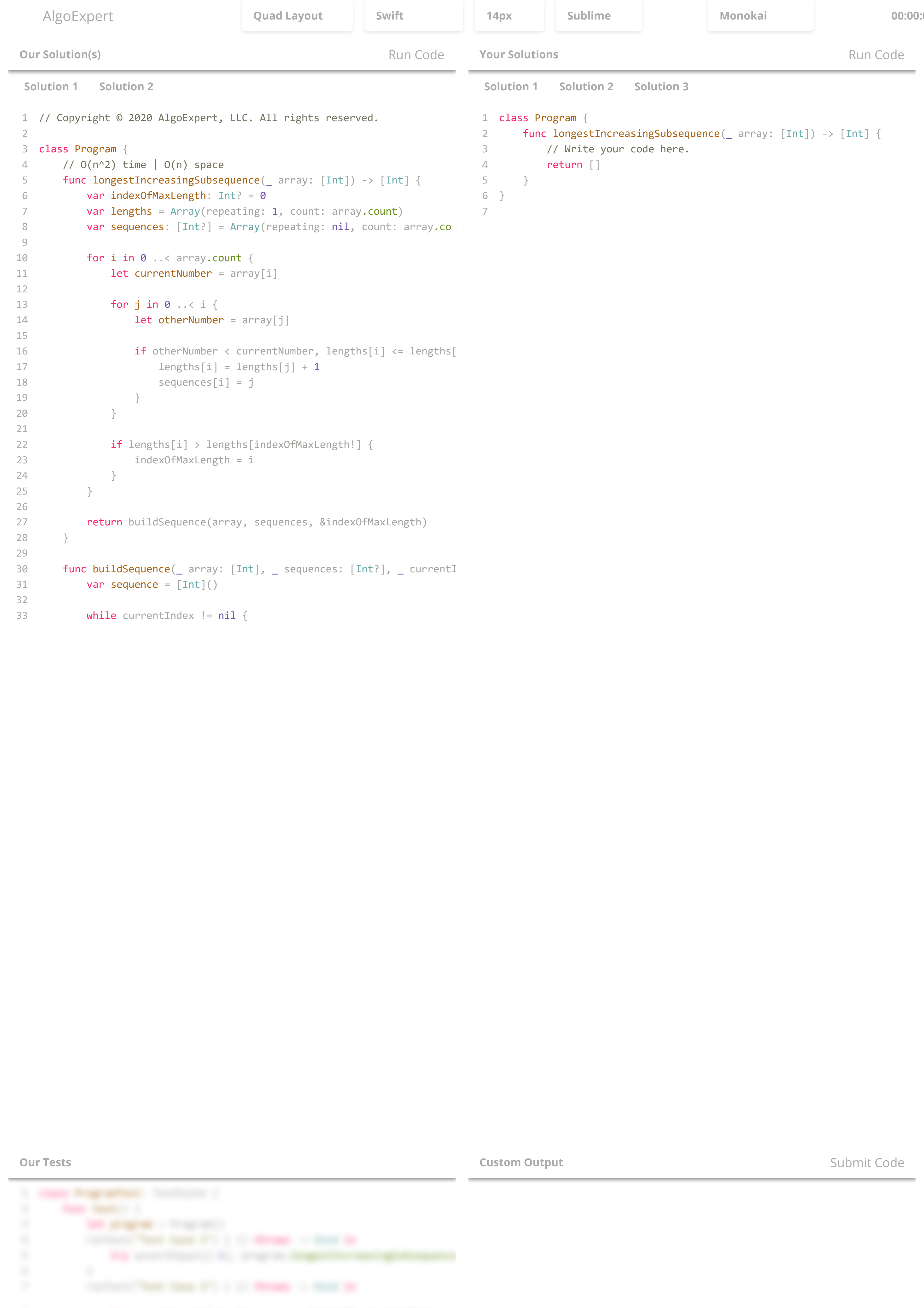

Our Solution(s) — Run Code

Solution 1 · Solution 2

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

class Program {
    // O(n^2) time | O(n) space
    func longestIncreasingSubsequence(_ array: [Int]) -> [Int] {
        var indexOfMaxLength: Int? = 0
        var lengths = Array(repeating: 1, count: array.count)
        var sequences: [Int?] = Array(repeating: nil, count: array.co

        for i in 0 ..< array.count {
            let currentNumber = array[i]

            for j in 0 ..< i {
                let otherNumber = array[j]

                if otherNumber < currentNumber, lengths[i] <= lengths[
                    lengths[i] = lengths[j] + 1
                    sequences[i] = j
                }
            }

            if lengths[i] > lengths[indexOfMaxLength!] {
                indexOfMaxLength = i
            }
        }

        return buildSequence(array, sequences, &indexOfMaxLength)
    }

    func buildSequence(_ array: [Int], _ sequences: [Int?], _ currentI
        var sequence = [Int]()

        while currentIndex != nil {
```

Your Solutions — Run Code

Solution 1 · Solution 2 · Solution 3

```swift
class Program {
    func longestIncreasingSubsequence(_ array: [Int]) -> [Int] {
        // Write your code here.
        return []
    }
}
```

Our Tests

[illegible]

Custom Output — Submit Code

Run or submit code when you're ready.