Our Solution(s) Run Code

Solution 1 Solution 2

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

// O(n^2) time | O(d) space - where n is the number of
// nodes in each array, respectively, and d is the depth
// of the BST that they represent
function sameBsts(arrayOne, arrayTwo) {
  return areSameBsts(arrayOne, arrayTwo, 0, 0, -Infinity, Infinity);
}

function areSameBsts(arrayOne, arrayTwo, rootIdxOne, rootIdxTwo, minV
  if (rootIdxOne === -1 || rootIdxTwo === -1) return rootIdxOne === r

  if (arrayOne[rootIdxOne] !== arrayTwo[rootIdxTwo]) return false;

  const leftRootIdxOne = getIdxOfFirstSmaller(arrayOne, rootIdxOne, mi
  const leftRootIdxTwo = getIdxOfFirstSmaller(arrayTwo, rootIdxTwo, mi
  const rightRootIdxOne = getIdxOfFirstBiggerOrEqual(arrayOne, rootIdx
  const rightRootIdxTwo = getIdxOfFirstBiggerOrEqual(arrayTwo, rootIdx

  const currentValue = arrayOne[rootIdxOne];
  const leftAreSame = areSameBsts(arrayOne, arrayTwo, leftRootIdxOne,
  const rightAreSame = areSameBsts(arrayOne, arrayTwo, rightRootIdxOne

  return leftAreSame && rightAreSame;
}

function getIdxOfFirstSmaller(array, startingIdx, minVal) {
  // Find the index of the first smaller value after the startingIdx.
  // Make sure that this value is greater than or equal to the minVal,
  // which is the value of the previous parent node in the BST. If it
  // isn't, then that value is located in the left subtree of the
  // previous parent node.
  for (let i = startingIdx + 1; i < array.length; i++) {
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```
function sameBsts(arrayOne, arrayTwo) {
  // Write your code here.
}

// Do not edit the line below.
exports.sameBsts = sameBsts;

```

Our Tests

Custom Output Submit Code

```
});

it("Test Case #6", function () {
  const array1 = [7, 8, 5, 4, 3, 2, 1];
  const array2 = [7, 8, 5, 4, 3, 2, 1];
  assert.equal(program.sameMirror(array1, array2), true
});

it("Test Case #7", function () {
  const array1 = [10, 10, 8, 12, 90, 90, 3, 2];
  const array2 = [10, 8, 3, 90, 3, 12, 90, 90];
  assert.equal(program.sameMirror(array1, array2), true
});

it("Test Case #8", function () {
```

Run or submit code when you're ready.