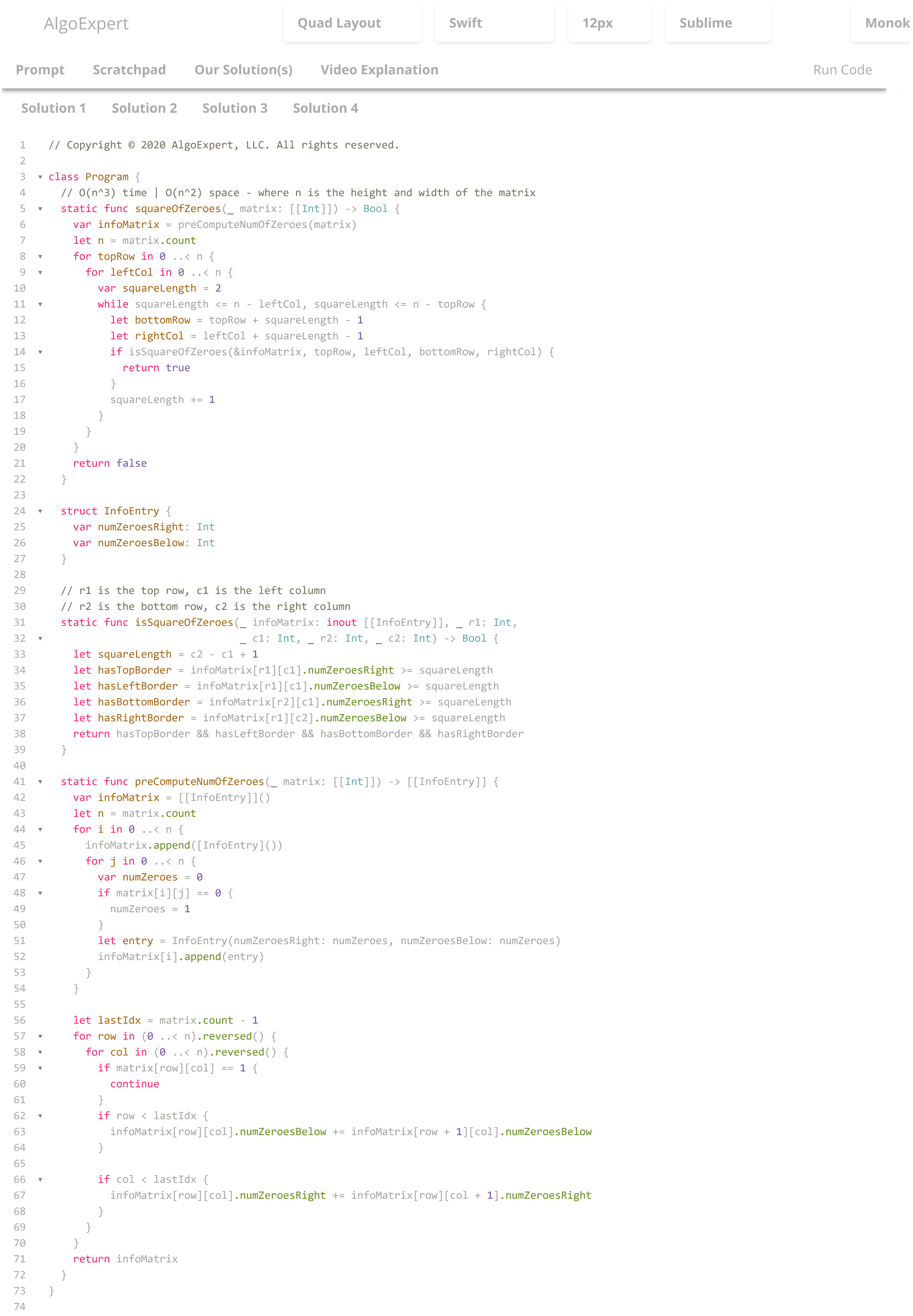

```swift
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

class Program {
  // O(n^3) time | O(n^2) space - where n is the height and width of the matrix
  static func squareOfZeroes(_ matrix: [[Int]]) -> Bool {
    var infoMatrix = preComputeNumOfZeroes(matrix)
    let n = matrix.count
    for topRow in 0 ..< n {
      for leftCol in 0 ..< n {
        var squareLength = 2
        while squareLength <= n - leftCol, squareLength <= n - topRow {
          let bottomRow = topRow + squareLength - 1
          let rightCol = leftCol + squareLength - 1
          if isSquareOfZeroes(&infoMatrix, topRow, leftCol, bottomRow, rightCol) {
            return true
          }
          squareLength += 1
        }
      }
    }
    return false
  }

  struct InfoEntry {
    var numZeroesRight: Int
    var numZeroesBelow: Int
  }

  // r1 is the top row, c1 is the left column
  // r2 is the bottom row, c2 is the right column
  static func isSquareOfZeroes(_ infoMatrix: inout [[InfoEntry]], _ r1: Int,
                               _ c1: Int, _ r2: Int, _ c2: Int) -> Bool {
    let squareLength = c2 - c1 + 1
    let hasTopBorder = infoMatrix[r1][c1].numZeroesRight >= squareLength
    let hasLeftBorder = infoMatrix[r1][c1].numZeroesBelow >= squareLength
    let hasBottomBorder = infoMatrix[r2][c1].numZeroesRight >= squareLength
    let hasRightBorder = infoMatrix[r1][c2].numZeroesBelow >= squareLength
    return hasTopBorder && hasLeftBorder && hasBottomBorder && hasRightBorder
  }

  static func preComputeNumOfZeroes(_ matrix: [[Int]]) -> [[InfoEntry]] {
    var infoMatrix = [[InfoEntry]]()
    let n = matrix.count
    for i in 0 ..< n {
      infoMatrix.append([InfoEntry]())
      for j in 0 ..< n {
        var numZeroes = 0
        if matrix[i][j] == 0 {
          numZeroes = 1
        }
        let entry = InfoEntry(numZeroesRight: numZeroes, numZeroesBelow: numZeroes)
        infoMatrix[i].append(entry)
      }
    }

    let lastIdx = matrix.count - 1
    for row in (0 ..< n).reversed() {
      for col in (0 ..< n).reversed() {
        if matrix[row][col] == 1 {
          continue
        }
        if row < lastIdx {
          infoMatrix[row][col].numZeroesBelow += infoMatrix[row + 1][col].numZeroesBelow
        }

        if col < lastIdx {
          infoMatrix[row][col].numZeroesRight += infoMatrix[row][col + 1].numZeroesRight
        }
      }
    }
    return infoMatrix
  }
}
```