Solution 1

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

using System;
public class Program {
  // O(n) time | O(1) space
  public static int[] SubarraySort(int[] array) {
    int minOutOfOrder = Int32.MaxValue;
    int maxOutOfOrder = Int32.MinValue;
    for (int i = 0; i < array.Length; i++) {
      int num = array[i];
      if (isOutOfOrder(i, num, array)) {
        minOutOfOrder = Math.Min(minOutOfOrder, num);
        maxOutOfOrder = Math.Max(maxOutOfOrder, num);
      }
    }
    if (minOutOfOrder == Int32.MaxValue) {
      return new int[] {-1, -1};
    }
    int subarrayLeftIdx = 0;
    while (minOutOfOrder >= array[subarrayLeftIdx]) {
      subarrayLeftIdx++;
    }
    int subarrayRightIdx = array.Length - 1;
    while (maxOutOfOrder <= array[subarrayRightIdx]) {
      subarrayRightIdx--;
    }
    return new int[] {subarrayLeftIdx, subarrayRightIdx};
  }

  public static bool isOutOfOrder(int i, int num, int[] array) {
    if (i == 0) {
      return num > array[i + 1];
    }
    if (i == array.Length - 1) {
      return num < array[i - 1];
    }
    return num > array[i + 1] || num < array[i - 1];
  }
}

```