Our Solution(s) Run Code

Solution 1 Solution 2 Solution 3 Solution 4

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

class Program {
    // O(nm * min(n, m)) time | O(min(n, m)^2) space
    func longestCommonSubsequence(firstString: String, secondString:
        let smallestString = firstString.count < secondString.count ?
        let biggestString = firstString.count >= secondString.count ?

        var evenLCS = Array(repeating: [String](), count: smallestStr
        var oddLCS = Array(repeating: [String](), count: smallestStri

        for i in stride(from: 1, to: biggestString.count + 1, by: 1) {
            if i % 2 == 0 {
                secondSolutionHelper(i, biggestString, smallestString,
            } else {
                secondSolutionHelper(i, biggestString, smallestString,
            }
        }

        return biggestString.count % 2 == 0 ? evenLCS[smallestString.c
    }

    func secondSolutionHelper(_ i: Int, _ biggestString: String, _ sma
        for j in stride(from: 1, to: smallestString.count + 1, by: 1)
            let firstIndex = biggestString.index(biggestString.startIn
            let secondIndex = smallestString.index(smallestString.star

            if biggestString[firstIndex] == smallestString[secondIndex
                var diagonal = previousLCS[j - 1]
                let char = String(smallestString[secondIndex])
                diagonal.append(char)

                currentLCS[j] = diagonal
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```swift
class Program {
    func longestCommonSubsequence(firstString: String, secondString: S
        // Write your code here.
        return []
    }
}

```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.