“Citadel ｜ Citadel Securities 杯”

THUPC 2019

清华大学学生程序设计竞赛暨高校邀请赛

目 录

A. 不用找的树 / Tree

时间限制： 10.0 秒

空间限制： 512 MB

【题目描述】

给出一棵 n 个节点的树，结点标号从 1 到 n。

定义树上两点 a,b 间的距离 $d(a,b)$ 是最小的非负整数 k，满足存在结点序列 $v_0,v_1,\ldots,v_k$，满足 $v_0=a$，$v_k=b$，且对于 $0\le i\le k-1$ 有 v_i 和 v_{i+1} 之间在树上有一条边相连。

有 m 个询问，每个询问包含参数 p_0,d_0,p_1,d_1，求：

$$\sum_{d(p_0,a)\le d_0}\sum_{d(p_1,b)\le d_1} d(a,b)$$

【输入格式】

从标准输入读入数据。

第一行一个整数 n，表示树的节点数目。

接下来一行 $n-1$ 个整数 $f_2,f_3,\ldots,f_n$，依次表示 i 和 f_i（$1\le f_i\le i-1$）之间有一条边。

接下来一行一个整数 m，表示询问数目。

接下来 m 行依次描述所有询问：每行四个整数 p_0,d_0,p_1,d_1（$1\le p_0,p_1\le n$，$0\le d_0,d_1\le n-1$）描述一组询问。

保证 $1\le n\le 10^5$，$1\le m\le 10^5$。

【输出格式】

输出到标准输出。

共 m 行，依次回答各组询问：每行输出一行一个整数表示这组询问的答案。

【样例输入】

```
7
1 1 2 3 5 2
5
5 1 5 0
2 0 5 0
2 2 4 5
```

```
7 2 2 4
3 2 5 4
```

【样例输出】

```
2
3
69
57
70
```

B. 组合数据结构问题 / Problem

时间限制： 1.0 秒

空间限制： 512 MB

【题目背景】

众所周知，小葱同学擅长计算，尤其擅长计算组合数，但这个题和组合数没什么关系。

【题目描述】

小葱同学在学习了组合数的计算之后，开始了研究数据结构的道路。通过十五分钟的刻苦学习，小葱同学初步掌握了队列、栈和堆这三种数据结构。小葱同学发现这三种数据结构都支持两种操作：

1. 将某个数插入该数据结构。
2. 从该数据结构中按照数据结构的原理取出一个数。

小葱同学为了检验自己对这三种数据结构的理解，设计了一个类似的黑箱模型。该模型也支持两种操作，向黑箱中输入一个数或者从黑箱中输出一个数。现在小葱对该黑箱做了若干次操作，并给出每次输入和输出的数，问这个黑箱实现的是否可能是队列、栈、大根堆或者小根堆。

虽然小葱同学对这四种数据结构了如指掌，但他还是决定告诉你它们的分别是什么：

- 如果黑箱是**队列**：黑箱初始为空，输入时数将被放入黑箱，输出时当前黑箱内**最早被放入的**数将被输出并移出黑箱。
- 如果黑箱是**栈**：黑箱初始为空，输入时数将被放入黑箱，输出时当前黑箱内**最晚被放入的**数将被输出并移出黑箱。
- 如果黑箱是**大根堆**：黑箱初始为空，输入时数将被放入黑箱，输出时当前黑箱内**值最大的**数将被输出并移出黑箱。特别地，如值最大的数有多个，则仅将其中一个移出黑箱。
- 如果黑箱是**小根堆**：黑箱初始为空，输入时数将被放入黑箱，输出时当前黑箱内**值最小的**数将被输出并移出黑箱。特别地，如值最小的数有多个，则仅将其中一个移出黑箱。

【输入格式】

从标准输入读入数据。

第一行一个整数 N 代表操作的个数。

接下来 N 行每行两个整数 opt, v。如果 $opt = 1$，代表这次操作小葱同学向黑箱中插入了 v 这个数；如果 $opt = 2$，代表小葱这次操作从黑箱中取出了 v 这个数。

保证 $0 \le N \le 10^5$，$-2^{31} \le v < 2^{31}$，$opt \in \{1, 2\}$。

注意输入的数据仅保证在格式和范围上完全正确，不保证任何其他条件。

【输出格式】

输出到标准输出。

共四行，每行可能是 `Yes` 或者 `No`，分别依次代表该黑箱是否可能是队列、栈、大根堆或者小根堆。

【样例 1 输入】

```
6
1 1
1 2
1 3
2 1
2 2
2 3
```

【样例 1 输出】

```
Yes
No
No
Yes
```

C. 过河卒二 / Chess

时间限制： 1.0 秒

空间限制： 512 MB

【题目背景】

首先我们回忆一下经典难题过河卒问题：

棋盘上 A 点有一个过河卒，需要走到目标 B 点。卒行走的规则：可以向上、或者向右。同时在棋盘上 C 点有一个对方的马，该马所在的点和所有跳跃一步可达的点称为对方马的控制点，因此称之为“马拦过河卒”。

棋盘用坐标表示，A 点 $(1,1)$、B 点 (N,M)，同样马的位置坐标是需要给出的。

现在要求你计算出卒从 A 点能够到达 B 点的路径的条数，假设马的位置是固定不动的，并不是卒走一步马走一步。

请注意，上述背景内容与本题无关！

【题目描述】

Kiana 喜欢玩象棋，尤其是喜欢用象棋玩过河卒的游戏。在传统的过河卒问题中，Kiana 需要控制一个卒从起点走到终点，在路中避开一个对方的马的攻击，然后假装不会算并询问你从起点到终点的路径总数。

在今天的过河卒二游戏中，Kiana 还是控制一个卒在一个 $N \times M$ 的棋盘上移动，初始时卒位于左下方坐标为 $(1,1)$ 位置，但为了增加难度，Kiana 对游戏规则做出了一些修改。传统的过河卒每步只能向上或向右移动 1 格，Kiana 规定自己的过河卒二还可以在一步中向右上方移动 1 格，即如果当前卒位于坐标 (x,y) 处，则下一步可以走到 $(x+1,y)$、$(x,y+1)$ 或 $(x+1,y+1)$ 中的任意一格里面去，同时 Kiana 认为，如果两种移动方案在某一步时卒移动的方向（右、上或右上）不同，则两种方案就是不同的，例如从 $(1,1)$ 先走到 $(1,2)$ 再走到 $(2,2)$、从 $(1,1)$ 先走到 $(2,1)$ 再走到 $(2,2)$ 和从 $(1,1)$ 直接走到 $(2,2)$ 是三种不同的移动方案。

其次，过河卒二的终点不再是一个特定的位置，Kiana 规定卒可以从棋盘的上方或右方走出棋盘，此时就视为游戏成功。注意在走出棋盘时仍然有方向选择的不同，例如若过河卒位于 $(1,M)$ 处，则下一步它可以向右或者向右上用两种方式走出棋盘，若过河卒位于 (N,M) 处，则下一步它可以向上、向右或者向右上用三种方式走出棋盘，以不同的方式走出棋盘仍然被算作是不同的移动方案。

此外，对方马的攻击范围不再是有规律的几个位置，而是 Kiana 规定好的 K 个特定坐标，并要求过河卒在移动的过程中不能走到这 K 个坐标的任何一个上，在除这些坐标以外的位置上过河卒都可以按规则自由移动。

现在 Kiana 想知道，过河卒二有多少种不同的移动方案可以走出棋盘，这个答案可能非常大，她只想知道方案数对 59393 取模后的结果。由于她不会算，所以希望由

你来告诉她。

【输入格式】

从标准输入读入数据。

第一行包含三个整数 N、M 和 K，分别表示棋盘的坐标范围与对方马的攻击格子数（即 Kiana 规定的不能经过的坐标数）。

接下来 K 行，第 i 行包含两个正整数 X_i 和 Y_i，表示对方马的第 i 个攻击坐标为 (X_i, Y_i)。

对于所有数据，保证 $1 \le N \le 10^9, 1 \le M \le 10^5, 0 \le K \le 20, 1 \le X_i \le N, 1 \le Y_i \le M$，$(1, 1)$ 一定不会被对方马攻击，且被攻击的格子中不存在两个坐标相同的格子。

【输出格式】

输出到标准输出。

输出一行一个整数，表示过河卒走出棋盘的方案数对 59393 取模后的结果。

【样例 1 输入】

```
3 3 1
2 2
```

【样例 1 输出】

```
24
```

【样例 1 解释】

用 ↑ 表示过河卒向上移动了一格，用 → 表示过河卒向右移动了一格，用 ↗ 表示过河卒向右上移动了一格，由此可以简化样例解释的表述。

24 种移动方案如下：(↑↑↑)、(↑↑↗)、(↑↑→↑)、(↑↑→↗)、(↑↑→→↑)、(↑↑→→↗)、(↑↑→→→)、(↑↗↑)、(↑↗↗)、(↑↗→↑)、(↑↗→↗)、(↑↗→→)、(→→→)、(→→↗)、(→→↑→)、(→→↑↗)、(→→↑↑→)、(→→↑↑↗)、(→→↑↑↑)、(→↗→)、(→↗↗)、(→↗↑→)、(→↗↑↗)、(→↗↑↑)。

D. 鸽鸽的分割 / Divide

时间限制： 1.0 秒

空间限制： 512 MB

【题目描述】

牛牛有一块蛋糕，他想把蛋糕分给小朋友们。蛋糕一开始是圆形的，牛牛会在圆周上选择 n 个不重合的点，将这几个点两两用线段连接。这些线段将会把蛋糕分成若干块。

现在，牛牛想知道，蛋糕最多会被分成多少块，请你告诉他答案。

【输入格式】

从标准输入读入数据。

输入包含至多 20 行，每行一个整数 n，含义见【题目描述】。保证 $0 \le n \le 64$。

【输出格式】

输出到标准输出。

依次回答牛牛的每个问题，对于每个问题，输出一行，包含一个整数表示答案。

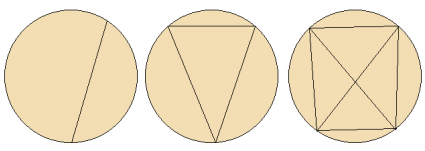
【样例 1 输入】

```
2
3
4
```

【样例 1 输出】

```
2
4
8
```

【样例 1 解释】



E. 能量波 / Energywave

时间限制： 15.0 秒

空间限制： 1024 MB

【题目背景】

有两个超级英雄在宇宙中战斗。英雄 A 为了击败对手英雄 B，使出了他的绝招，发出了一记能量光波。

英雄 B 有一些体力值，如果英雄 A 的能量光波对英雄 B 的伤害足够大，英雄 B 就会被击倒。否则英雄 A 就会因为耗尽能量而被英雄 B 抓住破绽，从而反而被英雄 B 击倒。所以英雄 A 迫切想知道自己的能量光波到底能够对英雄 B 造成多少伤害。

【题目描述】

为了简化问题，英雄 B 可以被描述为空间中的多个凸多面体拼成的物体。这些凸多面体可能有重合的部分，重合部分作为英雄 B 的身体只会被计算一次。

英雄 A 发出的能量光波也可以被描述为空间中的另外一个凸多面体，这个光波在空间中每秒均匀移动一个向量 $\vec{v} = (v_x, v_y, v_z)$. 能量光波可以穿过任何物体，并且在穿越的过程中对对方造成伤害。

在时刻 t, 假设英雄 A 的能量光波和英雄 B 的身体的交的体积为 $f(t) = V$ ，那么在这一瞬间，能量光波造成的瞬时伤害速率就恰好是 V。而所有时刻的总计伤害就可以被表示为

$$\int_0^\infty f(t)dt.$$

英雄 A 想要知道自己的能量光波对英雄 B 的身体造成了多大的总计伤害。

【输入格式】

从标准输入读入数据。

第 1 行一个正整数 m_B，表示英雄 B 的身体有多少个凸多面体组成。

接下来有 m_B 个输入块，每块表示英雄 B 的身体的一个组成部分。

每个输入块开头第一行为一个正整数 n，表示这个凸多面体的顶点的个数，

接下来 n 行每行一个三元组，其中第 i 个 (x_i, y_i, z_i) 表示第 i 个点的坐标。

接下来一行一个正整数 n_A，表示英雄 A 的能量光波的对应凸多面体的顶点数，

接下来 n_A 行每行一个三元组，其中第 i 个 (x_i, y_i, z_i) 表示第 i 个点的坐标。

再接下来一行一个三元组 (v_x, v_y, v_z)，表示英雄 A 的能量光波的移动速度。

我们保证 $1 \le m_B \le 4$, $4 \le n, n_A \le 8$ ，所有输入的点的坐标都是 $[-100, 100]$ 内的整数。并且所有速度向量的分量都是 $[-10, 10]$ 内的实数。所有凸多面体都是不退化的（意思是这个凸多面体的体积非 0）。

输入中可能会出现四点共面或者三点共线的情况。

我们保证在第 0 秒能量光波和英雄 B 是不相交的。并且在第 10^4 秒之后交集的体积一直是 0。

【输出格式】

输出到标准输出。

一行输出一个实数，表示总计伤害的值。

输出与标准答案的绝对误差或相对误差小于 10^{-6} 就会被算作正确。

【样例输入】

```
1
8
0 0 0
0 0 1
0 1 0
0 1 1
1 0 0
1 0 1
1 1 0
1 1 1
8
2 0 0
2 0 1
2 1 0
2 1 1
3 0 0
3 0 1
3 1 0
3 1 1
-1 0 0
```

【样例输出】

```
1.0000000000
```

F. 摆家具 / Furniture

时间限制： 8.0 秒

空间限制： 512 MB

【题目描述】

你有 k 件不同的家具，需要摆放在 n 个不同的房间中。假设每个房间足够大，并且只考虑每件家具处于哪个房间（而不考虑房间内部如何摆放），那么总共有 n^k 种不同的摆放方式（注意，不是 k^n）。

摆放家具也算一门学问了，至少不太好乱摆的吧？对于每种摆放方式，我们可以给这种方式打分。例如，某个方案把餐桌放到了卫生间，或是一个卧室放了两张床而另一个卧室没有床，就会获得比较低的分数。由于这个分数关于每件家具、每个房间不是独立的，我们会输入所有 n^k 种摆放方式的分数。

你现在心血来潮，想换一换房间的布局。给出一种初始时的摆放方式，你会重复 T 次下述操作：每次，你会任选一件家具，然后将这件家具移动到任意一个其他房间中。每一轮有 $k(n-1)$ 种决策（选择家具的方案数乘以选择另一个房间的方案数），所以总共有 $k^T(n-1)^T$ 种决策。你需要计算这每一种决策后的摆放方式的得分之和。

不仅如此，我们会给出 q 次询问，每次输入初始时的摆放方式与 T，你需要**在线地**回答 $k^T(n-1)^T$ 种决策后的得分之和（取模）。详见输入与输出格式。

我们如下定义一种摆放方式的编号：

我们将家具用 0 到 $k-1$ 的不同整数编号，房间用 0 到 $n-1$ 的不同整数编号。设在某种摆放方式下，第 i 号家具被放在了 p_i 号房间中，则定义这种摆放方式的编号为 $\sum_{i=0}^{k-1} p_i n^i$。可以发现，所有的 n^k 种摆放方式的编号恰好是 0 到 n^k-1 的不同整数。

另外，设 $P = 998244353$。

【输入格式】

从标准输入读入数据。

第一行输入三个正整数 n, k, q。

接下来 n^k 行，每行输入一个小于 P 的正整数，依次表示编号为 $0, 1, \ldots, n^k-1$ 的摆放方式的得分。

接下来 q 行，每行输入两个非负整数。设某行的输入为 a, b（保证 $0 \le a < n^k, 0 \le b < P$），则此次询问的初始摆放方式的编号为 a，而 $T = b \cdot r \bmod P$，其中 r 是你上一个输出的数（对于第一次询问为 1）。

同一行内输入的相邻两个数之间以一个空格隔开。

保证 $n \ge 2$，$k \ge 1$；$n^k \le 10^6$；$q \le 5 \times 10^5$

【输出格式】

输出到标准输出。

对于每次询问输出一行，包含一个非负整数，表示该询问的得分之和对 P 取模的结果。

【样例 1 输入】

```
2 3 3
1
10
100
1000
998244245
100000
1000000
10000000
0 1
0 1
1 233
```

【样例 1 输出】

```
2
2202003
444957911
```

【样例 1 解释】

第一次询问中，初始摆放方式的编号为 0，$T = 1$。

初始时，0 号家具放在 0 号房间，1 号家具放在 0 号房间，2 号家具放在 0 号房间。经过 1 次操作后，可能的情况有：

- 将 0 号家具移动到 1 号房间，此后的摆放方式编号为 1，得分为 10；
- 将 1 号家具移动到 1 号房间，此后的摆放方式编号为 2，得分为 100；
- 将 2 号家具移动到 1 号房间，此后的摆放方式编号为 4，得分为 998244245。

所以，所有情况的总得分为 998244355，对 P 取模后为 2。

第二次询问中，初始摆放方式的编号为 0，$T = 2$。

初始时，0 号家具放在 0 号房间，1 号家具放在 0 号房间，2 号家具放在 0 号房间。经过 2 次操作后，可能的情况有：

- 将 0 号家具移动到 1 号房间，然后将 0 号家具移动到 0 号房间，此后的摆放方式编号为 0，得分为 1；
- 将 0 号家具移动到 1 号房间，然后将 1 号家具移动到 1 号房间，此后的摆放方式编号为 3，得分为 1000；
- 将 0 号家具移动到 1 号房间，然后将 2 号家具移动到 1 号房间，此后的摆放方式编号为 5，得分为 100000；
- 将 1 号家具移动到 1 号房间，然后将 0 号家具移动到 1 号房间，此后的摆放方式编号为 3，得分为 1000；
- 将 1 号家具移动到 1 号房间，然后将 1 号家具移动到 0 号房间，此后的摆放方式编号为 0，得分为 1；
- 将 1 号家具移动到 1 号房间，然后将 2 号家具移动到 1 号房间，此后的摆放方式编号为 6，得分为 1000000；
- 将 2 号家具移动到 1 号房间，然后将 0 号家具移动到 1 号房间，此后的摆放方式编号为 5，得分为 100000；
- 将 2 号家具移动到 1 号房间，然后将 1 号家具移动到 1 号房间，此后的摆放方式编号为 6，得分为 1000000；
- 将 2 号家具移动到 1 号房间，然后将 2 号家具移动到 0 号房间，此后的摆放方式编号为 0，得分为 1。

所以，所有情况的总得分为 2202003，对 P 取模后为 2202003。

第三次询问中，初始摆放方式的编号为 1，$T = 513066699$。初始时，0 号家具放在 1 号房间，1 号家具放在 0 号房间，2 号家具放在 0 号房间。

……（省略至少 $3^{513066699}$ 行）

G. 令人难以忘记的题目名称 / Game

时间限制： 3.0 秒

空间限制： 512 MiB

【题目描述】

现在有一个长度为 N 的整数序列 S（下标从 0 开始），Alice 和 Bob 在这个序列上博弈。

游戏按轮进行，每一轮中：

- Alice 给出一个长度为 N 的正整数序列 T
- Bob 看到 Alice 给出的 T，然后选择 $[0, N-1]$ 里的一个整数 x
- 之后我们把 S 转化为 S'，规则如下：

$$S'_i = S_i + T_{(i+x) \bmod N}$$

- 以 S' 作为新的 S，结束这一轮。

如果某一轮结束后，S 中每个数都是一个给定质数 P 的倍数，那么 Alice 胜利。

给定 N 和初始序列 S，请问：Alice 是否能在有限步必胜，如果答案为是，最快可以在几轮内保证胜利。

【输入格式】

从标准输入读入数据。

第一行两个非负整数 N, P，保证 P 是一个质数。

接下来一行 N 个空格隔开的整数，描述初始序列 S（$0 \le S_i \le 10^9$）。

保证 $N \le 3 \times 10^5$，$P \le 200$。

【输出格式】

输出到标准输出。

输出一个整数，如果 Alice 不能在有限步必胜，输出 -1，否则输出一个整数 x 表示 Alice 最快能在几轮内胜利。

【样例 1 输入】

```
4 2
0 1 0 1
```

【样例 1 输出】

```
2
```

【样例 1 解释】

一种可能的游戏情形是：

- 第一轮 $T = [1, 0, 1, 0]$，$x = 0$，转化后的 $S' = [1, 1, 1, 1]$。
- 第二轮 $T = [1, 1, 1, 1]$，无论 x 取什么，转化后的 $S' = [2, 2, 2, 2]$。

可以证明 2 轮是最优的。

H. 鸭棋 / Duckchess

时间限制： 1.0 秒

空间限制： 512 MB

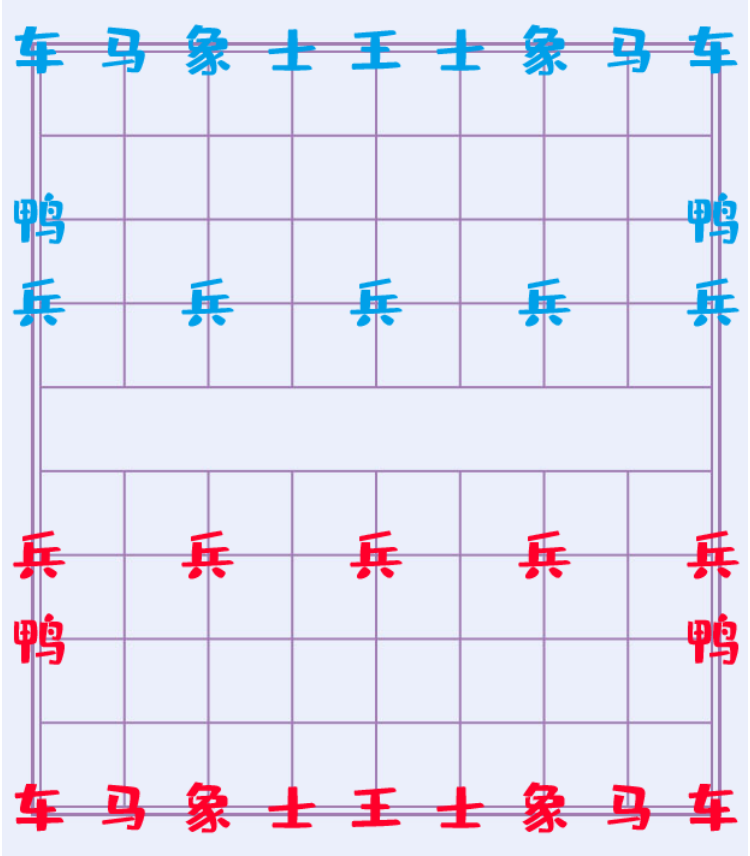
【题目背景】

鸭棋是一种风靡鸭子界的棋类游戏。事实上，它与中国象棋有一些相似之处，但规则不尽相同。在这里，我们将为你介绍鸭棋的规则。

同时，我们下发了一个模拟鸭棋规则的玩具，你可以结合这个玩具理解题目（也可以在 AK 后与你的队友进行对弈）。详情请见【玩具使用说明】。

鸭棋在一个 10×9（10 行 9 列）的网格棋盘上进行，网格上的每个格点都可以有棋子停留。对弈双方一方执红（`red`）棋、另一方执蓝（`blue`）棋，双方轮流执行操作，轮到一位玩家操作时，他必须选择一枚自己的棋子，并按照规则进行一步移动。

鸭棋发明者鸭子德规定一局鸭棋由红方执先手，并设计了初始棋盘布局如下：



【棋子类型与走子规则】

棋子分为 7 类，下面介绍了它们的名字以及它们的移动规则。介绍移动规则时，我们默认棋子所处位置为 (x, y)（表示第 x 行的第 y 列，下同），并列出它可以到达的位置：

- 王（`captain`）：可达的位置共 4 个，包括 $(x \pm 1, y)$ 及 $(x, y \pm 1)$。

- 士（guard）：可达的位置共 4 个，包括 $(x\pm1, y\pm1)$ 及 $(x\pm1, y\mp1)$。
- 象（elephant）：可达的位置至多 4 个，对于任意 $s_x, s_y \in \{1, -1\}$，分别有：
  - 如果位置 $(x + s_x \times 1, y + s_y \times 1)$ 上**无任意一方**的棋子停留，则 $(x + s_x \times 2, y + s_y \times 2)$ 为一个可达的位置。
- 马（horse）：可达的位置至多 8 个，对于任意 $s_x, s_y \in \{1, -1\}$，分别有：
  - 如果位置 $(x + s_x \times 1, y)$ 上**无任意一方**的棋子停留，则 $(x + s_x \times 2, y + s_y \times 1)$ 为一个可达的位置。
  - 如果位置 $(x, y + s_y \times 1)$ 上**无任意一方**的棋子停留，则 $(x + s_x \times 1, y + s_y \times 2)$ 为一个可达的位置。
- 车（car）：可在**不跨越其他棋子**的前提下，到达同行或同列的所有其他位置。
- 鸭（duck）：可达的位置至多 8 个，对于任意 $s_x, s_y \in \{1, -1\}$，分别有：
  - 如果位置 $(x + s_x \times 2, y + s_y \times 1), (x + s_x \times 1, y)$ 上均**无任意一方**的棋子停留，则 $(x + s_x \times 3, y + s_y \times 2)$ 为一个可达的位置。
  - 如果位置 $(x + s_x \times 1, y + s_y \times 2), (x, y + s_y \times 1)$ 上均**无任意一方**的棋子停留，则 $(x + s_x \times 2, y + s_y \times 3)$ 为一个可达的位置。
- 兵（soldier）：可达的位置共 8 个，包括 $(x\pm1, y)$ 及 $(x, y\pm1)$ 及 $(x\pm1, y\pm1)$ 及 $(x\pm1, y\mp1)$。

除上面描述的规则之外，棋子移动还有如下额外规则：

- 不能将棋子移动到棋盘外的某个位置。
- 玩家不能将棋子移动到**已经停留了己方棋子**的位置。
- 如果玩家将棋子移动到了一个**已经停留了对方棋子**的位置，那么原本停留在该位置上的这个**对方棋子**将被移出游戏。

【胜利条件与将军局面】

玩家在这个游戏中的目标是将对方的**王**移出游戏。一旦一方的**王**被移出游戏，则另一方立即宣告胜利。

对于一个棋盘的状态，如果存在一方有一步合法的操作能够将另一方的**王**移出游戏，则我们说当前局面是一个**将军**的局面。需要友情提示的是，根据定义，将军局面的形成包括（但不限于）如下这些可能：

1. 一方将一枚棋子移动到可以攻击对方**王**的位置
2. 在己方**王**受到威胁时不采取措施躲避
3. 主动将**王**移动至会受到攻击的位置

除此之外，需要特别说明的是，游戏结束后，由于双方不可再操作，因此不可能出现将军局面，即便此时另一方王处于被“攻击”的位置。

【题目描述】

今年的 IDCC（International Duck Chess Competition，国际鸭棋大赛）正在如火如荼地进行着。你观摩了一场精彩绝伦的比赛，但你对对弈过程的记忆已经模糊不清了，只有系统留下的他们的**操作序列**，序列中的每个**操作**为当前操作者试图移动某个位置的棋子至另一个位置。你希望用这个序列，来复现出整局棋局的对弈过程。即，对于每步操作，你需要**首先判其是否合法**，若合法，则**进一步求出**：

1. 这步操作移动了哪个棋子。
2. 这步操作后，是否存在棋子被移出游戏，如有则还需求出被移出游戏的棋子。
3. 这步操作后，是否形成将军局面。
4. 这步操作后，游戏是否结束。

可能包含的不合法情况如下：

- 此步移动的初始位置无己方棋子停留。
- 此步移动的初始位置有己方棋子停留，但移动不符合规则。
- 游戏已经结束。

序列中的不合法操作是需要被忽略的。比如，如果轮到红方移动，此时序列中的当前操作恰好是不合法的，则这个操作将被忽略，序列中的下一步操作将成为红方这步的操作（如仍不合法则继续忽略，直至出现合法的操作）。

【输入格式】

从标准输入读入数据。

第一行一个非负整数 Q，表示操作序列的长度。接下来依次描述每个操作。

接下来 Q 行，每行 4 个整数 x_s, y_s, x_t, y_t（$0 \le x_s, x_t < 10$，$0 \le y_s, y_t < 9$），描述一个欲将 (x_s, y_s) 处棋子移动到 (x_t, y_t) 的操作。在这里，我们规定左下角（即红方**车**摆放的位置，图见【题目背景】）为 $(0, 0)$。

保证 $Q \le 1000$。

【输出格式】

输出到标准输出。

输出 Q 行，对于每个操作依次输出复现结果。每行输出一个操作的结果：

- 如果该操作为不合法操作，则请输出 `Invalid command`。
- 如果为合法操作，则依次回答【题目描述】中的问题 1 ~ 4：
  - 被移动的棋子用 `[颜色] [类型]`（注意中间包含空格）来描述，请使用它们的英文名称（见【题目背景】）。如，红象为 `red elephant`，蓝王为 `blue captain`。

- 被移出游戏的棋子的描述方式与上面类似。特别地，**如果无棋子被移出游戏，则该问题的答案为 NA**。
- 用 yes、no 分别表示形成、不形成将军局面。
- 用 yes、no 分别表示游戏结束、游戏不结束。
- 用 ;（分号）将所有问题的答案隔开。
- 比如，四个问题的答案分别为：被移动的棋子是蓝车，无棋子被移出游戏，形成将军局面，游戏未结束。则该行应输出 `blue car;NA;yes;no`。

【样例 1】

见题目目录下的 ***1.in*** 与 ***1.ans***。

【玩具使用说明】

你可以在玩具所在目录下执行如下命令来运行玩具：

```
./duckchess.sh
```

特别地，在**初次运行前**，你需要执行如下命令为它添加运行权限：

```
chmod +x duckchess.sh
chmod +x duckchess
```

I. 不等式 / Inequality

时间限制： 4.0 秒

空间限制： 512 MB

【题目背景】

时光回到 2017 年 6 月 7 日。午后，阳光正好。

现在的你，在考场中笔耕不辍。在刷刷声中，你填写着交给从前和未来的自己的答卷。

像无数次训练过的那样，你直接跳到了这张数学试卷的最后一道大题，二选一的题目直接选择了后者。快速地掠过了题目描述，紧缩的眉头渐渐放松。

“稳了。”

你一刻也不敢停留，又向你的梦想靠近了一小步。

【题目描述】

已知两个 n 维实向量 $\vec{a}=(a_1,a_2,\ldots,a_n),\vec{b}=(b_1,b_2,\ldots,b_n)$，定义 n 个定义域为 $\mathbb{R}$ 函数 $f_1,f_2,\ldots,f_n$：

$$f_k(x)=\sum_{i=1}^{k}|a_ix+b_i| \quad (k=1,2,\ldots,n)$$

现在，对于每个 $k=1,2,\ldots,n$，试求 f_k 在 $\mathbb{R}$ 上的最小值。可以证明最小值一定存在。

【输入格式】

从标准输入读入数据。

第一行一个整数 n，表示向量的长度及函数的个数。

接下来两行，每行 n 个整数，分别描述向量 $\vec{a},\vec{b}$ 的各个分量，以空格隔开。

对于所有的输入数据，都满足 $1\le n\le 5\times 10^5,|a_i|,|b_i|<10^5$。

【输出格式】

输出到标准输出。

输出 n 行，第 i（$i=1,2,\ldots,n$）行为一个实数，表示 f_i 在 $\mathbb{R}$ 上的最小值。

输出与标准答案的绝对误差或相对误差小于 10^{-6} 就会被算作正确。

【样例 1 输入】

```
2
1 1
1 2
```

【样例 1 输出】

```
0.00000
1.00000
```

【样例 1 解释】

$f_1(x) = |x + 1|$，显然在 $x = -1$ 处取到最小值 0；
$f_2(x) = |x + 1| + |x + 2|$，可以证明其在 $[-2, -1]$ 中任意位置取到最小值 1。

【后记】

后来，全国三卷的考生们又回想起了被参数方程支配的恐惧。

J. 改善生活 / Improve

时间限制： 1.0 秒

空间限制： 512 MB

【题目描述】

“改善生活”是小 Z 创建的一个群聊。在群聊里，小 Z 和他的 $n-1$ 个朋友们（共 n 名群友，小 Z 的编号为 1，他的朋友们的编号从 2 至 n）无话不说，畅谈甚欢。然而，经常水群会被冠以水王的名号，这让小 Z 头痛不已。

今天，小 Z 预见到了群里可能会有 n 个话题（编号从 1 至 n）。其中，第 i 个话题是 c_i 号群友（当然也有可能是小 Z 自己）感兴趣的话题，这意味着该话题如果出现，这位群友将会进行 w_i 分钟的**激烈发言**。方便起见，你可以认为，除此之外，群友不会进行激烈发言。

所有 n 个话题之间有 m 组引导关系，每组引导关系的形式是一个二元组 (u,v)，它表示如果 u 号话题出现，**必定**会导致 v 号话题出现。

巧合的是，小 Z 发现，所有他自己的**不同**话题都不存在**直接或间接**的引导关系。

由于期中考试的临近，除小 Z 外的群友们都忙于复习，因此他们不会主动发起话题（发起话题指让一个话题出现，下同），也就是说，**所有 $c_i \neq 1$ 的话题都只能由引导关系直接或间接引出**。这让想要水群、却又希望摆脱水王名号的小 Z 左右为难。因此，他决定主动发起**一个或以上**的**自己感兴趣**的话题，来诱导其他话题的出现，致使**水群最多的另一位群友激烈发言的时间**与**小 Z 自己激烈发言的时间**的比值尽可能大。即最大化下面这个式子：

$$\frac{\max\limits_{k=2}^{n} sum(k)}{sum(1)}$$

其中，$sum(k)$ 表示所有**出现**且**群友 k 感兴趣**的话题的 w 值总和。

为避免精度误差，你只需要求出最大值**向下取整**的结果即可。

【输入格式】

从标准输入读入数据。

第一行两个正整数 n,m，分别表示话题数（恰好也是群人数）、引导关系组数。

第二行 n 个正整数 $c_1,\ldots,c_n$（$1 \le c_i \le n$），依次描述对各话题感兴趣的群友编号。保证至少存在一个 i 使得 $c_i = 1$。

第三行 n 个正整数 $w_1,\ldots,w_n$（$1 \le w_i \le 100$），依次描述各话题感兴趣的群友将激烈发言的时间。

接下来 m 行描述引导关系，每行两个正整数 u,v（$1 \le u,v \le n$），描述一组引导关系 (u,v)，具体意义见【题目描述】，**保证所有不同的 $c_i = 1$ 的话题之间两两不存在直**

接或间接的引导关系。

对于每一行，如果行内包含多个数，则用单个空格将它们隔开。

保证 $1 \le n \le 700$，$1 \le m \le 60000$。

【输出格式】

输出到标准输出。

一行一个整数，表示所求式子最大值向下取整的结果，即不超过该值的最大整数。

【样例 1 输入】

```
7 8
2 2 1 1 3 3 4
100 100 40 20 100 50 40
1 3
2 3
1 4
2 4
3 5
4 6
3 7
4 7
```

【样例 1 输出】

```
2
```

【样例 1 解释】

小 Z 可以选择发起编号为 3 和 4 的话题，这将致使编号为 5、6、7 的话题出现，并引发 3 号群友时长 150 分钟的激烈发言、以及 4 号群友时长 40 分钟的激烈发言。由于 3 号群友激烈发言时间更长，且小 Z 自己的激烈发言时长为 60 分钟，因此所求最大比值为 $\frac{150}{60} = 2.5$，这个值向下取整的结果是 2。

可以证明小 Z 不存在更优的策略。

K. 找树 / Findtree

时间限制： 8.0 秒

空间限制： 512 MiB

【题目描述】

定义 $\otimes_1, \otimes_2, \otimes_3$ 分别为按位与、按位或、按位异或运算。记 a_i 表示 a 的从低位到高位的第 i 个二进制位。定义一个作用在 w 位二进制数上的新运算 $\oplus$，满足对于结果 $a \oplus b$ 的每一位 $(a \oplus b)_i$ 有 $(a \oplus b)_i = a_i \otimes_{o_i} b_i$。不难验证 $\oplus$ 运算满足结合律和交换律。

给出一张 n 个点 m 条边的无向图，每一条边的权值是一个 w 位二进制数（即小于 2^w 的非负整数）。请你找一棵原图的生成树。设你找出的生成树中的边边权分别为 $v_1, \cdots, v_{n-1}$，请你最大化 $v_1 \oplus v_2 \oplus \cdots \oplus v_{n-1}$。

【输入格式】

从标准输入读入数据。

第一行两个正整数 n, m；

第二行一个长度为 w 的串，串中的每个字符为 `&`、`|`、`^` 中的一个（分别代表与、或和异或），表示每一个 $\otimes_{o_i}$ 。

接下来 m 行，每一行三个非负整数 x, y, v，表示一条连接 x 和 y 权值为 v 的边，保证 $1 \le x, y \le n$，$0 \le v < 2^w$。

保证 $n \le 70$，$w \le 12$，$m \le 5000$。

【输出格式】

输出到标准输出。

输出一行一个数，表示答案。如果图不连通，输出 `-1`。

【样例 1 输入】

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

【样例 1 输出】

```
1
```

L. 大碗宽面 / Large

时间限制： 2.0 秒

空间限制： 512 MB

【题目描述】

Yazid 喜欢吃大碗宽面。现有 m 碗宽面，其中第 i 碗宽面（$1 \le i \le m$）共包含 n_i 根面条，它们的宽度分别为 $A_{i,1}, A_{i,2}, \ldots A_{i,n_i}$。

记 $f(u,v)$ 表示若混合第 u 碗宽面和第 v 碗宽面，将得到的超大碗宽面的第 $\lfloor \frac{n_u+n_v+1}{2} \rfloor$ 小的面条宽度（$\lfloor x \rfloor$ 表示不超过 x 的最大整数）。

Yazid 想求出所有 $f(u,v)$，但为了节省你的输出时间，你只需要对所有 $1 \le u \le m$ 求出：

- $R(u) = \operatorname{xor}_{v=1}^{m} (f(u,v) + u + v)$（xor 指异或运算，在 C++ 语言中对应 ^ 运算符）。

【输入格式】

从标准输入读入数据。

第一行一个正整数 m，表示宽面碗数。

接下来 m 行，每行若干个用单个空格隔开的整数描述一碗宽面：这部分的第 i 行的第一个正整数为 n_i，表示第 i 碗宽面包含的面条数；接下来 n_i 个非负整数 $A_{i,1}, A_{i,2}, \ldots A_{i,n_i}$ 描述各面条的宽度。

保证 $m \le 10000$，$n_i \le 500$，$0 \le A_{i,j} \le 10^9$。

【输出格式】

输出到标准输出。

输出 n 行，每行一个整数，其中第 i 行的整数为 $R(i)$。

【样例 1 输入】

```
3
3 1 2 3
3 3 4 5
2 4 2
```

【样例 1 输出】

```
4
7
7
```

【样例 1 解释】

$$R(1) = (f(1,1)+2) + (f(1,2)+3) + (f(1,3)+4) = 4 \operatorname{xor} 6 \operatorname{xor} 6 = 4$$
$$R(2) = (f(2,1)+3) + (f(2,2)+4) + (f(2,3)+5) = 6 \operatorname{xor} 8 \operatorname{xor} 9 = 7$$
$$R(3) = (f(3,1)+4) + (f(3,2)+5) + (f(3,3)+6) = 6 \operatorname{xor} 9 \operatorname{xor} 8 = 7$$

M. 历史行程 / M

时间限制： 1.0 秒

空间限制： 512 MB

【题目背景】

人生是怎么样的呢？我们是怎样活着的呢？

无论是谁，都是从黑暗中降生，最后再回归于黑暗。几年，几十年，短暂而漫长的时光。形形色色的人，形形色色的事，如过眼云烟，转瞬即逝。过眼繁花终有尽时，有些东西还能弥留于你的记忆之中，但更多的，早已无可寻觅。

但是，有个人，你不会忘记。

在你降生于黑暗之中时，那个人就一直陪在你身边。当你突破那黑暗，第一次见到世界的光明的时候，那个人也在你身边。

第一次的啼哭，第一次的说话，第一次的走路，第一次的奔跑。

每一次的成长，是你自己的一次飞跃。而对那个人来说，则是最纯真的喜悦和感动。

在你不知道的时候，在你不知道的地方，那个人倾尽自己的一切，为你创造最好的一切，保护你的一切，为你指引走向未来的桥梁。

那个人每天都在期盼着，期盼着你能早点长大，变得拥有保护你自己的力量。

那个人每天都在担心着，担心着你与那个人分别那天，终究还是会来到。

你终究会一直成长，你终究会拥有自立自强的能力。那个人，也不能再像以前一样，保护你。

有一天，你将走向更宽阔的未来，你会拥有和以前完全不一样的一切。而那个人，已经不再能够保护你了。

而你和那个人之间能够拥有的时间，也就越来越短了。

五年，十年，二十年，你一天一天长大。

五年，十年，二十年，那个人一天天老去。

那个人已经不能保护你了，那么现在，该你，去保护那个人了。

也许那个人不能永远在你身边，也许那个人终将离你而去。

但你永远不会忘记那个人，因为那个人，始终在你身边。

今天，是那个人的节日。

【题目描述】

母亲节是一个为感谢母亲而庆祝的节日，而在世界各地的母亲节的日期有所不同。母亲们在这一天里通常会收到孩子们送的礼物；而在许多人心目中，康乃馨被视作最适于献给母亲的鲜花之一。

1913 年，美国国会确定将每年 5 月的第二个星期日作为法定的母亲节，这也是现代母亲节的起源。

给定一个年份，请你输出这一年的母亲节的日期。方便起见，你只需要输出它是这年 5 月的第几天即可。

【输入格式】

从标准输入读入数据。

一行一个整数 y，表示年份。

保证 $1913 \leq y \leq 2019$。

【输出格式】

输出到标准输出。

一行一个整数，表示这年的母亲节是 5 月的第几天。（请注意不要输出多余的前导零）

【样例 1 输入】

```
2019
```

【样例 1 输出】

```
12
```

【样例 1 解释】

2019 年的母亲节是 5 月 12 日。