# 题目

Follow up for "Unique Paths":  
Now consider if some obstacles are added to the grids. How many unique paths would there be?  
An obstacle and empty space is marked as 1 and 0 respectively in the grid.  
For example,  
There is one obstacle in the middle of a 3x3 grid as illustrated below.

[

　[0,0,0],

　[0,1,0],

　[0,0,0]

]

The total number of unique paths is 2.

Note: m and n will be at most 100.

# 解题思路

1. 当(i, j)有障碍时dp[i][j] = 0
2. dp[0][j]和dp[i][0]未必为1.  
   dp[0][j] = obstacleGrid[0][j] ? 0 : dp[0][j-1]  
   dp[i][0] = obstacleGrid[i][0] ? 0 : dp[i-1][0]
3. 当obstacleGrid [0][0] = 1时，return 0

# 解题代码

class Solution {

public:

int uniquePathsWithObstacles(const vector<vector<int>>& obstacleGrid) const {

if (obstacleGrid[0][0] == 1) return 0;

vector<int>::size\_type m = obstacleGrid.size();

vector<int>::size\_type n = obstacleGrid[0].size();

vector<vector<int>> table(m, vector<int>(n));

for (vector<int>::size\_type i = 1; i < m; i++) {

//table[i-1][0] 对于 table[i][0] 非常重要，决定了后面节点是否可达

table[i][0] = (table[i - 1][0] && obstacleGrid[i][0] == 0) ? 1 : 0;

}

for (vector<int>::size\_type i = 1; i < n; i++) {

//table[0][i] 对于 table[0][i-1] 非常重要，决定了后面节点是否可达

table[0][i] = (table[0][i - 1] && obstacleGrid[0][i] == 0) ? 1 : 0;

}

for (vector<int>::size\_type i = 1; i < m; i++) {

for (vector<int>::size\_type j = 1; j < n; j++) {

table[i][j] = table[i][j - 1] + table[i - 1][j];

if (obstacleGrid[i][j] == 1)

table[i][j] = 0;

}

}

return obstacleGrid[m - 1][n - 1] ? 0 : table[m - 1][n - 1];

}

};