/\*\*

 \* @param {character[][]} matrix

 \* @return {number}

 \*/

var maximalRectangle = function(matrix) {

    if (matrix.length === 0 || matrix[0].length === 0) return 0;

    const rows = matrix.length;

    const cols = matrix[0].length;

    const heights = new Array(cols).fill(0);

    let maxArea = 0;

    for (let i = 0; i < rows; i++) {

        // 更新高度数组

        for (let j = 0; j < cols; j++) {

            heights[j] = matrix[i][j] === '1' ? heights[j] + 1 : 0;

        }

        // 计算当前行的最大矩形面积

        maxArea = Math.max(maxArea, largestRectangleArea(heights));

    }

    return maxArea;

};

// 单调栈解法计算柱状图最大矩形面积

function largestRectangleArea(heights) {

    heights = [0, ...heights, 0];

    const stack = [0];

    let maxArea = 0;

    for (let i = 1; i < heights.length; i++) {

        while (heights[i] < heights[stack[stack.length - 1]]) {

            const height = heights[stack.pop()];

            const width = i - stack[stack.length - 1] - 1;

            maxArea = Math.max(maxArea, height \* width);

        }

        stack.push(i);

    }

    return maxArea;

}

