# 题目描述

Given a triangle, find the minimum path sum from top to bottom. Each step you may move to adjacent numbers on the row below.

For example, given the following triangle

[

[2],

[3,4],

[6,5,7],

[4,1,8,3]

]

The minimum path sum from top to bottom is 11 (i.e., 2 + 3 + 5 + 1 = 11).

Note:

Bonus point if you are able to do this using only O(n) extra space, where n is the total number of rows in the triangle.

# 解题思路

因为只有两个方向：下、右下；所以递推公式局部是

dp[m + 1][n] = min(dp[m][n], dp[m][n - 1]) + triangle[m + 1][n]

解题的时候要从一般到特殊，既边缘 case 放在后面考虑（边缘 case 的解决属于思维收敛）。  
完整的递推公式如下

if n > 1

dp[m + 1][n] = min(dp[m][n], dp[m][n - 1]) + triangle[m + 1][n]

else

dp[m + 1][0] = dp[m][0] + triangle[m + 1][0]

横向遍历三角形的时候，需要从右向左倒叙地遍历元素，因为 dp table 压缩存储方式成一维空间以后，会面临 dp table 的值被覆盖的问题。

# 代码实现

class Solution {

public:

int minimumTotal(const vector<vector<int>>& triangle) const {

vector<int> v(triangle.size(), INT\_MAX);

v[0] = triangle[0][0];

for (size\_t i = 1; i < triangle.size(); i++) {

for (int j = i; j >= 0; j--) {

switch (j)

{

case 0:

v[0] += triangle[i][0];

break;

default:

v[j] = min(v[j], v[j - 1]) + triangle[i][j];

break;

}

}

}

return \*min\_element(v.begin(), v.end());

}

};