratchpad | Solutions | Video Explanation

Difficulty: Easy **Category:** Binary Search Trees **Successful Submissions:** 72,477+

Find Closest Value In BST

Write a function that takes in a Binary Search Tree (BST) and a target integer value and returns the closest value to that target value contained in the BST.

You can assume that there will only be one closest value.

Each `BST` node has an integer `value`, a `left` child node, and a `right` child node. A node is said to be a valid `BST` node if and only if it satisfies the BST property: its `value` is strictly greater than the values of every node to its left; its `value` is less than or equal to the values of every node to its right; and its children nodes are either valid `BST` nodes themselves or `None` / `null`.

Sample Input

```
tree =   10
       /     \
      5       15
    /   \   /   \
   2     5 13   22
 /           \
1             14
target = 12
```

Sample Output

```
13
```

Hints

Hint 1

Try traversing the BST node by node, all the while keeping track of the node with the value closest to the target value. Calculating the absolute value of the difference between a node's value and the target value should allow you to check if that node is closer than the current closest one.

Hint 2

Make use of the BST property to determine what side of any given node has values close to the target value and is therefore worth exploring.

Hint 3

What are the advantages and disadvantages of solving this problem iteratively as opposed to recursively?

Optimal Space & Time Complexity

Average: O(log(n)) time | O(1) space - where n is the number of nodes in the BST Worst: O(n) time | O(1) space - where n is the number of nodes in the BST

Your Solutions | Run Code

Solution 1 | Solution 2 | Solution 3

```python
class BST:
    def __init__(self, value):
        self.value = value
        self.left = None
        self.right = None

def find_closest(tree, target, closest):
    if tree is None:
        return closest

    if abs(target - closest) > abs(target - tree.value):
        closest = tree.value

    if target > tree.value:
        return find_closest(tree.right, target, closest)
    elif target < tree.value:
        return find_closest(tree.left, target, closest)
    else:
        return target

def findClosestValueInBst(tree, target):
    inf = float("inf")
    return find_closest(tree, target, inf)

# Kunal Wadhwa

# https://github.com/kunal5042
# https://leetcode.com/kunal5042/
# https://www.hackerrank.com/kunalwadhwa_cs
# https://www.linkedin.com/in/kunal5042/

```

Custom Output | Raw Output | Sub

Yay, your code passed all the test cases!

12 / 12 test cases passed.

- Test Case 1 passed!
- Test Case 2 passed!
- Test Case 3 passed!
- Test Case 4 passed!
- Test Case 5 passed!
- Test Case 6 passed!
- Test Case 7 passed!
- Test Case 8 passed!
- Test Case 9 passed!
- Test Case 10 passed!
- Test Case 11 passed!
- Test Case 12 passed!