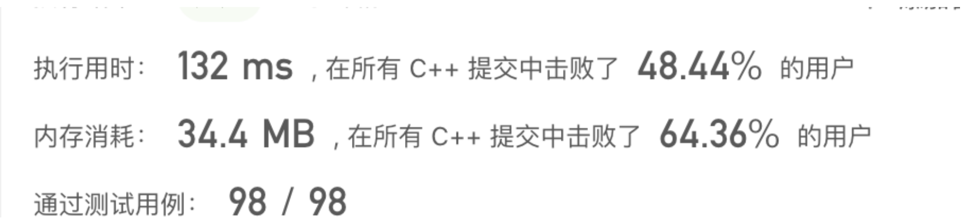
1975 Maximum Matrix Sum



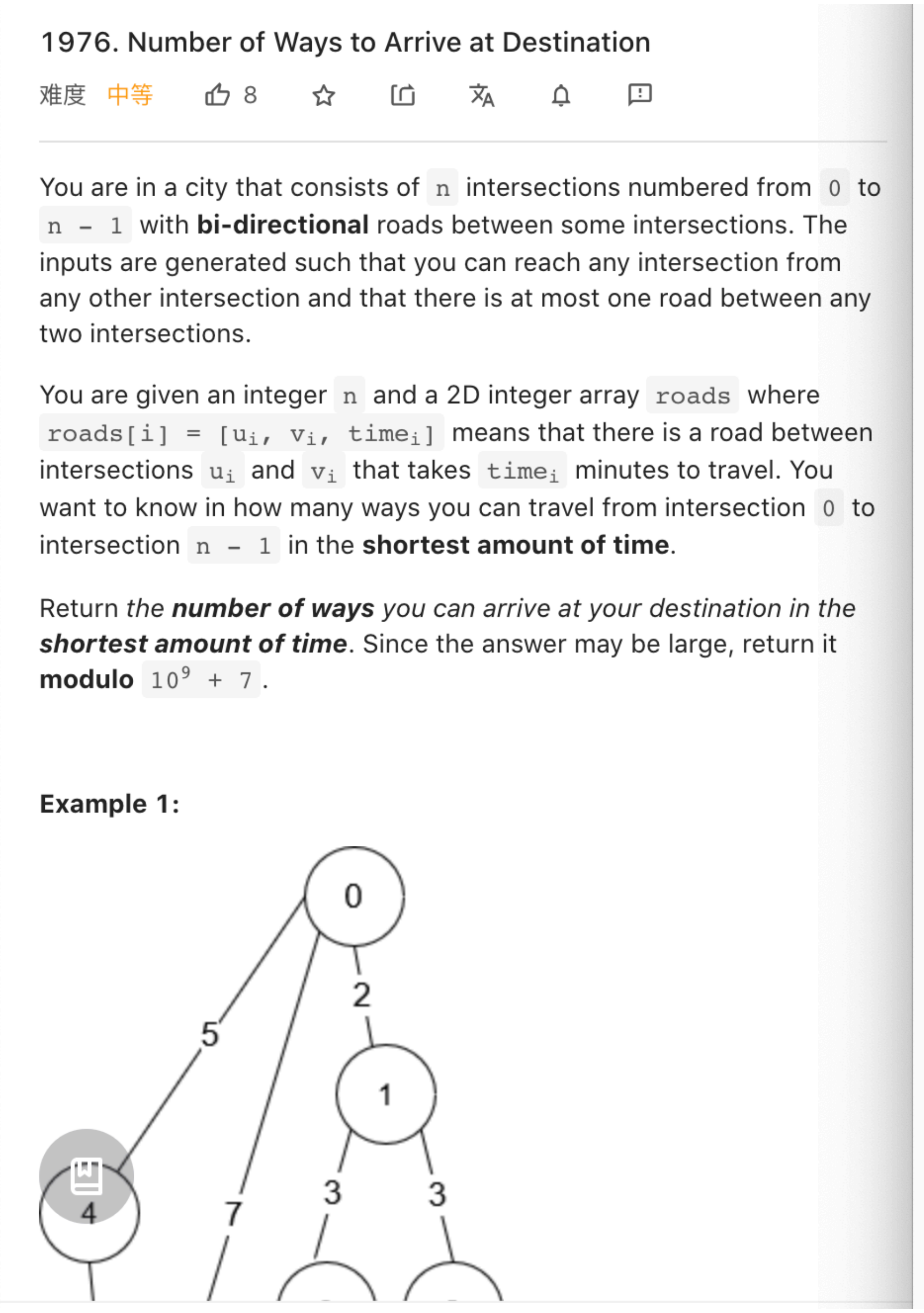
执行用时： 132 ms ，在所有 C++ 提交中击败了 48.44% 的用户

内存消耗： 34.4 MB ，在所有 C++ 提交中击败了 64.36% 的用户

通过测试用例： 98 / 98

```
/*
    脑经急转弯题目
    如果是奇数个负数，一定会剩一个
    如果是偶数个负数，一定能全部变正数
*/

class Solution {
public:
    long long maxMatrixSum(vector<vector<int>>& matrix) {
        int row = matrix.size();
        int col = matrix[0].size();

        int negNum = 0;
        long long totalSum = 0;
        int minValue = abs(matrix[0][0]);
        for(int i = 0; i < row; i++){
            for(int j = 0; j < col; j++){
                if(matrix[i][j] < 0){
                    negNum++;
                }

                int val = std::abs(matrix[i][j]);
                totalSum += val;
                minValue = std::min(minValue, val);
            }
        }

        return negNum % 2 == 0 ? totalSum : (totalSum - 2 * minValue);
    }
};
```

1976 Number of Ways to Arrive at Destination



1976. Number of Ways to Arrive at Destination

难度 中等 8

You are in a city that consists of n intersections numbered from 0 to $n - 1$ with **bi-directional** roads between some intersections. The inputs are generated such that you can reach any intersection from any other intersection and that there is at most one road between any two intersections.

You are given an integer n and a 2D integer array `roads` where $roads[i] = [u_i, v_i, time_i]$ means that there is a road between intersections u_i and v_i that takes $time_i$ minutes to travel. You want to know in how many ways you can travel from intersection 0 to intersection $n - 1$ in the **shortest amount of time**.

Return *the **number of ways** you can arrive at your destination in the **shortest amount of time***. Since the answer may be large, return it **modulo** $10^9 + 7$.

Example 1: