

Our Solution(s)

Run Code

Your Solutions

Run Code

| Solution 1  | Solution 2 | Solution 3 | Solution 4 |
|---|------------|------------|------------|
| <pre>1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved. 2 3 class Program { 4     // O(nm * min(n, m)) time   O(min(n, m)^2) space 5     func longestCommonSubsequence(firstString: String, secondString: String) -&gt; String { 6         let smallestString = firstString.count &lt; secondString.count ? firstString : secondString 7         let biggestString = firstString.count &gt;= secondString.count ? firstString : secondString 8 9         var evenLCS = Array(repeating: [String](), count: smallestString.count + 1) 10        var oddLCS = Array(repeating: [String](), count: smallestString.count + 1) 11 12        for i in stride(from: 1, to: biggestString.count + 1, by: 1) { 13            if i % 2 == 0 { 14                secondSolutionHelper(i, biggestString, smallestString, evenLCS, oddLCS) 15            } else { 16                secondSolutionHelper(i, biggestString, smallestString, oddLCS, evenLCS) 17            } 18        } 19 20        return biggestString.count % 2 == 0 ? evenLCS[smallestString.count] : oddLCS[smallestString.count] 21    } 22 23    func secondSolutionHelper(_ i: Int, _ biggestString: String, _ smallestString: String, _ evenLCS: [String], _ oddLCS: [String]) { 24        for j in stride(from: 1, to: smallestString.count + 1, by: 1) { 25            let firstIndex = biggestString.index(biggestString.startIndex, offsetBy: i - 1) 26            let secondIndex = smallestString.index(smallestString.startIndex, offsetBy: j - 1) 27 28            if biggestString[firstIndex] == smallestString[secondIndex] { 29                var diagonal = previousLCS[j - 1] 30                let char = String(smallestString[secondIndex]) 31                diagonal.append(char) 32 33                currentLCS[j] = diagonal</pre> |            |            |            |
| Solution 1  | Solution 2 | Solution 3 |            |
| <pre>1 class Program { 2     func longestCommonSubsequence(firstString: String, secondString: String) -&gt; String { 3         // Write your code here. 4         return [] 5     } 6 } 7</pre>   |            |            |            |

Our Tests

Custom Output

Submit Code

```
1 class Program {
2     func longestCommonSubsequence(firstString: String, secondString: String) -> String {
3         // Write your code here.
4         return []
5     }
6 }
7
```

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     // O(nm * min(n, m)) time | O(min(n, m)^2) space
5     func longestCommonSubsequence(firstString: String, secondString: String) -> String {
6         let smallestString = firstString.count < secondString.count ? firstString : secondString
7         let biggestString = firstString.count >= secondString.count ? firstString : secondString
8
9         var evenLCS = Array(repeating: [String](), count: smallestString.count + 1)
10        var oddLCS = Array(repeating: [String](), count: smallestString.count + 1)
11
12        for i in stride(from: 1, to: biggestString.count + 1, by: 1) {
13            if i % 2 == 0 {
14                secondSolutionHelper(i, biggestString, smallestString, evenLCS, oddLCS)
15            } else {
16                secondSolutionHelper(i, biggestString, smallestString, oddLCS, evenLCS)
17            }
18        }
19
20        return biggestString.count % 2 == 0 ? evenLCS[smallestString.count] : oddLCS[smallestString.count]
21    }
22
23    func secondSolutionHelper(_ i: Int, _ biggestString: String, _ smallestString: String, _ evenLCS: [String], _ oddLCS: [String]) {
24        for j in stride(from: 1, to: smallestString.count + 1, by: 1) {
25            let firstIndex = biggestString.index(biggestString.startIndex, offsetBy: i - 1)
26            let secondIndex = smallestString.index(smallestString.startIndex, offsetBy: j - 1)
27
28            if biggestString[firstIndex] == smallestString[secondIndex] {
29                var diagonal = previousLCS[j - 1]
30                let char = String(smallestString[secondIndex])
31                diagonal.append(char)
32
33                currentLCS[j] = diagonal
```

```
10         self.assertAlmostEqual(self.epsilon, self.epsilon, self.epsilon)
11     def test_epsilon(self):
12         self.assertAlmostEqual(self.epsilon, self.epsilon, self.epsilon)
13     def test_epsilon(self):
14         self.assertAlmostEqual(self.epsilon, self.epsilon, self.epsilon)
15     def test_epsilon(self):
16         self.assertAlmostEqual(self.epsilon, self.epsilon, self.epsilon)
17     def test_epsilon(self):
18         self.assertAlmostEqual(self.epsilon, self.epsilon, self.epsilon)
19     def test_epsilon(self):
20         self.assertAlmostEqual(self.epsilon, self.epsilon, self.epsilon)
21     def test_epsilon(self):
22         self.assertAlmostEqual(self.epsilon, self.epsilon, self.epsilon)
23     def test_epsilon(self):
24         self.assertAlmostEqual(self.epsilon, self.epsilon, self.epsilon)
25     def test_epsilon(self):
26         self.assertAlmostEqual(self.epsilon, self.epsilon, self.epsilon)
```

Run or submit code when you're ready.