说明

建议阅读PDF版本的题面

前置知识

在开启本题之前，让我们先来了解一个名字为“**快速幂**”的算法：

快速幂算法是一种高效计算幂的算法，通常用于在$O(\log n)$时间内计算 a^b（其中 a 是底数，b 是指数）的值。其核心思想是利用指数的二进制表示来减少乘法次数，例如$b = 13$可以表示为1101_2（即$1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$）,$13 = 8 + 4 + 1$, 那么$a^{13}$可以写成$a^8 \times a^4 \times a^1$ 。

我们利用循环“遍历”$\{a^1, a^2, a^4, a^8, ...\}$，当对应的二进制表示为1时，将对应的2的整数次幂乘到答案ans中，参考的C++代码如下：

```
int fastPow(int x, int p) {    // 求x的p次方
      int ans = 1;
      while (p) {                      // 倒着遍历二进制表示的每一位
            if (p & 1) {
                  ans = ans * x;
            }
            x = x * x;
            p >>= 1;
      }
      return ans;
}
```

> 实际上，上面的代码很可能会超出 `int` 类型的数据范围

题目描述

现在，你需要求解一个$n \times n$矩阵的幂A^p，A^p为n个A做**矩阵乘法**的结果。为了避免答案过大，矩阵中的每一个值都要对$M = 10^9 + 7$取余数。

为了你的程序效率，你需要仿照上面的快速幂代码，实现**矩阵快速幂**。

输入格式

第一行两个整数n, p

接下来n行，每行n个整数，表示矩阵A

输出格式

输出共n行，每行n个整数，表示A^p中每项对M取余数的结果

输入样例

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

输出样例

```
3 2
2 1
```

题目提示

- 为了避免矩阵计算在程序中多次出现，建议使用**结构体**或**类**对矩阵`Matrix`进行**封装**。
- 在C++中，你可以**重载运算符**。也就是说，你可以定义你自己`Matrix`类的`*`运算符。当然，这在本次作业中不是必须的~~（因为可能还没教到）~~。你也可以定义一个`mul`函数来计算矩阵乘法并获得所有的分数。
- 一些基础的数学知识：
    - 带取余的矩阵乘法：$C[i][j] = \sum_{k=1}^{n} A[i][k] \times B[k][j] \mod M$
- $(a+b)\%p = (a\%p + b\%p)\%p$， 其中%表示取余数
    - $(a \times b)\%p = (a\%p \times b\%p)\%p$， 其中%表示取余数
    - 你也许需要**单位矩阵**这个概念的帮助，这里不再赘述。
- 注意选择合适的数据类型

数据范围与约定

- 对于50%的数据，$n \le 4, p \le 1000$，这意味着你不使用快速幂算法也可以获得这一部分的分数
- 对于100%的数据，$n \le 5, p \le 10^9, 0 \le A[i][j] \le 10^9$