3	2	5	4	7	2	5	6
3	3	5	5	7	7	7	7

找规律

3	7	5	4	1	2	5	6
4	8	8	8	6	5	6	7
6	10	10	10	10	10	8	9
9	13	13	13	13	13	13	13

D……？ P……？

dynamic programming is a method for solving a complex problem by **breaking it down into a collection of simpler subproblems**.

Sublimation

Room 201

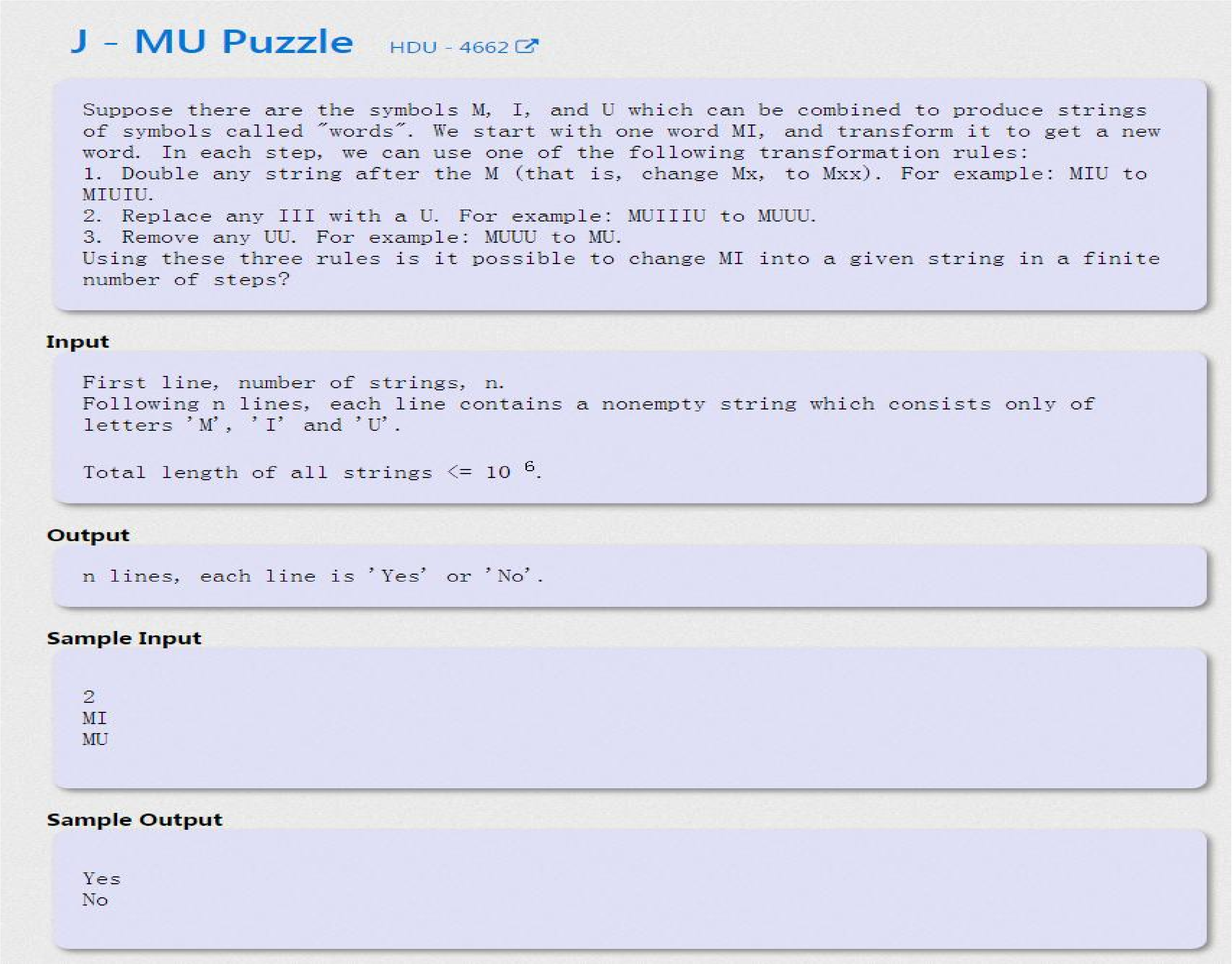
msc_sublimation@outlook.com

qq

Important？

有n个物品，对于第i个物品，能用ai~bi个

总共取m个的方案数？



J - MU Puzzle

HDU - 4662

Suppose there are the symbols M, I, and U which can be combined to produce strings of symbols called "words". We start with one word MI, and transform it to get a new word. In each step, we can use one of the following transformation rules:
1. Double any string after the M (that is, change Mx, to Mxx). For example: MIU to MIUIU.
2. Replace any III with a U. For example: MUIIIU to MUUU.
3. Remove any UU. For example: MUUU to MU.
Using these three rules is it possible to change MI into a given string in a finite number of steps?

Input

First line, number of strings, n.
Following n lines, each line contains a nonempty string which consists only of letters 'M', 'I' and 'U'.

Total length of all strings <= 10 ^{6}.

Output

n lines, each line is 'Yes' or 'No'.

Sample Input

```
2
MI
MU
```

Sample Output

```
Yes
No
```

- LCS
- LIS

- F[i][j]?

最长公共子序列

HDOJ-1159:

Sample Input

```
  abcfbc abfcab
  programming contest
  abcd mnp
```

Sample Output

```
  4
  2
  0
```

辅助空间变化示意图

	a	b	c	f	b	c
a	1	1	1	1	1	1
b	1	2	2	2	2	2
f	1	2	2	3	3	3
c	1	2	3	3	3	4
a	1	2	3	3	3	4
b	1	2	3	3	4	4

子问题(结构)特征：

$$f(i,j)=\begin{cases} f(i-1,j-1)+1 & (a[i]==b[j]) \\ \max(f(i-1,j),f(i,j-1)) & (a[i]!=b[j]) \end{cases}$$

- 由于f(i,j)只和f(i-1,j-1), f(i-1,j)和f(i,j-1)有关, 而在计算f(i,j)时, 只要选择一个合适的顺序, 就可以保证这三项都已经计算出来了, 这样就可以计算出f(i,j). 这样一直推到f(len(a),len(b))就得到所要求的解了.

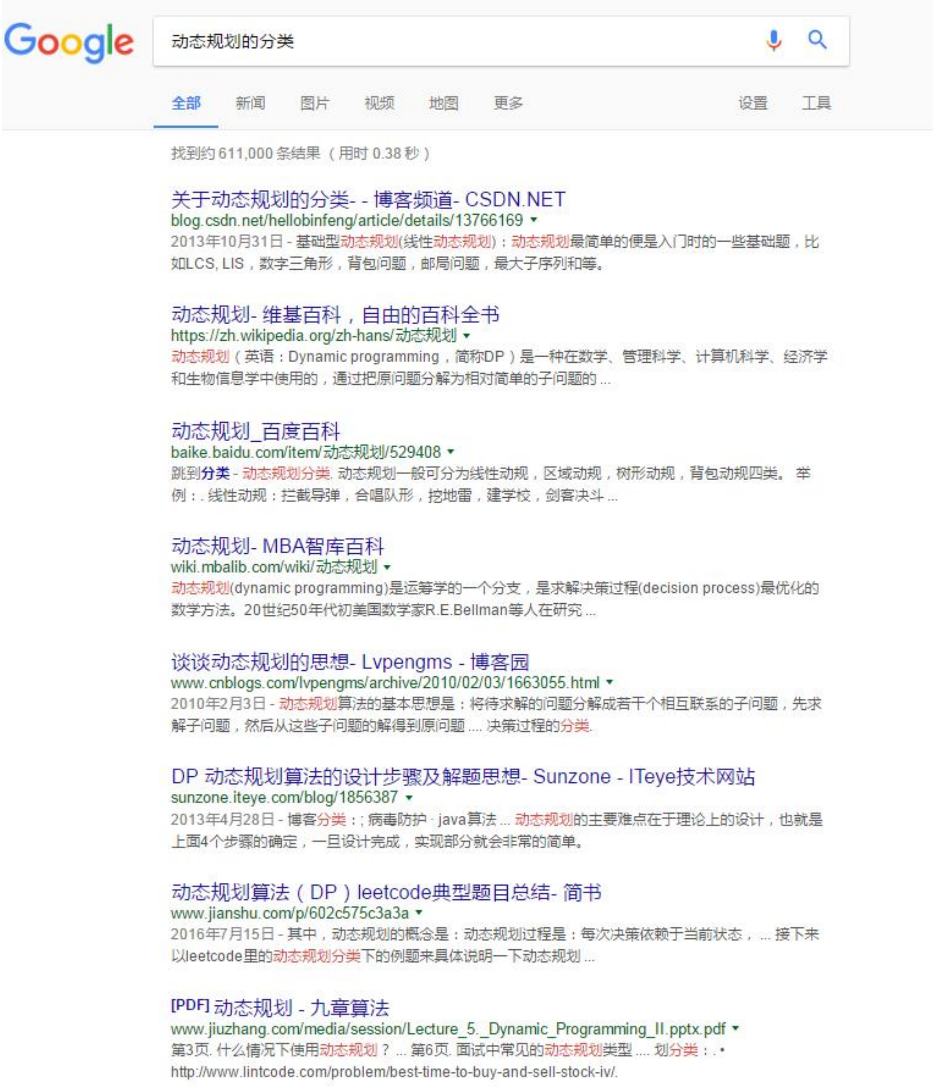
What！！！！

第一个串长度为1000000

第二个串长度为1000

Dp与分治

分类？



动态规划的分类

全部 新闻 图片 视频 地图 更多 设置 工具

找到约 611,000 条结果（用时 0.38 秒）

关于动态规划的分类- - 博客频道- CSDN.NET
blog.csdn.net/hellobinfeng/article/details/13766169 ▾
2013年10月31日 - 基础型动态规划(线性动态规划)：动态规划最简单的便是入门时的一些基础题，比如LCS, LIS，数字三角形，背包问题，邮局问题，最大子序列和等。

动态规划- 维基百科，自由的百科全书
https://zh.wikipedia.org/zh-hans/动态规划 ▾
动态规划（英语：Dynamic programming，简称DP）是一种在数学、管理科学、计算机科学、经济学和生物信息学中使用的，通过把原问题分解为相对简单的子问题的 ...

动态规划_百度百科
baike.baidu.com/item/动态规划/529408 ▾
跳到**分类** - 动态规划分类. 动态规划一般可分为线性动规，区域动规，树形动规，背包动规四类。 举例：. 线性动规：拦截导弹，合唱队形，挖地雷，建学校，剑客决斗 ...

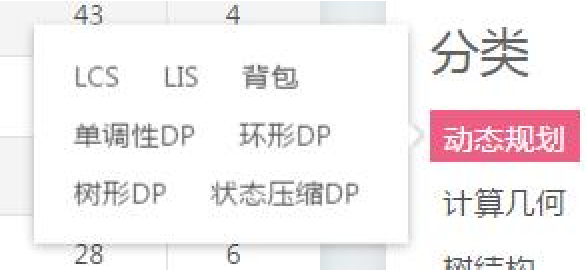
动态规划- MBA智库百科
wiki.mbalib.com/wiki/动态规划 ▾
动态规划(dynamic programming)是运筹学的一个分支，是求解决策过程(decision process)最优化的数学方法。20世纪50年代初美国数学家R.E.Bellman等人在研究 ...

谈谈动态规划的思想- Lvpengms - 博客园
www.cnblogs.com/lvpengms/archive/2010/02/03/1663055.html ▾
2010年2月3日 - 动态规划算法的基本思想是：将待求解的问题分解成若干个相互联系的子问题，先求解子问题，然后从这些子问题的解得到原问题 .... 决策过程的分类.

DP 动态规划算法的设计步骤及解题思想- Sunzone - ITeye技术网站
sunzone.iteye.com/blog/1856387 ▾
2013年4月28日 - 博客分类：; 病毒防护 · java算法 ... 动态规划的主要难点在于理论上的设计，也就是上面4个步骤的确定，一旦设计完成，实现部分就会非常的简单。

动态规划算法（DP）leetcode典型题目总结- 简书
www.jianshu.com/p/602c575c3a3a ▾
2016年7月15日 - 其中，动态规划的概念是：动态规划过程是：每次决策依赖于当前状态， ... 接下来以leetcode里的动态规划分类下的例题来具体说明一下动态规划 ...

[PDF] 动态规划 - 九章算法
www.jiuzhang.com/media/session/Lecture_5._Dynamic_Programming_II.pptx.pdf ▾
第3页. 什么情况下使用动态规划？ ... 第6页. 面试中常见的动态规划类型 .... 划分类：. • http://www.lintcode.com/problem/best-time-to-buy-and-sell-stock-iv/.



Outline?

区间
数轴（背包）

数位
集合(装压,插头)

树，dag，AC自动机、后缀树上

Outline！

最优化　　　　　　//统计(概率)

状态(写搜索+记忆化)

转移(决策,excel大法好，刷表法)

子问题（结构）

边界！！！敲黑板

套套套！！！

- 1、背包模型
  包括0-1背包、无限背包、有限背包、有价值背包等！
  2、最长非降子序列模型
  渡河问题、合唱队型等
  3、最大子段和模型
  K大子段和、最佳游览，最大子矩阵和等。
  4、LCS模型
  回文字串、多串的LCS等
  5、括号序列模型
  cf,（母函数、默慈金数等
  6、递推模型

  7、线段覆盖问题
  snoi 2012！

  8、连续段划分模型
  即要求把数列划分成k个连续段，使每段和的最大值最小。
  9、游戏模型
  这类题的阶段（一般是时间）和决策（一般就是游戏目标）很清楚，因此比较容易想到。改版：免费馅饼（NOI98）

背包

变种多！

- 变体积
- 转换维度
- 大体积
- 多重复（数学）

- 01背包
- 完全背包
- **多重背包**
- 混合三种背包
- 二维费用背包
- 分组背包
- 有依赖的背包

CCPC2015

有n个木条，放在桌子长L上，问最多放多少个

N<=1000,L<=10000

UVA 12260

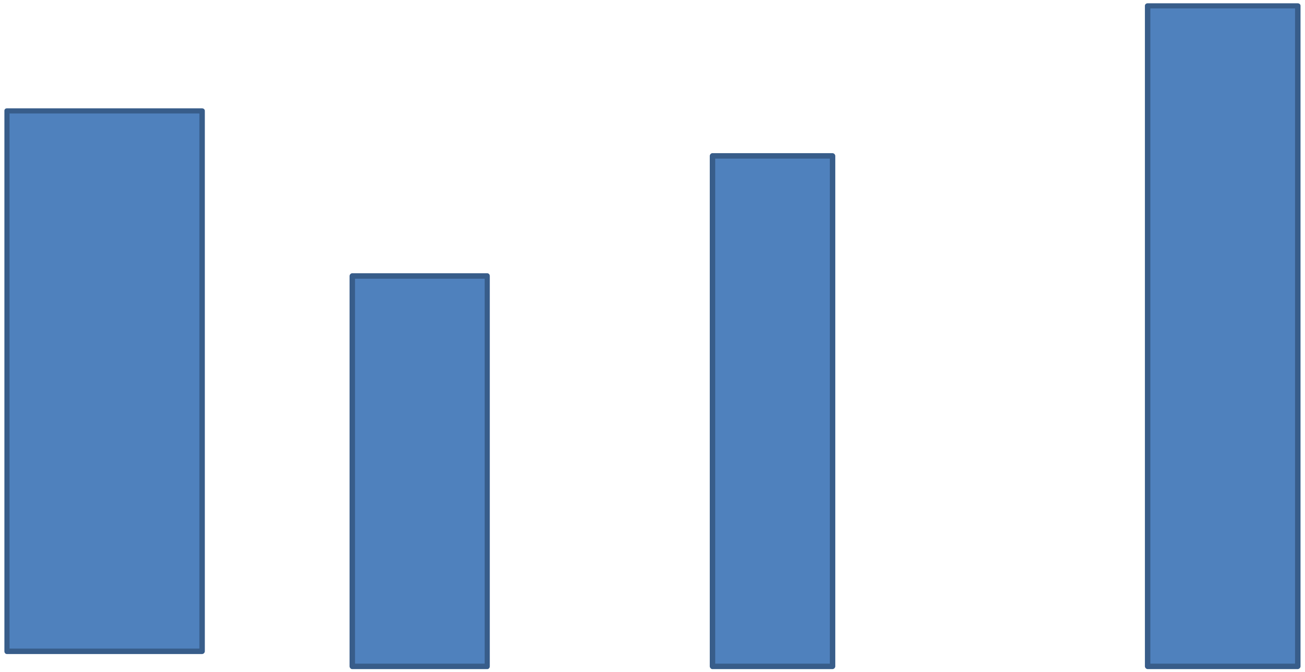
n(<=1000)个物品，一个味道指数，一个是金钱指数，艾老师和数老师玩游戏，两人轮流取，每次一个。因为A老师非常憨厚，每次取味道最好的（相等时让对方收益最高），S老师很机智，取**最终**能让自己收益最高的（相等时让对方美味度最低）

sweet<=1e9,value<=1e9

- 多次背包问题：给定**n** 种物品和一个背包。第**i**种物品 的价值是$\mathbf{W_i}$，其体积为$\mathbf{V_i}$，数量是$\mathbf{K_i}$件，背包的容量为**C**。可以任意选择装入背包中的物品，求装入背包中物品的最大总价值。

二进制

单调队列、单调栈



n个数，数值范围0~10^9，从中取出任意多个数，求多少种方案使得和为C的倍数。

1<=N<=1e6

1<=C<=1000

从n个可变体积的物品中取一些，第i个物品的体积满足l[i]~r[i]，获得的价值是基础价值a[i]加额外(p-l[i])*b[i]，p为所取长度。

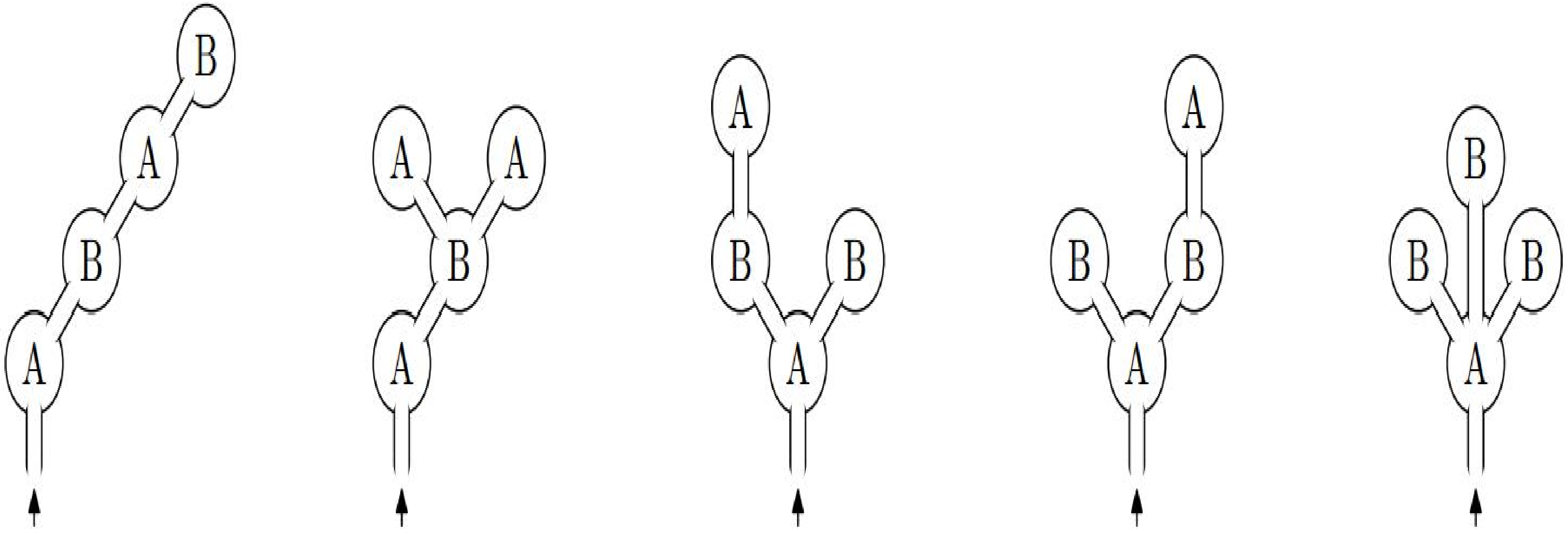
求价值最大！

n,l<=1k,a,b<=1e9

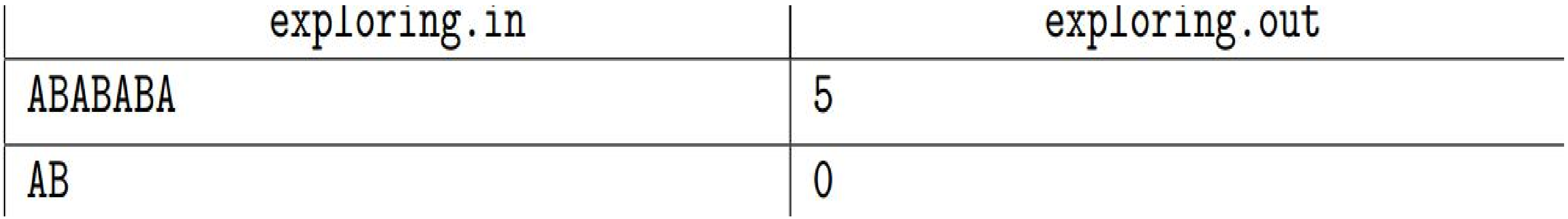
有n个产品，编号为1~n。要在m个机器人的手中生产完成。其中，第i个产品在第j个机器人手中的生产时间给出 。要把这些产品按照编号从 小到大生产.求生 产完所有产品的最短时间是多少。其中n<=5*10^5，m<=5。

区间

- http://codeforces.com/gym/101334（e）

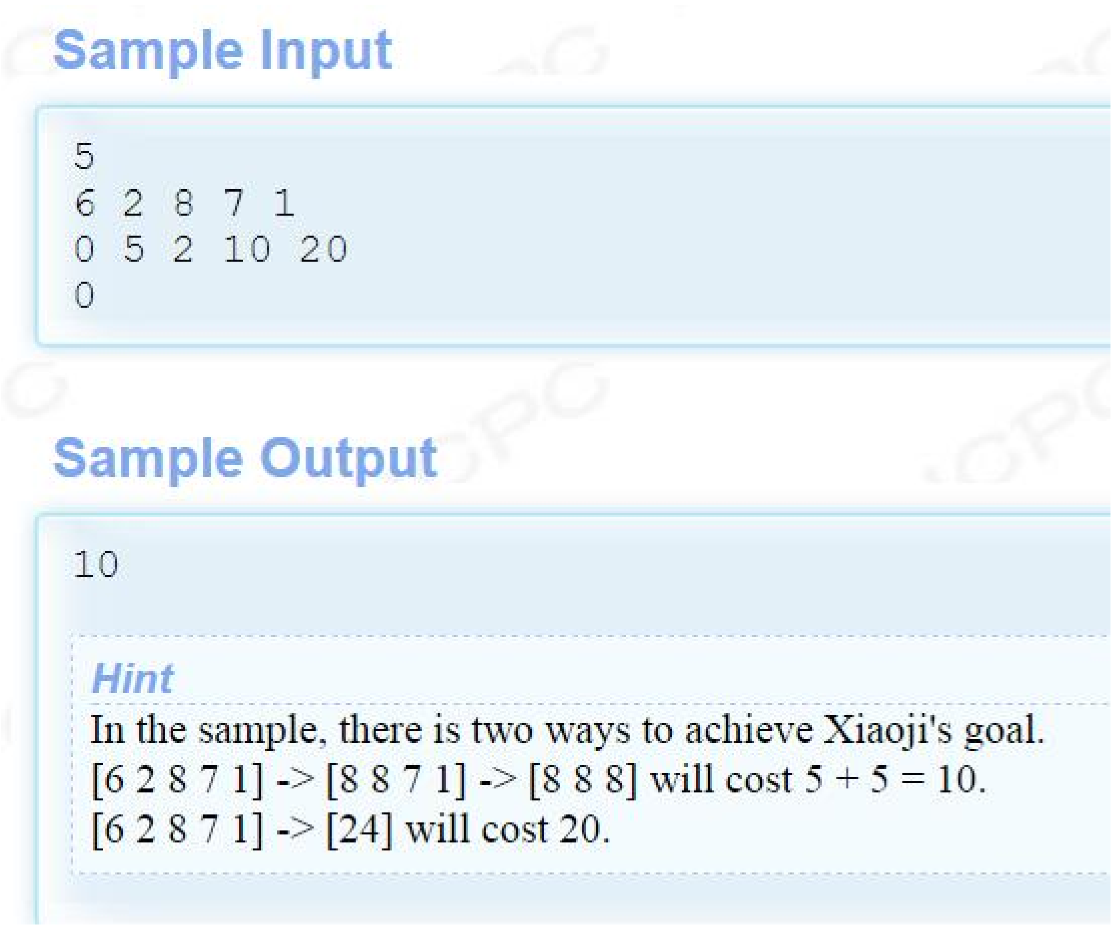


(a) (b) (c) (d) (e)



exploring.in	exploring.out
ABABABA	5
AB	0

划分



Sample Input

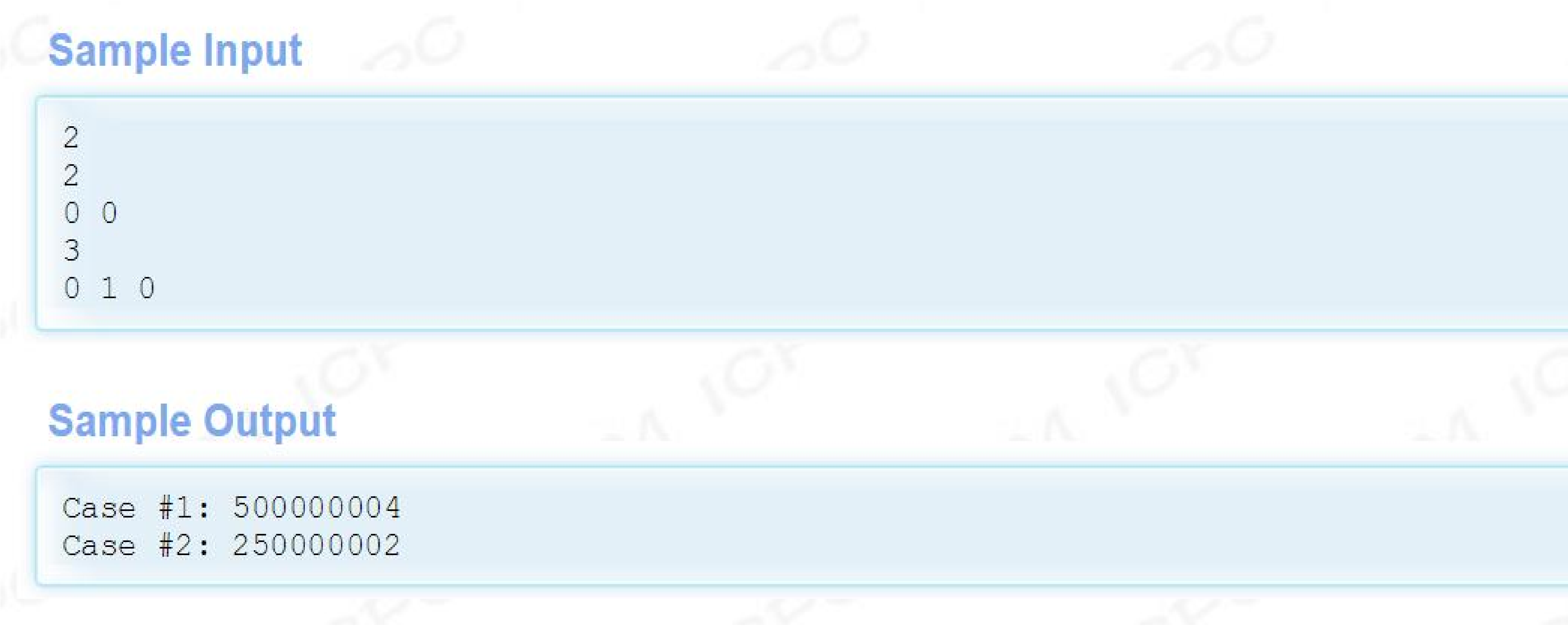
```
5
6 2 8 7 1
0 5 2 10 20
0
```

Sample Output

```
10
```

Hint

In the sample, there is two ways to achieve Xiaoji's goal.
[6 2 8 7 1] -> [8 8 7 1] -> [8 8 8] will cost 5 + 5 = 10.
[6 2 8 7 1] -> [24] will cost 20.

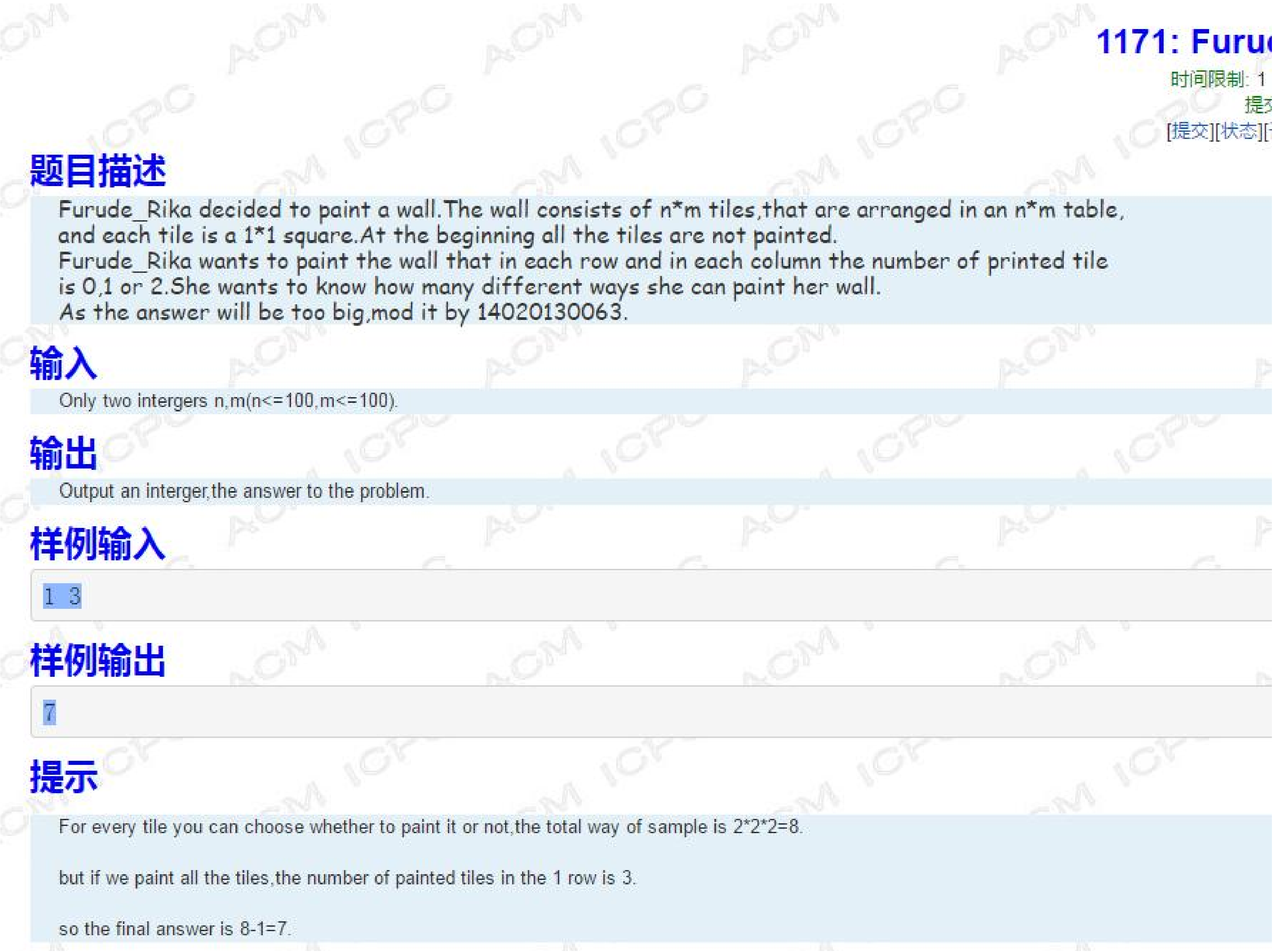


Sample Input

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

Sample Output

```
Case #1: 500000004
Case #2: 250000002
```



1171: Furu

时间限制: 1

提

[提交][状态][

题目描述

Furude_Rika decided to paint a wall.The wall consists of n*m tiles,that are arranged in an n*m table, and each tile is a 1*1 square.At the beginning all the tiles are not painted.

Furude_Rika wants to paint the wall that in each row and in each column the number of printed tile is 0,1 or 2.She wants to know how many different ways she can paint her wall.

As the answer will be too big,mod it by 14020130063.

输入

Only two intergers n,m(n<=100,m<=100).

输出

Output an interger,the answer to the problem.

样例输入

```
1 3
```

样例输出

```
7
```

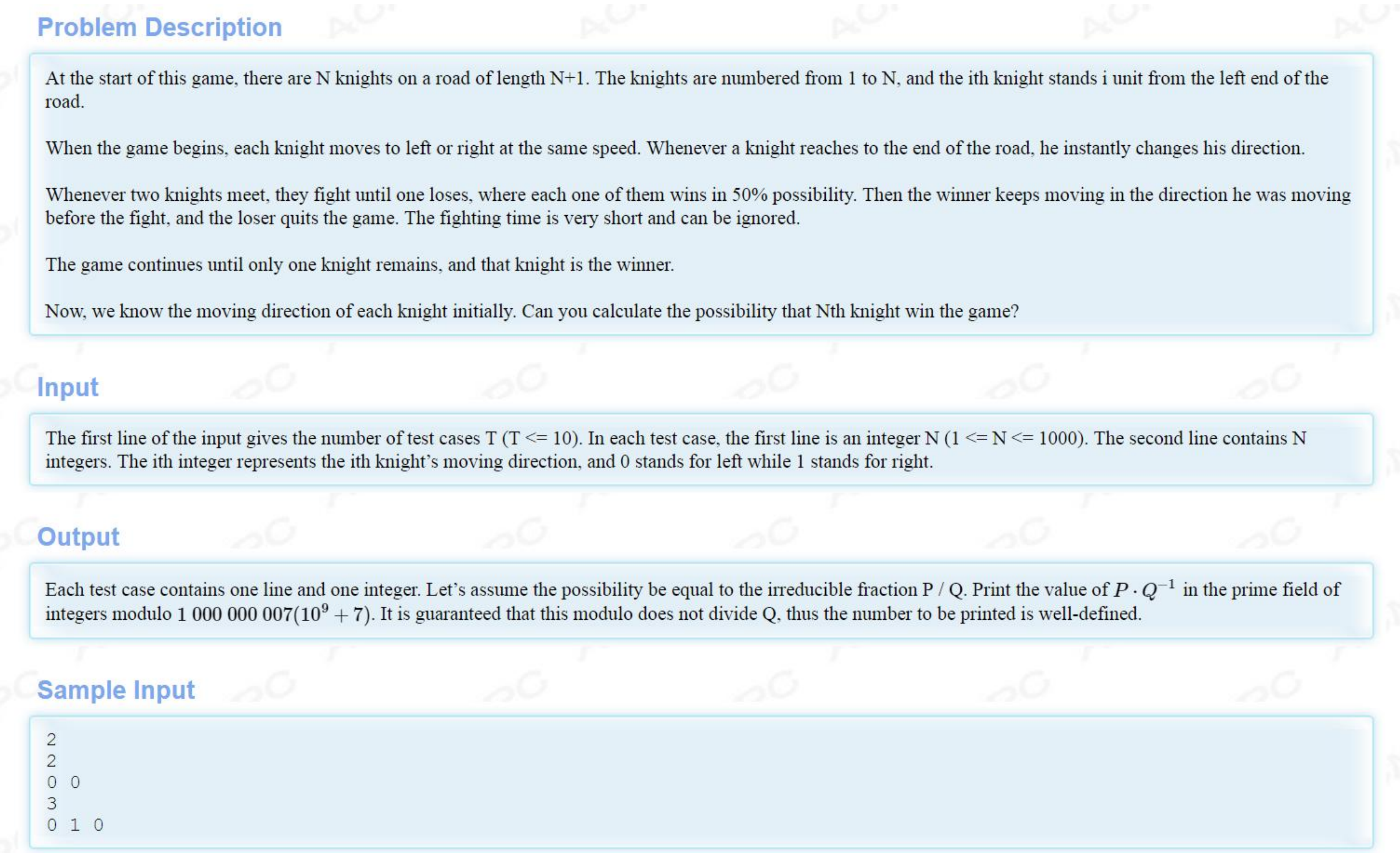
提示

For every tile you can choose whether to paint it or not,the total way of sample is 2*2*2=8.

but if we paint all the tiles,the number of painted tiles in the 1 row is 3.

so the final answer is 8-1=7.

概率



Problem Description

At the start of this game, there are N knights on a road of length N+1. The knights are numbered from 1 to N, and the ith knight stands i unit from the left end of the road.

When the game begins, each knight moves to left or right at the same speed. Whenever a knight reaches to the end of the road, he instantly changes his direction.

Whenever two knights meet, they fight until one loses, where each one of them wins in 50% possibility. Then the winner keeps moving in the direction he was moving before the fight, and the loser quits the game. The fighting time is very short and can be ignored.

The game continues until only one knight remains, and that knight is the winner.

Now, we know the moving direction of each knight initially. Can you calculate the possibility that Nth knight win the game?

Input

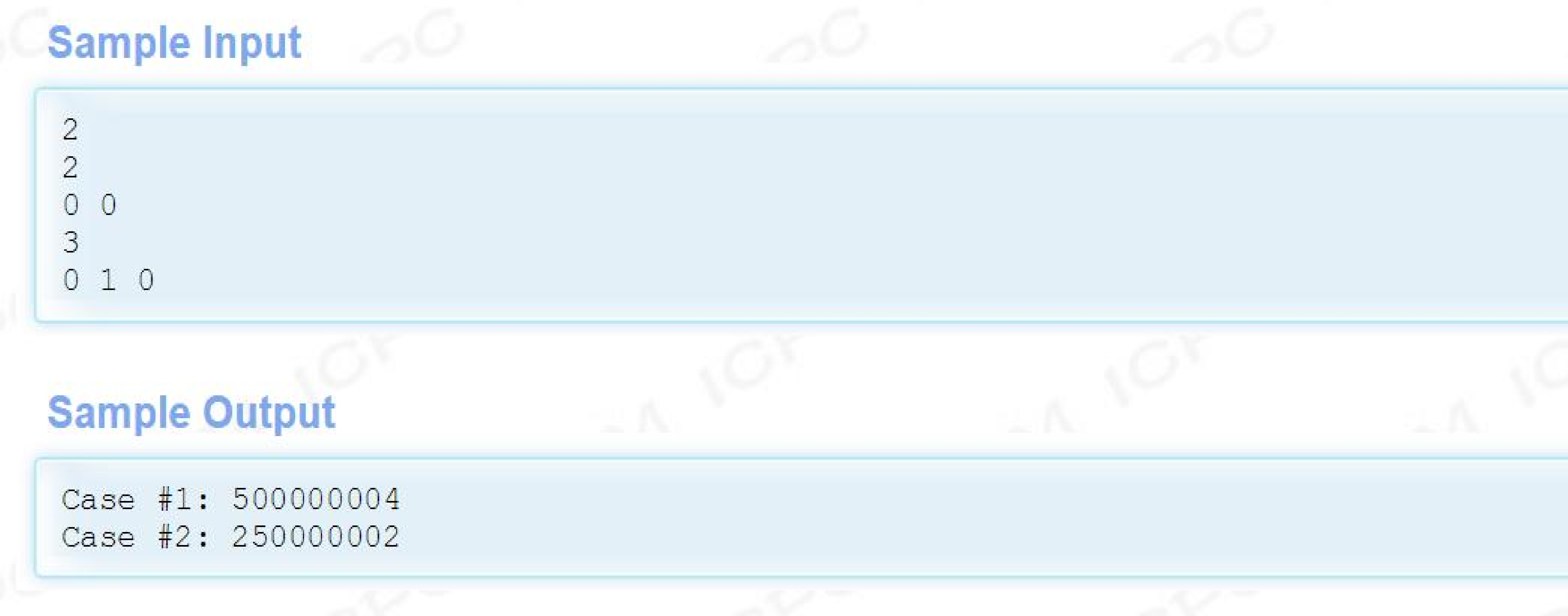
The first line of the input gives the number of test cases T (T <= 10). In each test case, the first line is an integer N (1 <= N <= 1000). The second line contains N integers. The ith integer represents the ith knight's moving direction, and 0 stands for left while 1 stands for right.

Output

Each test case contains one line and one integer. Let's assume the possibility be equal to the irreducible fraction P / Q. Print the value of $P \cdot Q^{-1}$ in the prime field of integers modulo 1 000 000 007$(10^9 + 7)$. It is guaranteed that this modulo does not divide Q, thus the number to be printed is well-defined.

Sample Input

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

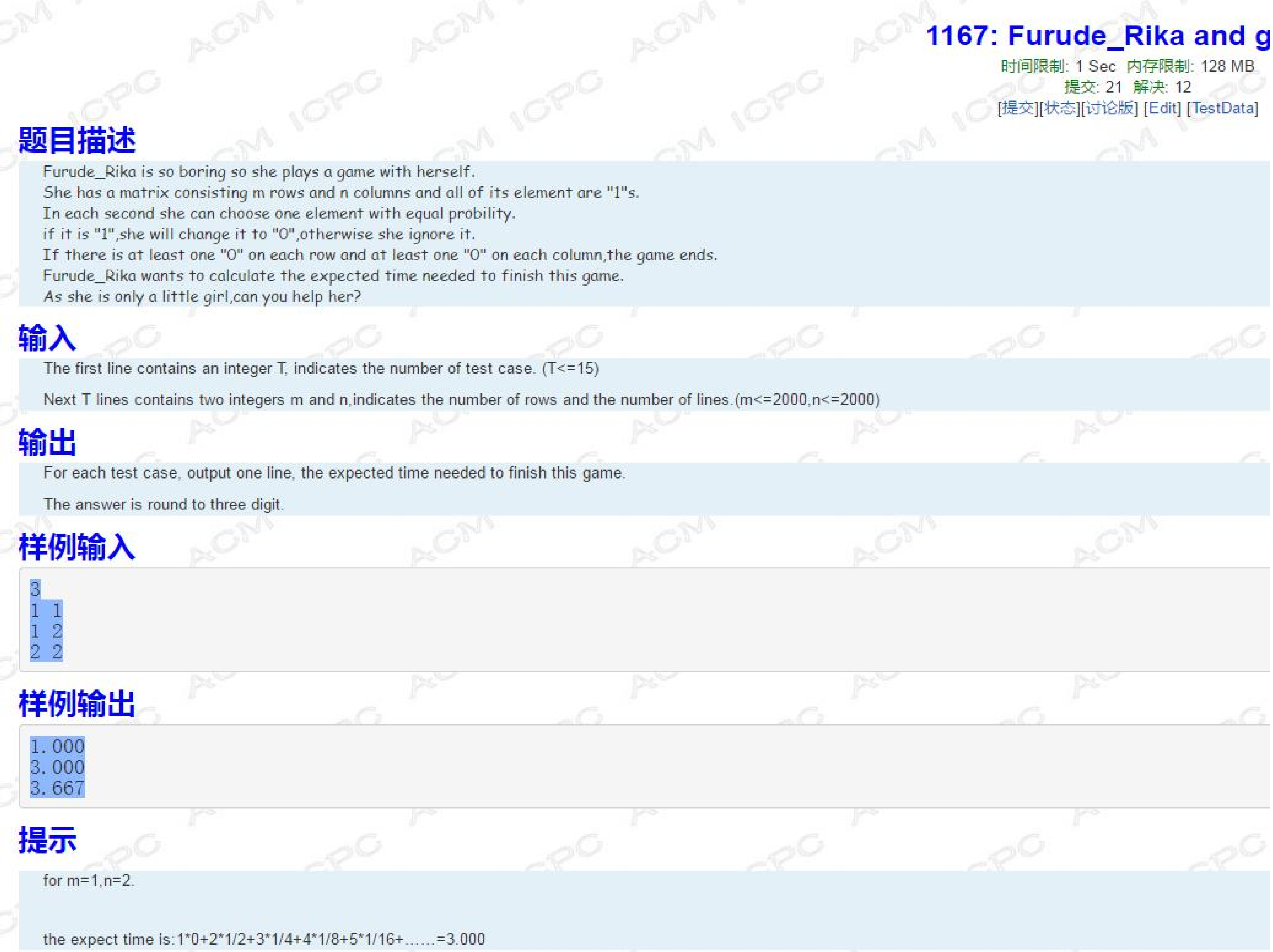


Sample Input

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

Sample Output

```
Case #1: 500000004
Case #2: 250000002
```



1167: Furude_Rika and g

时间限制: 1 Sec 内存限制: 128 MB
提交: 21 解决: 12
[提交][状态][讨论版] [Edit] [TestData]

题目描述

Furude_Rika is so boring so she plays a game with herself.
She has a matrix consisting m rows and n columns and all of its element are "1"s.
In each second she can choose one element with equal probility.
if it is "1",she will change it to "0",otherwise she ignore it.
If there is at least one "0" on each row and at least one "0" on each column,the game ends.
Furude_Rika wants to calculate the expected time needed to finish this game.
As she is only a little girl,can you help her?

输入

The first line contains an integer T, indicates the number of test case. (T<=15)

Next T lines contains two integers m and n,indicates the number of rows and the number of lines.(m<=2000,n<=2000)

输出

For each test case, output one line, the expected time needed to finish this game.

The answer is round to three digit.

样例输入

```
3
1 1
1 2
2 2
```

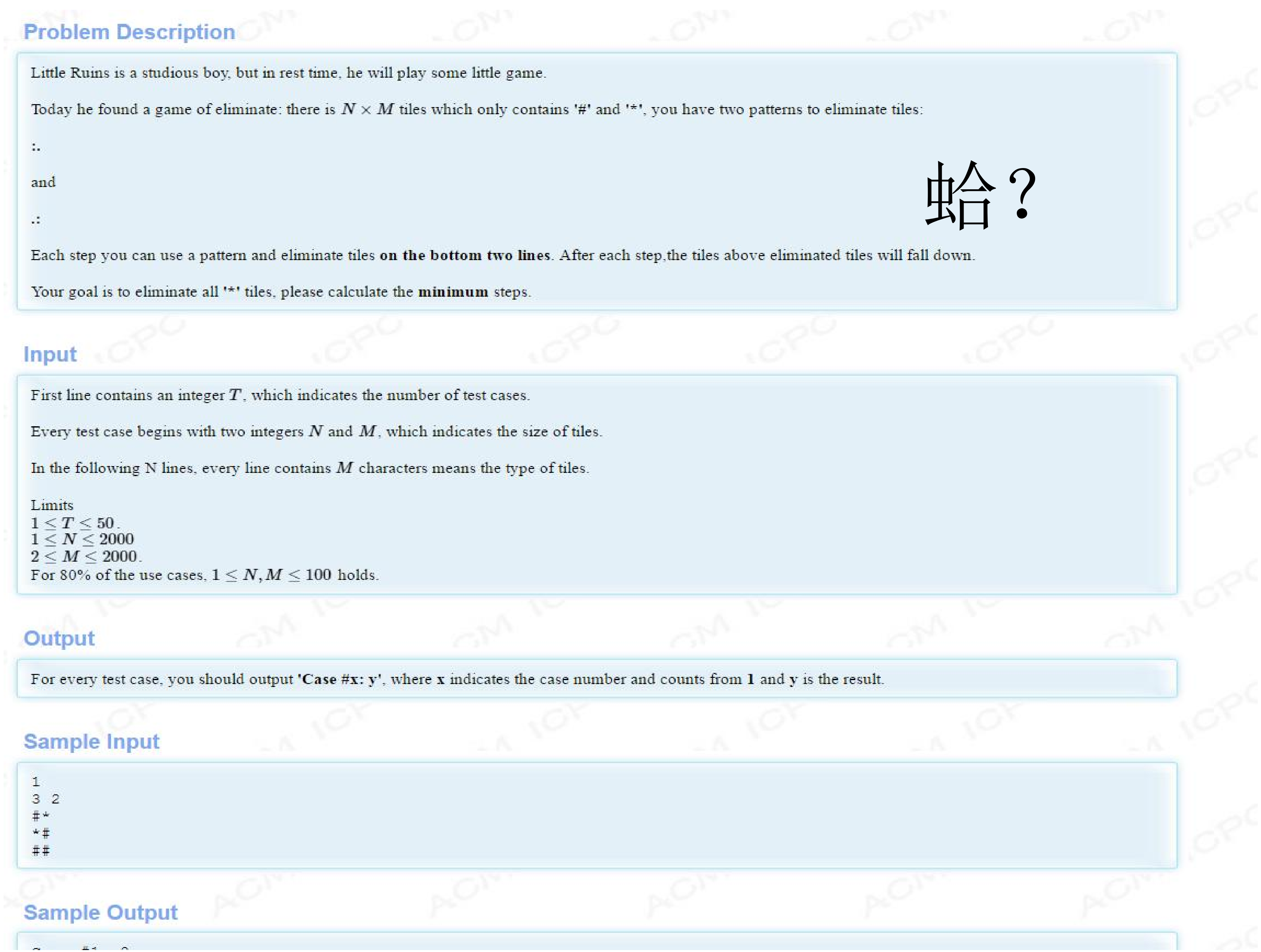
样例输出

```
1.000
3.000
3.667
```

提示

for m=1,n=2.

the expect time is:1*0+2*1/2+3*1/4+4*1/8+5*1/16+……=3.000



Problem Description

Little Ruins is a studious boy, but in rest time, he will play some little game.

Today he found a game of eliminate: there is $N \times M$ tiles which only contains '#' and '*', you have two patterns to eliminate tiles:

:.

and

.:

蛤？

Each step you can use a pattern and eliminate tiles **on the bottom two lines**. After each step,the tiles above eliminated tiles will fall down.

Your goal is to eliminate all '*' tiles, please calculate the **minimum** steps.

Input

First line contains an integer T, which indicates the number of test cases.

Every test case begins with two integers N and M, which indicates the size of tiles.

In the following N lines, every line contains M characters means the type of tiles.

Limits
$1 \leq T \leq 50$.
$1 \leq N \leq 2000$
$2 \leq M \leq 2000$.
For 80% of the use cases, $1 \leq N, M \leq 100$ holds.

Output

For every test case, you should output **'Case #x: y'**, where **x** indicates the case number and counts from **1** and **y** is the result.

Sample Input

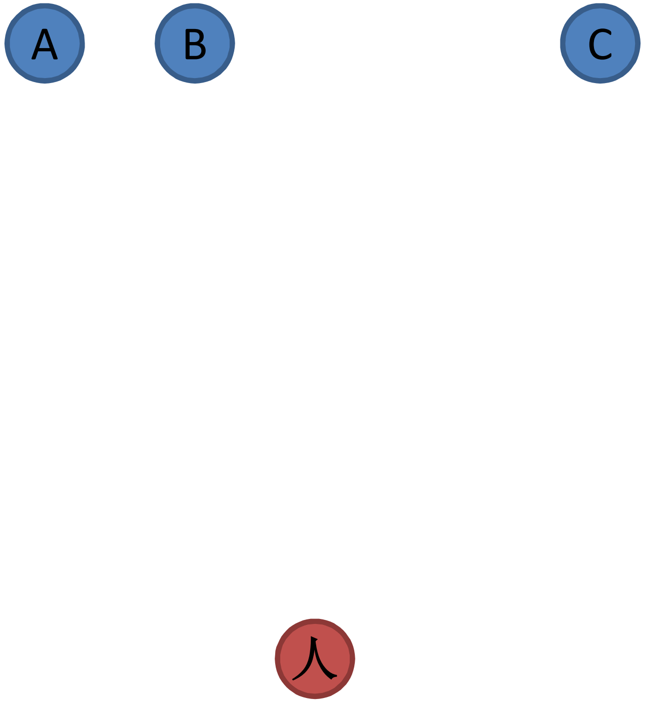
```
1
3 2
#*
*#
##
```

Sample Output

后效性

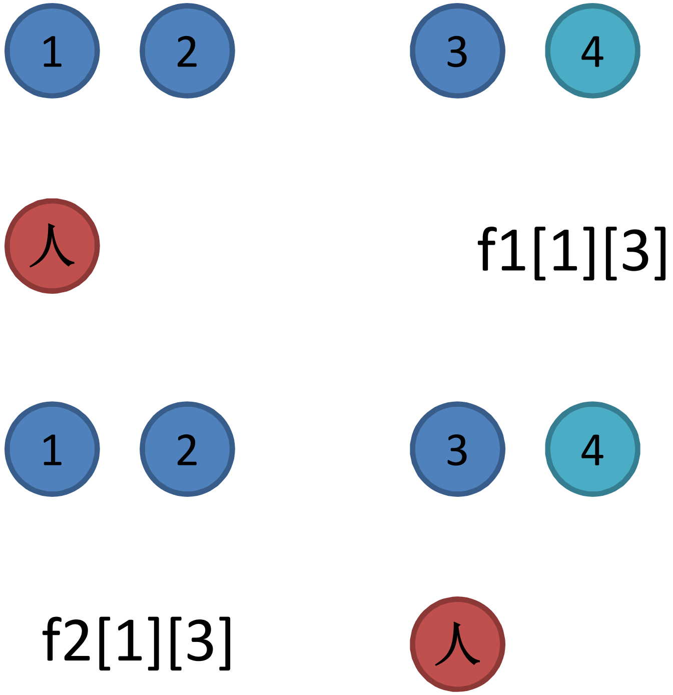
问题n

- 有n个彩蛋，分别位于(x_i,y_i)，以v_i的速度匀速下落。
- 你从坐标x出发，速度为1，每次可以向左或向右走到一个未被射落的彩蛋，将其射落。得分为被射彩蛋y坐标的千分之一。
- 你的目标是射落所有彩蛋并使得分最高。



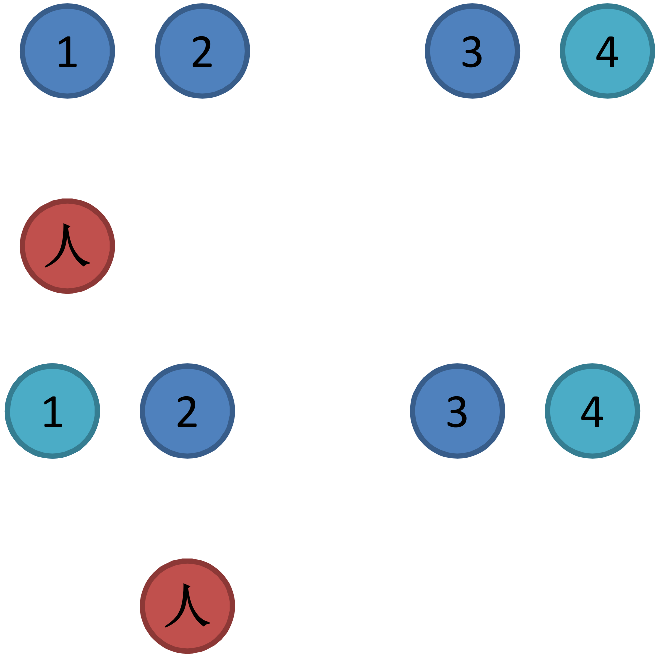
问题n

- 已射的彩蛋集合是不断增大的。
- 状态？



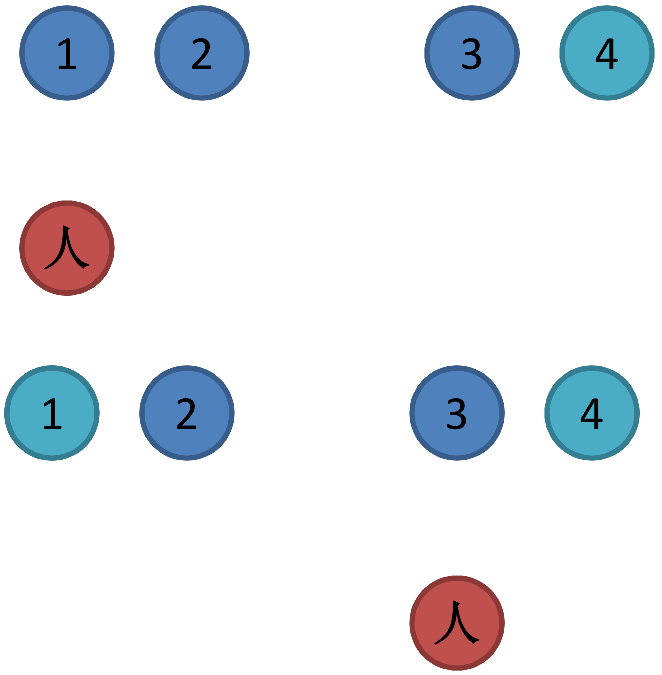
问题n

- 考虑f1[1][3]，当前处于位置1。
- 可以由f1[2][3]沿着2->1走来。再射落1号彩蛋。

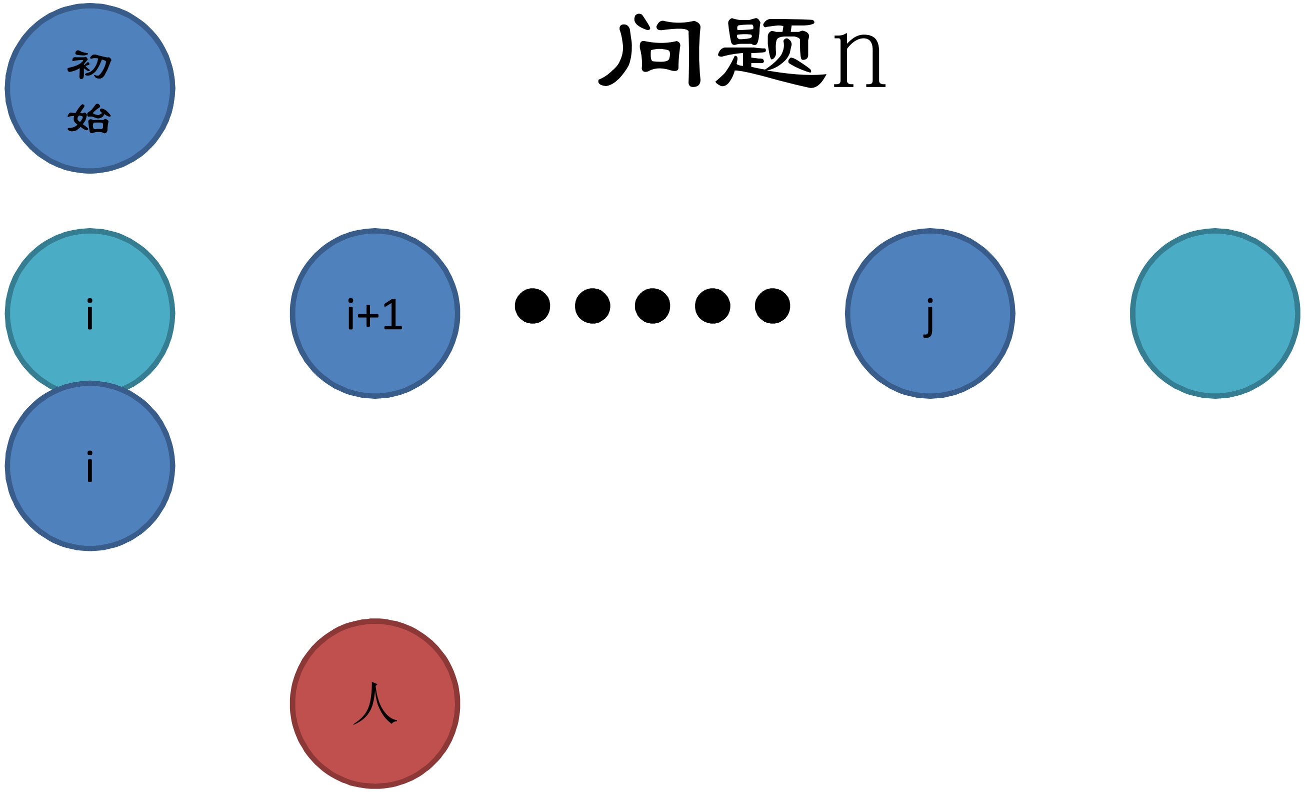


问题n

- 考虑f1[1][3]，当前处于位置1。
- 可以由f1[2][3]沿着2->1走来。再射落1号彩蛋。
- 可以由f2[2][3]沿着3->1走来。再射落1号彩蛋。



问题n



- 将-t*v_i在射落i之前计算。
- 每次移动都要把未来会减少的得分计算在内。
- 射击i时再加上$y_i/1000$。

问题n

- 用w[i][j]表示除了i到j这段彩蛋的下落速度和。
- 从i+1走到i，f1[i][j]=f1[i+1][j]-(x_{i+1}-x_i)*w[i+1][j]
- 从j走到i，f1[i][j]=f2[i+1][j]-(xj-x_i)*w[i+1][j]
- f1[i][j]+=y_i/1000。
- f2[i][j]的方程类似。
- 答案就是max(f1[1][n]，f2[1][n])。
- 时间复杂度$0(n^2)$。

小结 sdoi (bzoj2037)

- 当前射击的费用受到之前决策的影响。
- 如果新增状态t表示过去决策的影响，状态数将会无法承受。
- 改变“时间观”，从过去考虑当前，即从当前考虑未来，把当前决策对未来的影响算作当前决策费用，计算到当前状态。

当前决策对未来“行动”费用的影响只与当前决策有关。

- 有一层高度为n(<=4000)的楼，每层楼可以建一个乒乓球室或者一个台球室，每层楼都有想玩这两种球的人，他们会到最近的台球室或者乒乓球室，问所有人的移动距离的最小值之和