Our Solution(s) Run Code

Solution 1 Solution 2

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

package main

// O(nm) time | O(nm) space
func LevenshteinDistance(a, b string) int {
  edits := make([][]int, len(b)+1)
  for y := range edits {
    edits[y] = make([]int, len(a)+1)
    for x := range edits[y] {
      edits[y][x] = x
    }
  }
  for i := 1; i < len(b)+1; i++ {
    edits[i][0] = edits[i-1][0] + 1
  }

  for i := 1; i < len(b)+1; i++ {
    for j := 1; j < len(a)+1; j++ {
      if b[i-1] == a[j-1] {
        edits[i][j] = edits[i-1][j-1]
      } else {
        edits[i][j] = 1 + min(edits[i-1][j-1], edits[i-1][j], ed
      }
    }
  }
  return edits[len(b)][len(a)]
}

func min(args ...int) int {
  curr := args[0]
  for _, num := range args {
    if curr > num {
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```go
package main

func LevenshteinDistance(a, b string) int {
  // Write your code here.
  return -1
}

```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.