NOIP 普及组复赛 C 类题解思路(C++)

------2017 T3

棋盘

有一个 m × m 的棋盘，棋盘上每一个格子可能是红色、黄色或没有任何颜色的。你现在要从棋盘的最左上角走到棋盘的最右下角。任何一个时刻，你所站在的位置必须是有颜色的（不能是无色的），你只能向上、下、左、右四个方向前进。当你从一个格子走向另一个格子时，如果两个格子的颜色相同，那你不需要花费金币；如果不同，则你需要花费 1 个金币。另外，你可以花费 2 个金币施展魔法让下一个无色格子暂时变为你指定的颜色。但这个魔法不能连续使用，而且这个魔法的持续时间很短，也就是说，如果你使用了这个魔法，走到了这个暂时有颜色的格子上，你就不能继续使用魔法；只有当你离开这个位置，走到一个本来就有颜色的格子上的时候，你才能继续使用这个魔法，而当你离开了这个位置（施展魔法使得变为有颜色的格子）时，这个格子恢复为无色。现在你要从棋盘的最左上角，走到棋盘的最右下角，求花费的最少金币是多少？

输入

第一行包含两个正整数 m，n，以一个空格分开，分别代表棋盘的大小，棋盘上有颜色的格子的数量。接下来的 n 行，每行三个正整数 x，y，c，分别表示坐标为（x，y）的格子有颜色 c。其中 c=1 代表黄色，c=0 代表红色。相邻两个数之间用一个空格隔开。棋盘左上角的坐标为（1，1），右下角的坐标为（m，m）。棋盘上其余的格子都是无色。保证棋盘的左上角，也就是（1，1）一定是有颜色的。

输出

输出一行，一个整数，表示花费的金币的最小值，如果无法到达，输出-1。

样例输入 1

5 7

1 1 0

1 2 0

2 2 1

3 3 1

3 4 0

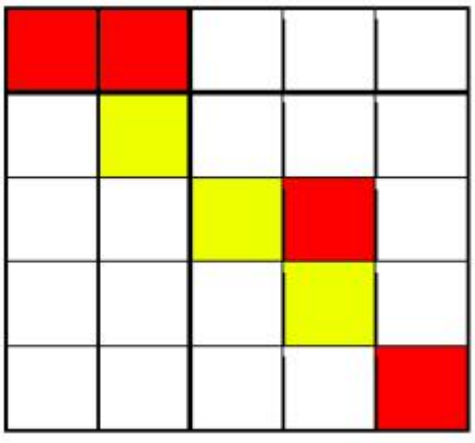
4 4 1

5 5 0

样例输出 1

8

说明：



从（1，1）开始，走到（1，2）不花费金币
从（1，2）向下走到（2，2）花费 1 枚金币
从（2，2）施展魔法，将（2，3）变为黄色，花费 2 枚金币

从（2，2）走到（2，3）不花费金币
从（2，3）走到（3，3）不花费金币
从（3，3）走到（3，4）花费 1 枚金币
从（3，4）走到（4，4）花费 1 枚金币
从（4，4）施展魔法，将（4，5）变为黄色，花费 2 枚金币
从（4，4）走到（4，5）不花费金币
从（4，5）走到（5，5）花费 1 枚金币

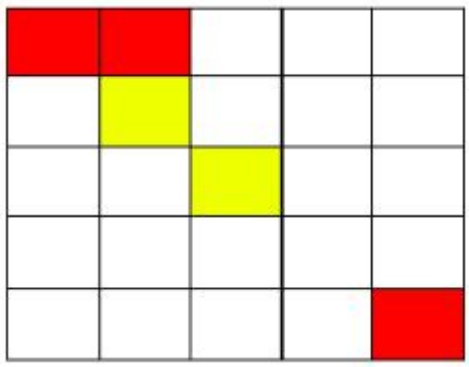
样例输入 2

```
5 5
1 1 0
1 2 0
2 2 1
3 3 1
5 5 0
```

样例输出 2

```
-1
```

说明



从（1，1）走到（1，2），不花费金币
从（1，2）走到（2，2），花费 1 金币
施展魔法将（2，3）变为黄色，从（2，2）到（2，3）花费 2 金币
从（2，3）走到（3，3）不花费金币
从（3，3）只能施展魔法到达（3，2），（2，3），（3，4），（4，3）
而从以上四点均无法到达（5，5），故无法到达终点，输出-1

数据规模与约定

对于 30%的数据，1 ≤ m ≤ 5， 1 ≤ n ≤ 10。
对于 60%的数据，1 ≤ m ≤ 20， 1 ≤ n ≤ 200。
对于 100%的数据，1 ≤ m ≤ 100， 1 ≤ n ≤ 1,000。

解 析

1、本题是算法中的经典迷宫（图论），何为经典迷宫，即是给定一个入口点和出口点，要求找到一条或者多条通路，并且求出最短的那条路径。

2、对于二维迷宫，一般用矩阵来构建数据结构，最基础的解法有深度优先搜索、宽度优先搜索、栈操作、回溯、减枝、递归、队列、贪心思想的综合应用，遇到问题求解一般是显示路径、找到最短路径、花费某种代价最少的路径。而本题就是求最小的花费代价。本题没有超出经典迷宫求解的算法范畴，这里给出 dfs（深搜）的题解思路。

3、首先我们可以确定，在一个二维矩阵中，某一个点，有四个方向的路径可以走，如图，一个 5×5 的迷宫矩阵，0 代表墙，不能走，1 代表路，可以走，现在给出入口点（1,1），出口点（5,5），要找出这个路径。

1	1	0	0	0
0	1	0	0	0
0	1	1	0	0

0	0	1	1	0
0	0	0	1	1

4、再看一下，每一个点都有 4 个方向可以走，但是在边界、或者遇到墙，则不可以走，于是，基于一个点去下一个点，限制条件应该有这么 3 个情况：

- *要走的点是边界以外*
- *要走的点是墙*
- *要走的点以前走过*

于是，基于当前点**(i,j)**，下一个点是**(i+1,j)**、**(i-1,j)**、**(i,j+1)**、**(i,j-1)**，注意，遇到边界、墙和已经走过的点，会相应去掉这几种情况。

	(i-1,j)	
(i,j-1)	**(i,j)**	**(i,j+1)**
	(i+1,j)	

5、从一点移动到另外一点，总有四个方向，到了另外一点，还是有四个方向，这样检阅下去，会发生两个情况，一是找到一条或多条路径，二是没找到，代表迷宫无解。

6、继续分析迷宫无解，这里我们先讨论在某一个点无解，就是当前点往下一步走，四个方向不是墙就是超出边界，这种情况代表当前路不通，需要在前一步换个方向，这就是回溯。具体到编码，用一个数值标记当前点是否走过，在试探过程中发现此路不通，需要把当前点的标记还原为原本状态，这样就可以换个方向继续试探了。如果通，因为已经标记过该点，所以下一个搜索不会再次搜该点了。

7、没错，每次试探的动作都一样，肯定会用到递归，说到递归，就是要找出**重复的动作体**和退出当前递归的条件，动作体是搜下个点，退出条件是**(i,j)**到达了终点，第二个退出条件是这个点已经被搜索过了，这就是记忆化搜索，需要再开一个数组，来记录该点是否被搜索，如果被搜索，直接返回。

8、对于本题，无非是把最短路径换成最少金币，都是求花费代价最少的结果。

9、**0** 代表一个颜色，**1** 代表另外一个颜色，**2** 代表无色。

10、三个情况，同色，异色，无色。无色只能用一次魔法。递归参数应该有 **x,y,coin,flag**，代表下一个搜索点**(i,j)**，走到下个点用的金币，标记是否本点是否使用魔法。

11、遇到无色，需要用魔法，就将下一个点颜色变成当前点颜色，这里没有理会具体变成哪种颜色，直接等于和该点同色，即是贪心一下，同时设置颜色回溯，把该点再标记成原本无色状态。

12、退出条件中，始终把最小花费更新，所有递归完成即可求得最终答案。