Our Solution(s) Run Code

Solution 1

```python
# Copyright © 2020 AlgoExpert, LLC. All rights reserved.

# O(n) time | O(1) space
def subarraySort(array):
    minOutOfOrder = float("inf")
    maxOutOfOrder = float("-inf")
    for i in range(len(array)):
        num = array[i]
        if isOutOfOrder(i, num, array):
            minOutOfOrder = min(minOutOfOrder, num)
            maxOutOfOrder = max(maxOutOfOrder, num)
    if minOutOfOrder == float("inf"):
        return [-1, -1]
    subarrayLeftIdx = 0
    while minOutOfOrder >= array[subarrayLeftIdx]:
        subarrayLeftIdx += 1
    subarrayRightIdx = len(array) - 1
    while maxOutOfOrder <= array[subarrayRightIdx]:
        subarrayRightIdx -= 1
    return [subarrayLeftIdx, subarrayRightIdx]


def isOutOfOrder(i, num, array):
    if i == 0:
        return num > array[i + 1]
    if i == len(array) - 1:
        return num < array[i - 1]
    return num > array[i + 1] or num < array[i - 1]

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```python
def subarraySort(array):
    # Write your code here.
    pass

```

Our Tests

Custom Output Submit Code

```
    def test_case_2(self):
        self.assertEqual(longestSubarray([1, 1], 0), 2)

    def test_case_3(self):
        self.assertEqual(longestSubarray([1, 1, 0, 1, 0, 1], 1),

    def test_case_4(self):
        self.assertEqual(longestSubarray([1, 1, 0, 1, 0, 1],

    def test_case_5(self):
        self.assertEqual(longestSubarray([1, 1, 0, 1, 0, 1],

    def test_case_6(self):
        self.assertEqual(longestSubarray([1, 1, 0, 0, 0, 1], 1), 4)
```

Run or submit code when you're ready.