

Our Solution(s)

Run Code

Your Solutions

Run Code

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

| Solution 1 | Solution 2 | Solution 3 |
|--|------------|------------|
| <pre>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</pre> | | |

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 }

16 }

17

1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.

2

3 using System;

4

5 public class Program {

6 // Average: O(log(n)) time | O(1) space

7 // Worst: O(n) time | O(1) 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 BST currentNode = tree;

14 while (currentNode != null) {

15 if (Math.Abs(target - closest) > Math.Abs(target - currentNode.value))

16 closest = currentNode.value;

17 }

18 if (target < currentNode.value) {

19 currentNode = currentNode.left;

20 } else if (target > currentNode.value) {

21 currentNode = currentNode.right;

22 } else {

23 break;

24 }

25 }

26 return (int)closest;

27 }

28

29 public class BST {

30 public int value;

31 public BST left;

32 public BST right;

33 }

```
10         return [0] * (len(arr) - 1)
11     }
12 }
13
14 // Test 1
15 [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10].map((val, index) => {
16     return index * val;
17 });
18
19 // Test 2
20 [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10].map((val, index) => {
21     return index * val;
22 });
23
24 // Test 3
25 [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10].map((val, index) => {
26     return index * val;
27 });
28
29 // Test 4
30 [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10].map((val, index) => {
31     return index * val;
32 });
```

Run or submit code when you're ready.