Leetcode May Challenge DAY: 21

1. Python

```
class Solution:

  def countSquares(self, matrix: List[List[int]]) -> int:


    for i in range(1, len(matrix)):

      for j in range(1, len(matrix[0])):

        if matrix[i][j] == 0:

          continue

        matrix[i][j] = min(matrix[i-1][j-1], matrix[i][j-1],matrix[i-1][j]) + 1


    s = 0

    for row in matrix:

      s += sum(row)


    return s
```

2. C++

```
int countSquares(vector<vector<int>>& matrix) {
    const int n = size(matrix);
    const int m = n ? size(matrix.front()) : 0;
    int ans = 0;
    vector<vector<int>> dp(n + 1, vector<int>(m + 1));
    for (int i = 1; i <= n; ++i)
      for (int j = 1; j <= m; ++j)
        if (matrix[i - 1][j - 1] == 1)
          ans += (dp[i][j] = 1 + min(dp[i - 1][j - 1], min(dp[i - 1][j], dp[i][j - 1])));
    return ans;
  }
```

3. JAVA

```
class Solution {
    public int countSquares(int[][] matrix) {
        int count = 0;

        if(matrix != null && matrix.length > 0){
            int R = matrix.length;
            int C = matrix[0].length;
            int i, j;
            int[][] DP = new int[R + 1][C + 1];

            for(i = 1; i <= R; i++)
                for(j = 1; j <= C; j++)
                    if(matrix[i - 1][j - 1] == 1){
                        DP[i][j] = 1 + Math.min(DP[i - 1][j - 1], Math.min(DP[i][j - 1], DP[i - 1][j]));
                        count = count + DP[i][j];
                    }
        }

        return count;
    }
}
```