

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 using System;
4
5 public class Program {
6     // Average: O(log(n)) time | O(log(n)) space
7     // Worst: O(n) time | O(n) space
8     public static int FindClosestValueInBst(BST tree, int target) {
9         return FindClosestValueInBst(tree, target, Int32.MaxValue);
10    }
11
12    public static int FindClosestValueInBst(BST tree, int target, double closest) {
13        if (Math.Abs(target - closest) > Math.Abs(target - tree.value)) {
14            closest = tree.value;
15        }
16        if (target < tree.value && tree.left != null) {
17            return FindClosestValueInBst(tree.left, target, closest);
18        } else if (target > tree.value && tree.right != null) {
19            return FindClosestValueInBst(tree.right, target, closest);
20        } else {
21            return (int)closest;
22        }
23    }
24
25    public class BST {
26        public int value;
27        public BST left;
28        public BST right;
29
30        public BST(int value) {
31            this.value = value;
32        }
33    }
```

Solution 1

Solution 2

Solution 3

```
1 public class Program {
2     public static int FindClosestValueInBst(BST tree, int target) {
3         // Write your code here.
4         return -1;
5     }
6
7     public class BST {
8         public int value;
9         public BST left;
10        public BST right;
11
12        public BST(int value) {
13            this.value = value;
14        }
15    }
16 }
17
```

Our Tests

Custom Output

Submit Code

```
1 public class Program {
2     public static int FindClosestValueInBst(BST tree, int target) {
3         // Write your code here.
4         return -1;
5     }
6
7     public class BST {
8         public int value;
9         public BST left;
10        public BST right;
11
12        public BST(int value) {
13            this.value = value;
14        }
15    }
```

```
10         return [0] * (len(arr) - 1)
11     }
12     [0] * (len(arr) - 1)
13     [0] * (len(arr) - 1)
14     [0] * (len(arr) - 1)
15     [0] * (len(arr) - 1)
16     [0] * (len(arr) - 1)
17     [0] * (len(arr) - 1)
18     [0] * (len(arr) - 1)
19     [0] * (len(arr) - 1)
20     [0] * (len(arr) - 1)
21     [0] * (len(arr) - 1)
22     [0] * (len(arr) - 1)
23     [0] * (len(arr) - 1)
24     [0] * (len(arr) - 1)
25     [0] * (len(arr) - 1)
26     [0] * (len(arr) - 1)
27     [0] * (len(arr) - 1)
28     [0] * (len(arr) - 1)
29     [0] * (len(arr) - 1)
30     [0] * (len(arr) - 1)
31     [0] * (len(arr) - 1)
32     [0] * (len(arr) - 1)
33     [0] * (len(arr) - 1)
34     [0] * (len(arr) - 1)
35     [0] * (len(arr) - 1)
36     [0] * (len(arr) - 1)
37     [0] * (len(arr) - 1)
38     [0] * (len(arr) - 1)
39     [0] * (len(arr) - 1)
40     [0] * (len(arr) - 1)
41     [0] * (len(arr) - 1)
42     [0] * (len(arr) - 1)
43     [0] * (len(arr) - 1)
44     [0] * (len(arr) - 1)
45     [0] * (len(arr) - 1)
46     [0] * (len(arr) - 1)
47     [0] * (len(arr) - 1)
48     [0] * (len(arr) - 1)
49     [0] * (len(arr) - 1)
50     [0] * (len(arr) - 1)
51     [0] * (len(arr) - 1)
52     [0] * (len(arr) - 1)
53     [0] * (len(arr) - 1)
54     [0] * (len(arr) - 1)
55     [0] * (len(arr) - 1)
56     [0] * (len(arr) - 1)
57     [0] * (len(arr) - 1)
58     [0] * (len(arr) - 1)
59     [0] * (len(arr) - 1)
60     [0] * (len(arr) - 1)
61     [0] * (len(arr) - 1)
62     [0] * (len(arr) - 1)
63     [0] * (len(arr) - 1)
64     [0] * (len(arr) - 1)
65     [0] * (len(arr) - 1)
66     [0] * (len(arr) - 1)
67     [0] * (len(arr) - 1)
68     [0] * (len(arr) - 1)
69     [0] * (len(arr) - 1)
70     [0] * (len(arr) - 1)
71     [0] * (len(arr) - 1)
72     [0] * (len(arr) - 1)
73     [0] * (len(arr) - 1)
74     [0] * (len(arr) - 1)
75     [0] * (len(arr) - 1)
76     [0] * (len(arr) - 1)
77     [0] * (len(arr) - 1)
78     [0] * (len(arr) - 1)
79     [0] * (len(arr) - 1)
80     [0] * (len(arr) - 1)
81     [0] * (len(arr) - 1)
82     [0] * (len(arr) - 1)
83     [0] * (len(arr) - 1)
84     [0] * (len(arr) - 1)
85     [0] * (len(arr) - 1)
86     [0] * (len(arr) - 1)
87     [0] * (len(arr) - 1)
88     [0] * (len(arr) - 1)
89     [0] * (len(arr) - 1)
90     [0] * (len(arr) - 1)
91     [0] * (len(arr) - 1)
92     [0] * (len(arr) - 1)
93     [0] * (len(arr) - 1)
94     [0] * (len(arr) - 1)
95     [0] * (len(arr) - 1)
96     [0] * (len(arr) - 1)
97     [0] * (len(arr) - 1)
98     [0] * (len(arr) - 1)
99     [0] * (len(arr) - 1)
100    [0] * (len(arr) - 1)
```

Run or submit code when you're ready.