# 剑指47 礼物的最大价值

标准的二维动态规划。ok。

```cpp
// my answer
// 多开一行一列的空间能够让代码更简洁
class Solution {
public:
    int maxValue(vector<vector<int>>& grid) {
        int m = grid.size();
        int n = grid[0].size();
        vector<vector<int>> dp(m + 1, vector<int>(n + 1, 0));
        for (int i = 1; i <= m; i++) {
            for (int j = 1; j <= n; j++) {
                dp[i][j] = max(dp[i][j-1], dp[i-1][j]) + grid[i-1][j-1];
            }
        }
        return dp[m][n];
    }
};
```

```cpp
// 其他细节区别：
// 直接使用原二维数组保存结果，不使用额外空间了。且提前初始化第一行和第一列，避免分类讨论。
class Solution {
    public int maxValue(int[][] grid) {
        int m = grid.length, n = grid[0].length;
        for(int j = 1; j < n; j++) // 初始化第一行
            grid[0][j] += grid[0][j - 1];
        for(int i = 1; i < m; i++) // 初始化第一列
            grid[i][0] += grid[i - 1][0];
        for(int i = 1; i < m; i++)
            for(int j = 1; j < n; j++)
                grid[i][j] += Math.max(grid[i][j - 1], grid[i - 1][j]);
        return grid[m - 1][n - 1];
    }
}

```