Prompt | Scratchpad | Our Solution(s) | Video Explanation | Run Code

Solution 1 | Solution 2

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

class Program {
    // O(nd) time | O(min(n, m) space)
    func levenshteinDistance(firstString: String, secondString: String) -> Int {
        let small = firstString.count < secondString.count ? firstString : secondString
        let big = firstString.count >= secondString.count ? firstString : secondString

        var evenEdits = [Int]()
        var oddEdits = Array(repeating: 0, count: small.count + 1)

        for i in 0 ..< small.count + 1 {
            evenEdits.append(i)
        }

        for i in 1 ..< big.count + 1 {
            if i % 2 == 1 {
                optimizedLevenshteinHelper(bigIndex: i, smallString: small, bigString: big,
            } else {
                optimizedLevenshteinHelper(bigIndex: i, smallString: small, bigString: big,
            }
        }

        return big.count % 2 == 0 ? evenEdits[small.count] : oddEdits[small.count]
    }

    func optimizedLevenshteinHelper(bigIndex: Int, smallString: String, bigString: String,
        currentEdits[0] = bigIndex

        for j in 1 ..< smallString.count + 1 {
            let firstIndex = bigString.index(bigString.startIndex, offsetBy: bigIndex - 1)
            let secondIndex = smallString.index(smallString.startIndex, offsetBy: j - 1)

            if bigString[firstIndex] == smallString[secondIndex] {
                currentEdits[j] = previousEdits[j - 1]
            } else {
                currentEdits[j] = 1 + min(previousEdits[j], min(previousEdits[j - 1], curre
            }
        }
    }
}
```

Your Solutions | Run Code

Solution 1 | Solution 2 | Solution 3

```swift
class Program {
    func levenshteinDistance(firstString: String, secondString: String) -> Int {
        // Write your code here.
        return -1
    }
}

```

Custom Output | Raw Output | Submit Code

Run or submit code when you're ready.