AlgoExpert Quad Layout C++ 12px Sublime Monok

Prompt Scratchpad Our Solution(s) Video Explanation Run Code

Solution 1 Solution 2 Solution 3 Solution 4

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

using namespace std;

struct InfoMatrixItem {
  int numZeroesBelow;
  int numZeroesRight;
};

bool isSquareOfZeroes(
  vector<vector<InfoMatrixItem>> infoMatrix,
  int r1,
  int c1,
  int r2,
  int c2
);
vector<vector<InfoMatrixItem>> preComputedNumOfZeroes(vector<vector<int>> matrix);

// O(n^3) time | O(n^2) space - where n is the height and width of the matrix
bool squareOfZeroes(vector<vector<int>> matrix) {
  vector<vector<InfoMatrixItem>> infoMatrix = preComputedNumOfZeroes(matrix);
  int n = matrix.size();
  for (int topRow = 0; topRow < n; topRow++) {
    for (int leftCol = 0; leftCol < n; leftCol++) {
      int squareLength = 2;
      while (squareLength <= n - leftCol && squareLength <= n - topRow) {
        int bottomRow = topRow + squareLength - 1;
        int rightCol = leftCol + squareLength - 1;
        if (isSquareOfZeroes(infoMatrix, topRow, leftCol, bottomRow, rightCol)) return true;
        squareLength++;
      }
    }
  }
  return false;
}

// r1 is the top row, c1 is the left column
// r2 is the bottom row, c2 is the right column
bool isSquareOfZeroes(
  vector<vector<InfoMatrixItem>> infoMatrix,
  int r1,
  int c1,
  int r2,
  int c2
) {
  int squareLength = c2 - c1 + 1;
  bool hasTopBorder = infoMatrix[r1][c1].numZeroesRight >= squareLength;
  bool hasLeftBorder = infoMatrix[r1][c1].numZeroesBelow >= squareLength;
  bool hasBottomBorder = infoMatrix[r2][c1].numZeroesRight >= squareLength;
  bool hasRightBorder = infoMatrix[r1][c2].numZeroesBelow >= squareLength;
  return hasTopBorder && hasLeftBorder && hasBottomBorder && hasRightBorder;
}

vector<vector<InfoMatrixItem>> preComputedNumOfZeroes(vector<vector<int>> matrix) {
  vector<vector<InfoMatrixItem>> infoMatrix;
  for (int i = 0; i < matrix.size(); i++) {
    vector<InfoMatrixItem> inner;
    for (int j = 0; j < matrix[i].size(); j++) {
      int numZeroes = matrix[i][j] == 0 ? 1 : 0;
      inner.push_back(InfoMatrixItem {numZeroes, numZeroes});
    }
    infoMatrix.push_back(inner);
  }

  int lastIdx = matrix.size() - 1;
  for (int row = lastIdx; row >= 0; row--) {
    for (int col = lastIdx; col >= 0; col--) {
      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;
}
```