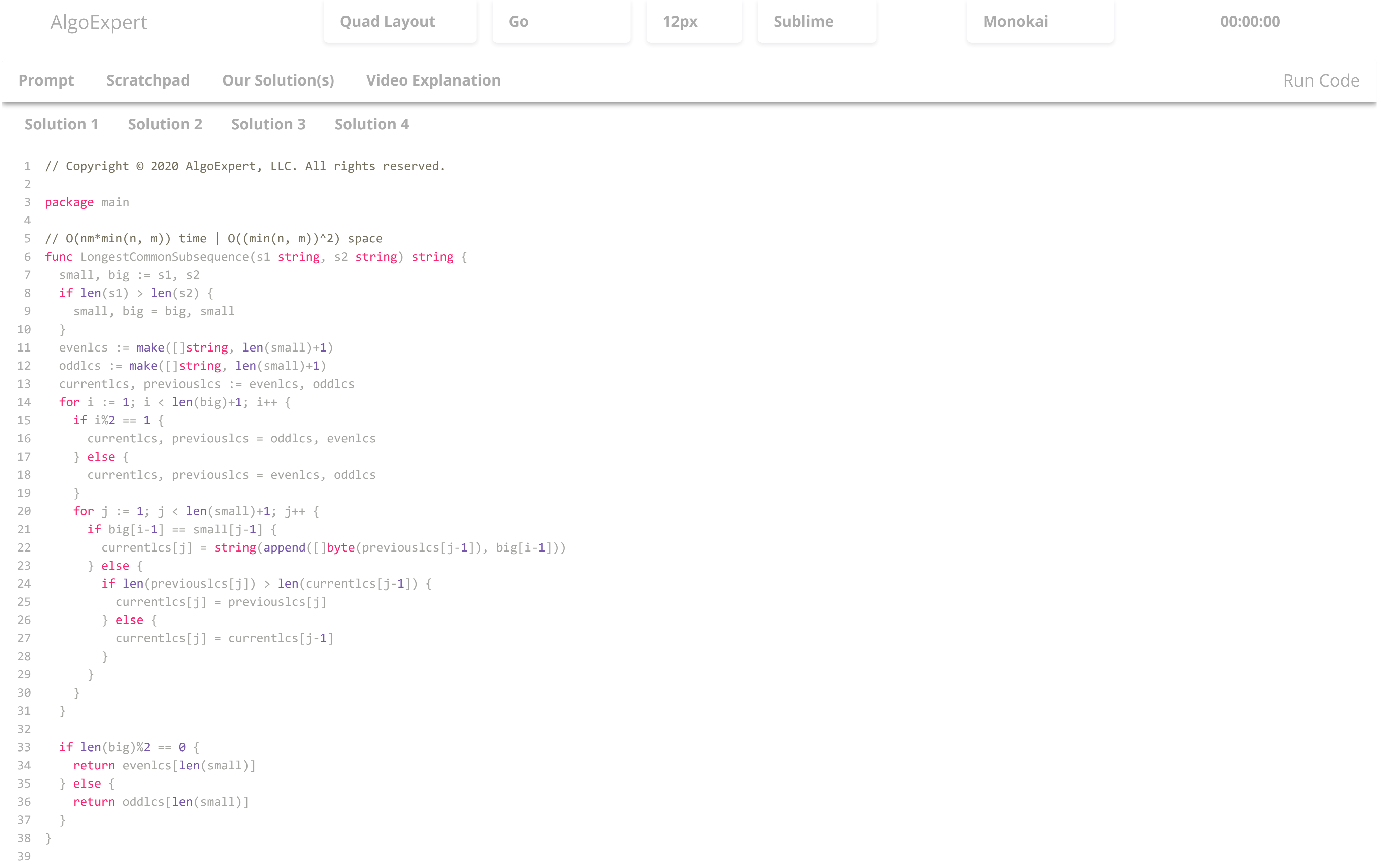

```go
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

package main

// O(nm*min(n, m)) time | O((min(n, m))^2) space
func LongestCommonSubsequence(s1 string, s2 string) string {
  small, big := s1, s2
  if len(s1) > len(s2) {
    small, big = big, small
  }
  evenlcs := make([]string, len(small)+1)
  oddlcs := make([]string, len(small)+1)
  currentlcs, previouslcs := evenlcs, oddlcs
  for i := 1; i < len(big)+1; i++ {
    if i%2 == 1 {
      currentlcs, previouslcs = oddlcs, evenlcs
    } else {
      currentlcs, previouslcs = evenlcs, oddlcs
    }
    for j := 1; j < len(small)+1; j++ {
      if big[i-1] == small[j-1] {
        currentlcs[j] = string(append([]byte(previouslcs[j-1]), big[i-1]))
      } else {
        if len(previouslcs[j]) > len(currentlcs[j-1]) {
          currentlcs[j] = previouslcs[j]
        } else {
          currentlcs[j] = currentlcs[j-1]
        }
      }
    }
  }

  if len(big)%2 == 0 {
    return evenlcs[len(small)]
  } else {
    return oddlcs[len(small)]
  }
}
```