

PromptScratchpadOur Solution(s)Video Explanation

Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 #include <vector>
4
5 using namespace std;
6
7 bool areSameBsts(vector<int> arrayOne, vector<int> arrayTwo, int rootIdxOne,
8                 int rootIdxTwo, int minVal, int maxVal);
9 int getIdxOffFirstSmaller(vector<int> array, int startingIdx, int minVal);
10 int getIdxOffFirstBiggerOrEqual(vector<int> array, int startingIdx, int maxVal);
11
12 // O(n^2) time | O(d) space - where n is the number of
13 // nodes in each array, respectively, and d is the depth
14 // of the BST that they represent
15 bool sameBsts(vector<int> arrayOne, vector<int> arrayTwo) {
16     return areSameBsts(arrayOne, arrayTwo, 0, 0, INT_MIN, INT_MAX);
17 }
18
19 bool areSameBsts(vector<int> arrayOne, vector<int> arrayTwo, int rootIdxOne,
20                 int rootIdxTwo, int minVal, int maxVal) {
21     if (rootIdxOne == -1 || rootIdxTwo == -1)
22         return rootIdxOne == rootIdxTwo;
23
24     if (arrayOne[rootIdxOne] != arrayTwo[rootIdxTwo])
25         return false;
26
27     int leftRootIdxOne = getIdxOffFirstSmaller(arrayOne, rootIdxOne, minVal);
28     int leftRootIdxTwo = getIdxOffFirstSmaller(arrayTwo, rootIdxTwo, minVal);
29     int rightRootIdxOne =
30         getIdxOffFirstBiggerOrEqual(arrayOne, rootIdxOne, maxVal);
31     int rightRootIdxTwo =
32         getIdxOffFirstBiggerOrEqual(arrayTwo, rootIdxTwo, maxVal);
33
34     int currentValue = arrayOne[rootIdxOne];
35     bool leftAreSame = areSameBsts(arrayOne, arrayTwo, leftRootIdxOne,
36                                   leftRootIdxTwo, minVal, currentValue);
37     bool rightAreSame = areSameBsts(arrayOne, arrayTwo, rightRootIdxOne,
38                                    rightRootIdxTwo, currentValue, maxVal);
39
40     return leftAreSame && rightAreSame;
41 }
42
43 int getIdxOffFirstSmaller(vector<int> array, int startingIdx, int minVal) {
44     // Find the index of the first smaller value after the startingIdx.
45     // Make sure that this value is greater than or equal to the minVal,
46     // which is the value of the previous parent node in the BST. If it
47     // isn't, then that value is located in the left subtree of the
48     // previous parent node.
49     for (int i = startingIdx + 1; i < array.size(); i++) {
50         if (array[i] < array[startingIdx] && array[i] >= minVal)
51             return i;
52     }
53     return -1;
54 }
55
56 int getIdxOffFirstBiggerOrEqual(vector<int> array, int startingIdx, int maxVal) {
57     // Find the index of the first bigger/equal value after the startingIdx.
58     // Make sure that this value is smaller than maxVal, which is the value
59     // of the previous parent node in the BST. If it isn't, then that value
60     // is located in the right subtree of the previous parent node.
61     for (int i = startingIdx + 1; i < array.size(); i++) {
62         if (array[i] >= array[startingIdx] && array[i] < maxVal)
63             return i;
64     }
65     return -1;
66 }
```