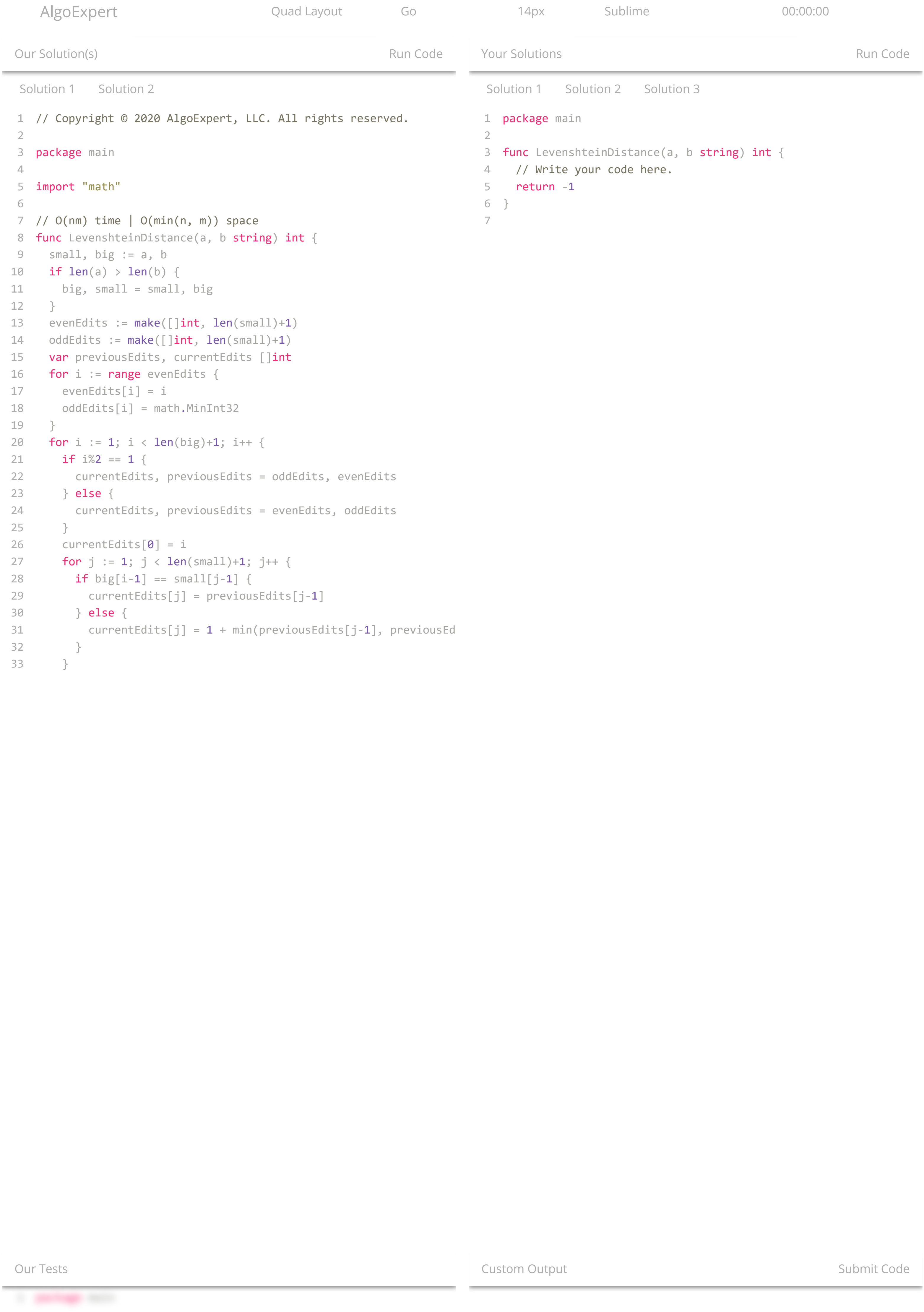

```go
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

package main

import "math"

// O(nm) time | O(min(n, m)) space
func LevenshteinDistance(a, b string) int {
  small, big := a, b
  if len(a) > len(b) {
    big, small = small, big
  }
  evenEdits := make([]int, len(small)+1)
  oddEdits := make([]int, len(small)+1)
  var previousEdits, currentEdits []int
  for i := range evenEdits {
    evenEdits[i] = i
    oddEdits[i] = math.MinInt32
  }
  for i := 1; i < len(big)+1; i++ {
    if i%2 == 1 {
      currentEdits, previousEdits = oddEdits, evenEdits
    } else {
      currentEdits, previousEdits = evenEdits, oddEdits
    }
    currentEdits[0] = i
    for j := 1; j < len(small)+1; j++ {
      if big[i-1] == small[j-1] {
        currentEdits[j] = previousEdits[j-1]
      } else {
        currentEdits[j] = 1 + min(previousEdits[j-1], previousEd
      }
    }
```

```go
package main

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

```

```go
package main
```

Run or submit code when you're ready.