Our Solution(s) Run Code

Solution 1 Solution 2

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

public class Program {
  // O(log(n)) time | O(1) space
  public static int[] SearchForRange(int[] array, int target) {
    int[] finalRange = {-1, -1};
    alteredBinarySearch(array, target, 0, array.Length - 1, finalRang
    alteredBinarySearch(array, target, 0, array.Length - 1, finalRang
    return finalRange;
  }

  public static void alteredBinarySearch(int[] array, int target, int
    int[] finalRange, bool goLeft) {
    while (left <= right) {
      int mid = (left + right) / 2;
      if (array[mid] < target) {
        left = mid + 1;
      } else if (array[mid] > target) {
        right = mid - 1;
      } else {
        if (goLeft) {
          if (mid == 0 || array[mid - 1] != target) {
            finalRange[0] = mid;
            return;
          } else {
            right = mid - 1;
          }
        } else {
          if (mid == array.Length - 1 || array[mid + 1] != target) {
            finalRange[1] = mid;
            return;
          } else {
            left = mid + 1;
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```
public class Program {
  public static int[] SearchForRange(int[] array, int target) {
    // Write your code here.
    return null;
  }
}

```

Our Tests

Custom Output Submit Code

```
}

[Test]
public void TestCase2() {
    int[] expected = {5, 6};
    int[] output = Program.SearchForRange(new int[] {5, 7, 7, 8, 8, 10
    Utils.AssertTrue(isEqual(output, expected));
}

[Test]
public void TestCase3() {
    int[] expected = {3, 4};
    int[] output = Program.SearchForRange(new int[] {5, 7, 7, 8, 8, 10
    Utils.AssertTrue(isEqual(output, expected));
}
```

Run or submit code when you're ready.