

Our Solution(s)Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     // O(n) time | O(1) space - where n is the length of the array
5     func isMonotonic(array: [Int]) -> Bool {
6         if array.count <= 2 {
7             return true
8         }
9
10        var direction = array[1] - array[0]
11        for i in 2 ..< array.count {
12            if direction == 0 {
13                direction = array[i] - array[i - 1]
14                continue
15            }
16
17            if breaksDirection(direction: direction, previousInt: direction, currentInt: array[i] - array[i - 1]) {
18                return false
19            }
20        }
21
22        return true
23    }
24
25    func breaksDirection(direction: Int, previousInt: Int, currentInt: Int) -> Bool {
26        let difference = currentInt - previousInt
27        if direction > 0 {
28            return difference < 0
29        }
30        return difference > 0
31    }
32 }
33
```

Your SolutionsRun Code

Solution 1Solution 2Solution 3

```
1 class Program {
2     func isMonotonic(array: [Int]) -> Bool {
3         // Write your code here.
4         return false
5     }
6 }
7
```

```

1 def test1():
2     program = Program()
3     testenv("Test case 1") & (0) 00000000000000000000
4     array = [0]
5     expected = 0
6     actual = program.callTestFunction(array, array)
7     assertEqual(expected, actual)
8
9
10 testenv("Test case 2") & (0) 00000000000000000000
11 array = [1]
12 expected = 1
13 actual = program.callTestFunction(array, array)
14 assertEqual(expected, actual)
15
16
17 testenv("Test case 3") & (0) 00000000000000000000
18 array = [5, 10]
19 expected = 15
20 actual = program.callTestFunction(array, array)
21 assertEqual(expected, actual)
22
23
24 testenv("Test case 4") & (0) 00000000000000000000

```

Run or submit code when you're ready.