Problem A. 2090 病毒

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	1024 megabytes

2090 年，蓝星被一种 “梗病毒” 席卷。每一个感染该病毒的人都只能使用 “梗语言” 交流。一句话可以看作一个仅包含小写英文字母的字符串 s。当且仅当满足以下所有条件时，该字符串被称为梗语言：

- s 的长度恰好为 8；
- 第 1、3、5、7 个字符是辅音字母（即不是 ‘a’、‘e’、‘i’、‘o’ 或 ‘u’）；
- 第 2、4、6、8 个字符是元音字母（‘a’、‘e’、‘i’、‘o’ 或 ‘u’）。

例如，“bababoyi” 和 “bibilabu”都是合法的梗语言。而 “wodedaodun”（长度不为 8）、“aiyogama”（第 1 个字符 ‘a’ 不是辅音）以及 “ruheneng”（第 8 个字符 ‘g’ 不是元音）都不是梗语言。

你现在是蓝星避难所的负责人，需要找出未被梗病毒感染的幸存者。判断方法是让每个人说一句话，如果这句话不是“梗语言”，则该人确认为未感染，否则视为已感染。给出 n 个人所说的话，请你判断每个人是否是幸存者。

Input

第一行包含一个整数 n $(1 \le n \le 10^5)$，表示待判断的人数。

接下来 n 行，每行包含一个仅由小写英文字母组成的字符串 s，表示每个人说的一句话。

保证所有字符串 s 的总长度不超过 10^6。

Output

输出 n 行。对于第 i 行，如果第 i 个人说的话是梗语言，输出 “Suspected Virus”（不含引号）；否则输出 “Well-Being”（不含引号）。

Example

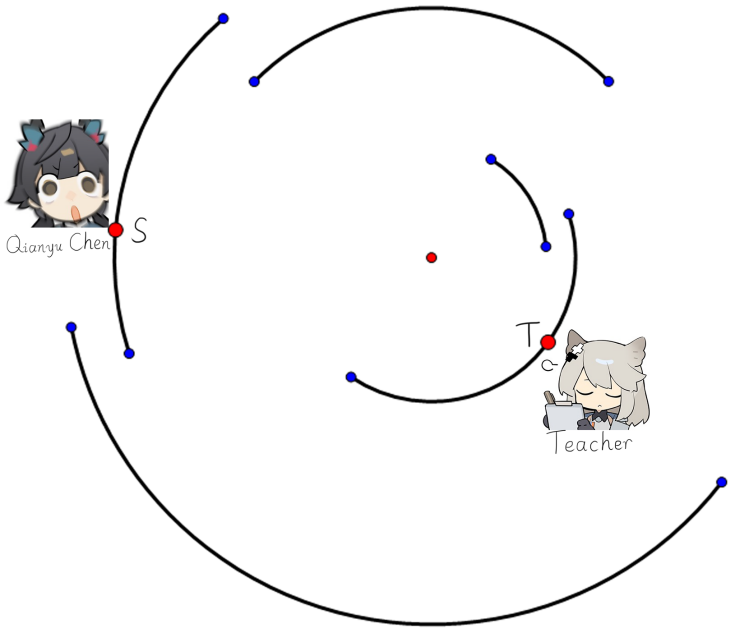
standard input	standard output
6 bababoyi bibilabu wodedaodun aiyogama ruheneng hehehahi	Suspected Virus Suspected Virus Well-Being Well-Being Well-Being Suspected Virus

Problem B. 当弃即弃

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

上课了，可是陈千语同学还在被窝里！

在学校的建筑俯视图中，教室和宿舍分布在 n 条半径不同的同心圆弧上。每条圆弧代表一座建筑，由其半径 r_i 和对应的圆心角范围 $\langle \theta_{i,1}, \theta_{i,2} \rangle$ 所界定。



行走规则如下：人可以沿着任意建筑对应的圆弧移动（圆周方向），也可以沿任意半径方向移动（径向直线）。然而，严禁穿过或停留在圆心（$r = 0$）处。如果我们用 (r, θ) 来表示位置，其中 r 为到圆心的距离，θ 为自 x 轴正半轴起逆时针计算的弧度角，那么具体的行走规则可以形式化地描述为：

- 人可以从 (r, θ_1) 走到 (r, θ_2)，当且仅当存在一条半径为 r 的建筑圆弧，使得 θ_1 到 θ_2（或反方向）之间的整个圆心角路径完全位于该圆弧的范围之内；
- 人可以沿径向直线从 (r_1, θ) 走到 (r_2, θ)，当且仅当 $r_1 > 0$ 且 $r_2 > 0$（即在同一角度 θ 下，可在任意非零半径之间进行连续的直线移动）。

陈千语同学从宿舍 (r_s, θ_s) 出发，需要前往教室 (r_t, θ_t)。请计算所需的最短路程。如果教室无法到达，输出 "`Cast off as cast!`"（当弃即弃！）。

Input

第一行包含五个整数 n $(1 \le n \le 10^5)$, r_s, d_s, r_t, d_t $(1 \le r_s, r_t \le 10^5, 0 \le d_s, d_t < 10^5)$，分别表示建筑的数量、宿舍的位置和教室的位置。任意位置的角度 θ 由输入的整数 d 通过 $\theta = \frac{2\pi d}{10^5}$ 弧度换算得到。

接下来 n 行，每行包含三个整数 r_i, $d_{i,1}$, $d_{i,2}$ $(1 \le r_i \le 10^5, 0 \le d_{i,1}, d_{i,2} < 10^5)$，描述一座建筑对应的圆弧。圆弧的圆心角范围 $\langle \theta_{i,1}, \theta_{i,2} \rangle$ 顺着逆时针方向解读如下：

- 若 $d_{i,1} \le d_{i,2}$，则圆弧从 $\theta_{i,1} = \frac{2\pi d_{i,1}}{10^5}$ 逆时针连续延伸至 $\theta_{i,2} = \frac{2\pi d_{i,2}}{10^5}$ 弧度。当两值相等时，建筑退化为单个点；

- 若 $d_{i,1} > d_{i,2}$，则圆弧从 $\theta_{i,1} = \frac{2\pi d_{i,1}}{10^5}$ 逆时针连续延伸至 2π（即 x 轴正半轴），随后立即从 0 继续延伸至 $\theta_{i,2} = \frac{2\pi d_{i,2}}{10^5}$ 弧度。

保证所有的 r_i 互不相同，且宿舍和教室均位于某座建筑上。

Output

如果教室无法到达，输出一行一个字符串 "`Cast off as cast!`"（不含引号）。否则，输出一行包含一个实数，表示最短路程。

如果你的答案的绝对误差或相对误差不超过 10^{-6}，则将被视为正确。形式化地，设你的输出为 a，标准答案为 b，当且仅当 $\frac{|a-b|}{\max(1,|b|)} \le 10^{-6}$ 时，你的输出会被接受。

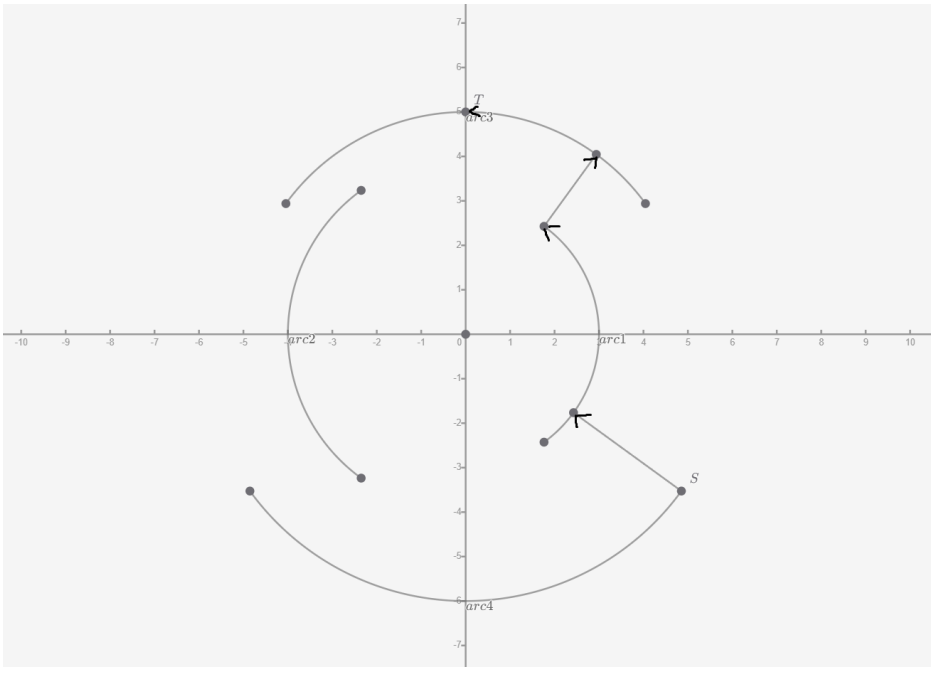
Examples

standard input	standard output
4 6 90000 5 25000 3 85000 15000 4 35000 65000 5 10000 40000 6 60000 90000	12.853981633974
2 1 0 2 1 1 0 0 2 1 1	Cast off as cast!

Note

对于第一个样例，如图所示，最短路径由四段组成：两段径向移动和两段圆弧移动。总路程为：

$$\underbrace{3}_{\text{径向}} + \underbrace{\left(\frac{25000 \cdot 2\pi}{10^5}\right) \cdot 3}_{\substack{\text{圆弧位于 } r=3 \\ \Delta d=25000}} + \underbrace{2}_{\text{径向}} + \underbrace{\left(\frac{10000 \cdot 2\pi}{10^5}\right) \cdot 5}_{\substack{\text{圆弧位于 } r=5 \\ \Delta d=10000}} = 5 + \frac{5\pi}{2} \approx 12.853981633974$$



Problem C. 大鱼吃小鱼

Input file:	standard input
Output file:	standard output
Time limit:	4 seconds
Memory limit:	1024 megabytes

给一个 $n \times m$ 的网格，部分格子上有鱼，其余格子是障碍。其中有一条鱼受你控制，可以反复将其移动到上下左右相邻的格子，但不能移出网格：若目标格子有一条大小不超过自身的鱼，则可以吃掉它，自身大小增加 1；若目标格子是障碍或鱼比自身大，则无法移动。

初始时所有格子都是障碍。依次有 q 次操作，操作有两种：

1. 把 (x, y) 处的障碍去除，并放入一条大小为 v 的鱼，要求出这条鱼在你的控制下最多能吃掉多少条鱼。保证 (x, y) 处当前是障碍，且 v 不小于之前放入过的所有鱼的大小。
2. 假设在假想过程开始前，可以将 (x, y) 处鱼的初始大小增加一个非负整数。设在所有可能的增加量中，该鱼理论上最多能吃到 k 条鱼。求为了能吃到 k 条鱼，所需增加的最小大小。保证 (x, y) 处之前放入过鱼。

对于两类操作，吃鱼的过程是假设性的，不会实际改变任何已放入鱼的状态。

Input

第一行包含三个整数 n，m $(1 \le n, m, n \times m \le 2.5 \times 10^5)$ 和 q $(1 \le q \le 5 \times 10^5)$，表示网格的大小和操作数量。

您需要依次处理 q 个操作。对于每个操作，给出加密后的坐标 x' 和 y'。令 l 为上一个操作的输出结果（对于第一个操作，$l = 0$）。计算解密后的坐标 $x = x' \oplus l$，$y = y' \oplus l$，其中 $\oplus$ 表示按位异或。保证 $1 \le x \le n$，$1 \le y \le m$。操作 1 中的 v 未经加密。操作有以下两种：

1. `1 x' y' v`：计算解密后的坐标 x 和 y，把 (x, y) 处的障碍去除，并放入一条大小为 v $(1 \le v \le 10^9)$ 的鱼，要求出这条鱼在你的控制下最多能吃掉多少条鱼。保证 (x, y) 处当前是障碍，且 v 不小于之前放入过的所有鱼的大小。
2. `2 x' y'`：计算解密后的坐标 x 和 y。假设在假想过程开始前，可以将 (x, y) 处鱼的初始大小增加一个非负整数。设在所有可能的增加量中，该鱼理论上最多能吃到 k 条鱼。求为了能吃到 k 条鱼，所需增加的最小大小。保证 (x, y) 处之前放入过鱼。

Output

输出 q 行，每行包含一个整数，表示每次操作的答案。

Example

standard input	standard output
2 3 9 1 1 2 1 1 2 1 1 1 2 2 2 2 3 0 1 0 2 8 2 1 2 1 4 4 9 2 6 6 2 5 7	0 0 2 1 3 5 4 4 0

Note

以下对样例数据进行解释。把输入的坐标解密后：

1. `1 1 2 1`，在 $(1,2)$ 处放入大小为 1 的鱼，最多能吃 0 条鱼。
2. `1 2 1 1`，在 $(2,1)$ 处放入大小为 1 的鱼，最多能吃 0 条鱼。
3. `1 2 2 2`，在 $(2,2)$ 处放入大小为 2 的鱼，最多能吃 2 条鱼。
4. `2 1 2`，在 $(1,2)$ 处的鱼大小为 1，在允许任意变大的前提下最多能吃 2 条鱼，至少变大 1 才能吃到这个最大数量的鱼。
5. `1 1 3 8`，在 $(1,3)$ 处放入大小为 8 的鱼，最多能吃 3 条鱼。
6. `2 2 1`，在 $(2,1)$ 处的鱼大小为 1，在允许任意变大的前提下最多能吃 3 条鱼，至少变大 5 才能吃到这个最大数量的鱼。
7. `1 1 1 9`，在 $(1,1)$ 处放入大小为 9 的鱼，最多能吃 4 条鱼。
8. `2 2 2`，在 $(2,2)$ 处的鱼大小为 2，在允许任意变大的前提下最多能吃 4 条鱼，至少变大 4 才能吃到这个最大数量的鱼。
9. `2 1 3`，在 $(1,3)$ 处的鱼大小为 8，在允许任意变大的前提下最多能吃 4 条鱼，无需变大即可吃到这个最大数量的鱼。

Problem D. 别具一格

Input file:	standard input
Output file:	standard output
Time limit:	2 seconds
Memory limit:	1024 megabytes

有一类特别的巧克力棒，它们可以被视作各边长之比为 $s_1 : s_2 : \cdots : s_m$ 的 m 维超矩形，其中 $s_1, s_2, \ldots, s_m$ 是给定的正偶数。

Alice 和 Bob 打算分食这类巧克力棒。首先，他们会选择一个正整数 n，然后订购此类巧克力棒里边长分别为 $s_1 n, s_2 n, \ldots, s_m n$ 的那款，并将其放在桌上。之后两人轮流操作，Alice 先手。每次操作中，当前操作者需要将桌上的巧克力棒一刀切成两个超矩形块，且每块的各边长均为正整数，然后，当前操作者取走其中一块吃掉，另一块则留在桌上。过程直到双方都无法进行任何合法切割时结束。

形式化地说，在某一轮中，设桌上剩余巧克力棒的边长为 $l_1, l_2, \ldots, l_m$。若 $l_1 = l_2 = \cdots = l_m = 1$，则该最后剩下的巧克力棒会被丢弃，不由任何人吃掉。否则，当前操作者选择一个满足 $l_i > 1$ 的维度 i（$1 \le i \le m$），并选择一个整数切割位置 k（$1 \le k < l_i$）。随后，巧克力棒被切成两个 m 维巧克力棒，它们的边长分别为 $l_1, \ldots, l_{i-1}, k, l_{i+1}, \ldots, l_m$ 和 $l_1, \ldots, l_{i-1}, (l_i - k), l_{i+1}, \ldots, l_m$。

然而，Alice 和 Bob 都患有蛀牙。他们既无法抵挡甜食的诱惑，却又担心蛀牙恶化。因此他们都会采取最优策略，使自己最终吃下的巧克力棒的总超体积（例如，二维巧克力棒对应面积，三维巧克力棒对应体积）尽可能小。每当一块边长为 $l_1, l_2, \ldots, l_m$ 的巧克力棒被吃掉时，它会为该总量贡献 $l_1 \times l_2 \times \cdots \times l_m$。

对于固定的 n，设 $A(n)$ 表示 Alice 最终吃下的总超体积，$B(n)$ 表示 Bob 最终吃下的总超体积。你的任务是计算当 $n \to \infty$ 时，差值 $A(n) - B(n)$ 的主渐近系数对 998 244 353 取模后的值。更具体地，可以证明存在一个次数 d 和一个非零有理数 C，使得对任意实数 $\varepsilon > 0$，都存在一个整数 N，使得对所有整数 $n \ge N$，都有 $\left| \frac{A(n)-B(n)}{n^d} - C \right| < \varepsilon$。也即，$C = \lim_{n\to\infty} \frac{A(n)-B(n)}{n^d}$。由于 C 可以被表示为一个最简分数 $\frac{P}{Q}$，请输出满足 $0 \le R < 998\,244\,353$ 且 $R \cdot Q \equiv P \pmod{998\,244\,353}$ 的整数 R。可以证明这样的 R 存在且唯一。

Input

第一行包含一个整数 m ($2 \le m \le 5 \times 10^5$)。

第二行包含 m 个整数 $s_1, s_2, \ldots, s_m$ ($2 \le s_i \le 10^9$，s_i 为偶数)。

Output

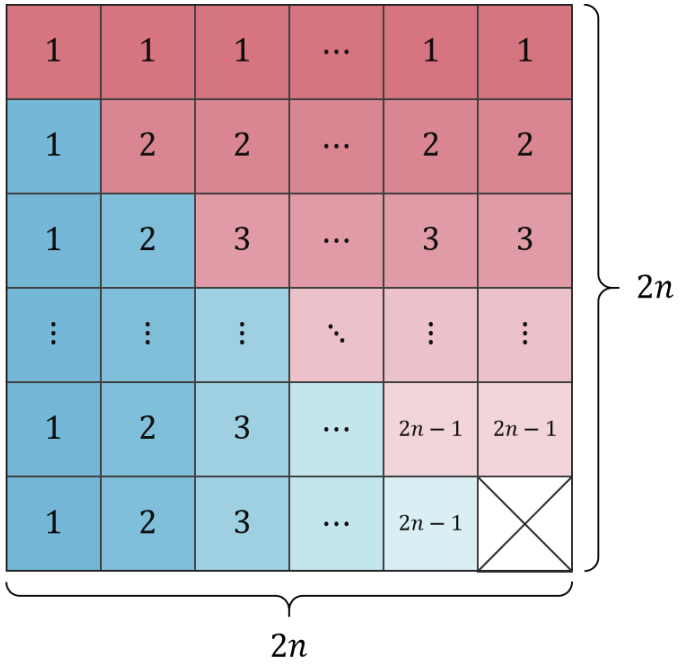
输出 C 对 998 244 353 取模后的值。

Examples

standard input	standard output
2 2 2	2
3 6 8 2	10

Note

对于第一个样例，巧克力棒是一个 $2n \times 2n$ 的正方形，因此每块被吃掉的巧克力棒的超体积就是其面积。下图展示了在最优策略下可能出现的一种结果：Alice 不断切下最上方的一行，Bob 不断切下最左侧的一列，最后剩下的那块 1×1 被丢弃。颜色表示每个单元格最终由 Alice 还是 Bob 吃掉，格子里的数字表示该格子是在第几轮被切下的。



每个由 Alice 吃掉的非对角线格子都有一个关于主对角线对称、由 Bob 吃掉的对应格子，唯一多出来的是 Alice 吃掉的 $2n-1$ 个对角线格子。因此 $A(n)-B(n)=2n-1$，其主渐近系数为 2，因为 $C=\lim_{n\to\infty}\frac{2n-1}{n}=2$。

Problem E. 排列求值

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

对于一个 0 到 $n-1$ 的排列 P，定义它的权值为：

$$f(P) = \sum_{0 \le i < j < n} (P_j - P_i)$$

给定一个 0 到 $n-1$ 的排列 P，输出它的权值。

Input

第一行包含一个整数 n ($1 \le n \le 2 \times 10^5$)。

第二行包含 n 个整数 $P_0, P_1, \ldots, P_{n-1}$，表示给定的 0 到 $n-1$ 的排列。

Output

输出一行包含一个整数，表示排列的权值。

Example

standard input	standard output
4 2 0 1 3	4

Note

对于样例：

$$\begin{aligned} f(P) &= (P_1 - P_0) + (P_2 - P_0) + (P_3 - P_0) + (P_2 - P_1) + (P_3 - P_1) + (P_3 - P_2) \\ &= (0-2) + (1-2) + (3-2) + (1-0) + (3-0) + (3-1) \\ &= -2 + (-1) + 1 + 1 + 3 + 2 \\ &= 4 \end{aligned}$$

Problem F. 排列生成

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	1024 megabytes

对于一个 0 到 $n-1$ 的排列 P，定义它的权值为：

$$f(P) = \sum_{0 \le i < j < n} (P_j - P_i)$$

给定一个 0 到 $n-1$ 的排列 P，以及两个整数 k 和 x $(0 \le k, x < n)$。要求构造一个 0 到 $n-1$ 的排列 P'，满足以下两个条件：

- ${P'}_k = x$（即 P' 的第 k 个元素为 x，下标从 0 开始）；
- $f(P') \equiv f(P) \pmod{n}$。

如果有解，输出任意一组符合条件的排列 P'；如果无解，输出 -1。

Input

第一行包含三个整数 n $(1 \le n \le 2 \times 10^5)$、$k$ 和 x $(0 \le k, x < n)$。

第二行包含 n 个整数 $P_0, P_1, \ldots, P_{n-1}$，表示给定的 0 到 $n-1$ 的排列。

Output

如果无解，输出一行包含一个整数 -1。否则输出一行包含 n 个整数，表示符合条件的排列 P'。

Example

standard input	standard output
4 1 3 2 0 1 3	0 3 1 2

Note

对于样例：

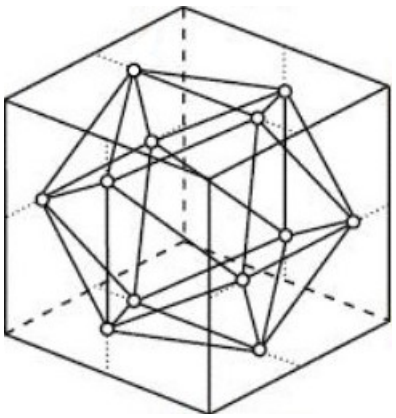
$$\begin{aligned} f(P) &= (P_1 - P_0) + (P_2 - P_0) + (P_3 - P_0) + (P_2 - P_1) + (P_3 - P_1) + (P_3 - P_2) \\ &= (0-2) + (1-2) + (3-2) + (1-0) + (3-0) + (3-1) \\ &= -2 + (-1) + 1 + 1 + 3 + 2 \\ &= 4 \equiv 0 \pmod{4} \end{aligned}$$

$$\begin{aligned} f(P') &= (P'_1 - P'_0) + (P'_2 - P'_0) + (P'_3 - P'_0) + (P'_2 - P'_1) + (P'_3 - P'_1) + (P'_3 - P'_2) \\ &= (3-0) + (1-0) + (2-0) + (1-3) + (2-3) + (2-1) \\ &= 3 + 1 + 2 + (-2) + (-1) + 1 \\ &= 4 \equiv 0 \pmod{4} \end{aligned}$$

Problem G. 精度误差？！

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

给定一个整数 n，你需要构造一个集合 S，该集合由不超过 $(2n+2)$ 个不同的点组成，这些点位于三维欧几里得空间中，使得对于每个点 $P \in S$，恰好有 n 个不同的点 $Q \in S$ 满足 $\mathrm{dis}(P,Q)=1$，其中 $\mathrm{dis}(P,Q)$ 是点 P 和点 Q 之间的欧几里得距离。



正二十面体，$n=5$ 的可行解之一。

为了容忍精度误差，设 $\epsilon=0.01$，只有在满足以下条件时你的答案才会被视为正确：

- 对于任何两个不同的点 $P,Q \in S$，$\mathrm{dis}(P,Q) > \epsilon$；
- 对于每个点 $P \in S$，恰好有 n 个不同的点 $Q \in S$ 满足 $1-\epsilon < \mathrm{dis}(P,Q) < 1+\epsilon$。

Input

输入的第一行包含一个整数 T $(1 \le T \le 100)$，表示测试数据组数。对于每组测试数据：
仅有一行包含一个整数 n $(1 \le n \le 100)$。

Output

对于每组测试数据：
第一行输出一个整数 m $(1 \le m \le 2n+2)$，表示集合 S 中点的数量。
接下来输出 m 行，每行包含三个实数 x，y 和 z $(-100 \le x,y,z \le 100)$，表示集合 S 中一个点的坐标。
可以证明在给定限制条件下总是存在可行解，如果有多种可行解，你可以输出任意一个。

Example

standard input	standard output
5 1 2 3 4 5	2 0.000000000 0.000000000 0.000000000 1.000000000 0.000000000 0.000000000 3 0.000000000 0.000000000 0.000000000 1.000000000 0.000000000 0.000000000 0.500000000 0.866025404 0.000000000 4 0.500000000 0.000000000 -0.353553391 -0.500000000 0.000000000 -0.353553391 0.000000000 0.500000000 0.353553391 0.000000000 -0.500000000 0.353553391 6 0.707106781 0.000000000 0.000000000 -0.707106781 0.000000000 0.000000000 0.000000000 0.707106781 0.000000000 0.000000000 -0.707106781 0.000000000 0.000000000 0.000000000 0.707106781 0.000000000 0.000000000 -0.707106781 12 0.000000000 0.500000000 0.809016994 0.000000000 -0.500000000 0.809016994 0.000000000 0.500000000 -0.809016994 0.000000000 -0.500000000 -0.809016994 0.500000000 0.809016994 0.000000000 0.500000000 -0.809016994 0.000000000 -0.500000000 0.809016994 0.000000000 -0.500000000 -0.809016994 0.000000000 0.809016994 0.000000000 0.500000000 0.809016994 0.000000000 -0.500000000 -0.809016994 0.000000000 0.500000000 -0.809016994 0.000000000 -0.500000000

Problem H. 石头剪刀布大师

Input file:	standard input
Output file:	standard output
Time limit:	4 seconds
Memory limit:	1024 megabytes

Alice 和 Bob 正在玩一个 k 轮的石头剪刀布游戏。游戏开始时，两位玩家各有恰好 3 张牌，每张牌是石头（R）、剪刀（S）或布（P）。每一轮游戏按照如下方式进行：

1. 双方均可看见彼此手中的全部 6 张手牌。
2. Alice 先从她的手牌中选择一张牌打出。
3. Bob 在看到 Alice 打出的牌后，选择自己的一张牌打出。
4. 根据标准规则判定胜负：石头胜剪刀，剪刀胜布，布胜石头。相同则为平局。若 Alice 获胜，她得 3 分；若平局，她得 1 分；若 Bob 获胜，她得 0 分。
5. 双方都将打出的牌丢弃，并各自独立地以等概率获得一张新的石头、剪刀或布。

Alice 希望最大化自己的期望总得分，Bob 希望最小化 Alice 的期望总得分。

给定游戏轮数 k、Alice 的初始手牌和 Bob 的初始手牌，你需要求出双方都采取最优策略时，Alice 的最大期望总得分。

Input

输入的第一行包含一个整数 T ($1 \le T \le 10^5$)，表示测试数据组数。对于每组测试数据：

第一行包含一个整数 k ($1 \le k \le 10^9$)，表示游戏轮数。

第二行包含一个长度为 3 的字符串，由字符 R, S, P 组成，表示 Alice 的初始手牌。

第三行包含一个长度为 3 的字符串，由字符 R, S, P 组成，表示 Bob 的初始手牌。

Output

对于每组测试数据，输出一行包含一个实数，表示 Alice 的最大期望总得分。

如果你的答案的绝对误差或相对误差不超过 10^{-6}，则将被视为正确。形式化地，设你的输出为 a，标准答案为 b，当且仅当 $\frac{|a-b|}{\max(1,|b|)} \le 10^{-6}$ 时，你的输出会被接受。

Example

standard input	standard output
2 1 RRS PSS 1000000000 RSP RSP	1.000000000000 706356363.862663573876

Problem I. 拼好题

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

还记得 2025 ICPC 沈阳区域赛中两道最为“臭名昭著”的题目 Problem F. 友谊地久天长 和 Problem K. 接力跳 吗？一道要求你给边定向，另一道让青蛙通过对称反射飞跃。如果你还记得，那么恭喜：今天它们卷土重来了，而且不是单个返场，而是打包成了一份套餐！

有 n 只编号为 1 到 n 的青蛙。青蛙 i 初始位于数轴上的整数坐标 x_i 处。若干对青蛙互为朋友，这些朋友关系可以被抽象为一张包含 n 个顶点和 m 条边的简单连通无向图。

你必须为每条无向边指定一个方向。随后，每只青蛙会独立地从自己的出边里选择恰好一条出边，并记住该出边另一端的青蛙作为自己在接下来的无限过程中的固定目标。在该过程的每一时刻，所有青蛙同时跳跃。若青蛙 i 的目标是青蛙 j，则青蛙 i 会落到其当前位置关于青蛙 j 当前位置的对称点上（若跳跃前青蛙 i 与青蛙 j 重合，则它仍落在该重合坐标处）。

你的任务是为所有边定向使得每个顶点都有至少一条出边，并且无论各青蛙如何选择其目标，每只青蛙都能距离自己的初始位置任意远。形式化地说，对于青蛙们任意可能的目标选择、任意青蛙 i，以及任意整数 $N > 0$，都必须存在某一时刻，使得青蛙 i 此时的坐标 x_i' 满足 $|x_i - x_i'| > N$。如果不存在这样的定向方案，请报告无解。

Input

输入的第一行包含一个整数 T $(1 \le T \le 10^5)$，表示测试数据的组数。对于每组测试数据：

第一行包含两个整数 n 和 m $(2 \le n \le 2 \times 10^5,\ n - 1 \le m \le \min\{2 \times 10^5, \frac{n(n-1)}{2}\})$，分别表示青蛙数和朋友关系对数。

第二行包含 n 个整数 $x_1, x_2, \ldots, x_n$ $(-10^9 \le x_i \le 10^9)$，其中 x_i 表示青蛙 i 的初始坐标。

接下来 m 行，每行包含两个整数 u 和 v $(1 \le u, v \le n)$，描述一条表示青蛙 u 与青蛙 v 互为朋友的无向边。保证给定图连通，且不含自环或重边。

保证所有测试数据中 n 的总和以及 m 的总和都不超过 2×10^5。

Output

对于每组测试数据：

如果可以为所有边定向来满足题目要求，则在第一行打印“`Yes`”（不含引号）。

接下来 m 行。每行必须包含两个整数 u' 和 v'，表示一条从 u' 指向 v' 的有向边。无序对 (u', v') 必须是输入里存在的某条无向边。你可以以任意顺序打印这些有向边。

如果不存在合法的定向方案，输出一行包含 "No"（不含引号）。

Example

standard input	standard output
2 4 5 1 2 3 4 1 2 1 3 1 4 2 3 3 4 2 1 4 -3 1 2	Yes 2 1 1 3 4 1 2 3 3 4 No

Note

对于样例的第一组测试数据，一种合法的定向方案是将各边定向为 $2 \to 1$、$1 \to 3$、$4 \to 1$、$3 \to 2$ 和 $3 \to 4$。若青蛙 3 选择青蛙 2 作为其固定目标，则所有青蛙的坐标变化为 $(1,2,3,4) \to (5,0,1,-2) \to (-3,10,-1,12) \to \cdots$。若青蛙 3 选择的是青蛙 4，则所有青蛙的坐标变化为 $(1,2,3,4) \to (5,0,5,-2) \to (5,10,-9,12) \to \cdots$。可以证明，这两种情况下每只青蛙都能距离自己的初始位置任意远。

对于样例的第二组测试数据，若将唯一的边定向为 $1 \to 2$ 会让 2 没有出边，定向为 $2 \to 1$ 会让 1 没有出边。因此，不存在合法的定向方案。

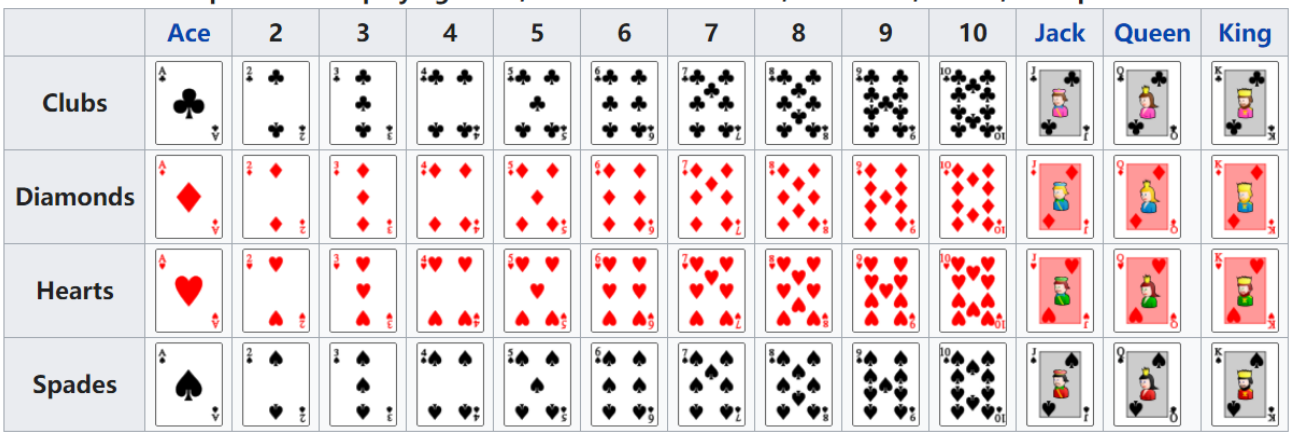
Problem J. 梭哈

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

你是今晚打老虎，正在与法国赌神皮尔克松进行一场梭哈对决。游戏使用一副标准的 52 张扑克牌。当前，每位玩家都有四张明牌和一张暗牌。由于法国赌神拥有超强的特异功能，他可以先把自己的暗牌变成除桌面上已经出现的八张明牌之外的任意一张牌。法国赌神变牌之后，你完全知晓他的选择，然后你可以把自己的暗牌变成除桌面上已经出现的八张明牌以及法国赌神变牌后的暗牌之外的任意一张牌。也就是说，除去已知的九张不同牌，总共有 43 张牌可选。如果你有必胜策略，输出“`WoYaoYanPai`”；如果法国赌神有必胜策略，输出“`GeiWoCaPiXie`”；如果双方都没有必胜策略，最终平局，输出“`PaiMeiYouWenTi`”。

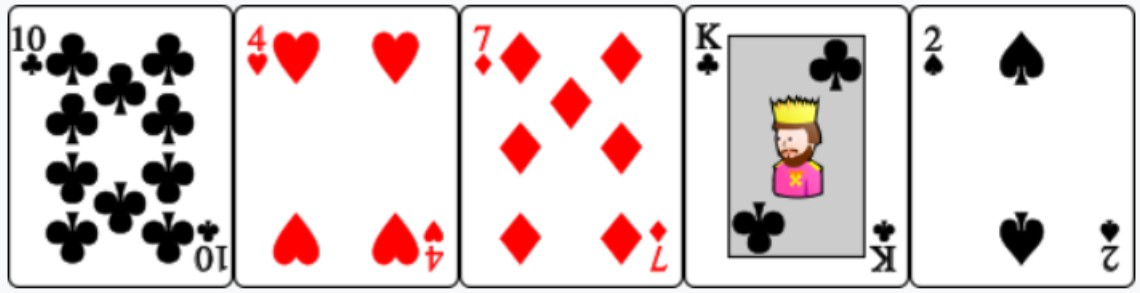
在梭哈游戏中，每位玩家最终手里有五张牌，并且所有玩家的牌都互不相同。所有牌都来自一副标准的 52 张扑克牌。标准扑克牌包含四种花色，每种花色各有 13 个点数：梅花（♣）、方块（♢）、红桃（♡）和黑桃（♠）。每种花色都包含 A、K、Q、J，以及从 2 到 10 的数字牌；每张牌上会印有对应花色的符号。同一张牌不能被抽取超过一次。

Example set of 52 playing cards; 13 of each suit: clubs, diamonds, hearts, and spades

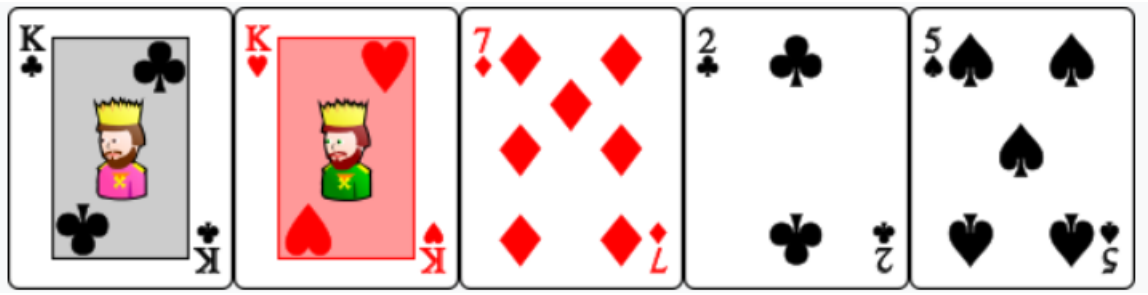


单张牌的点数大小关系如下（从大到小）：A, K, Q, J, 10, 9, 8, 7, 6, 5, 4, 3, 2。下表给出了所有可能的五张牌牌型，按其价值从小到大排列。每种牌型都有一种特定的五张牌排序方式，下面会进行说明。

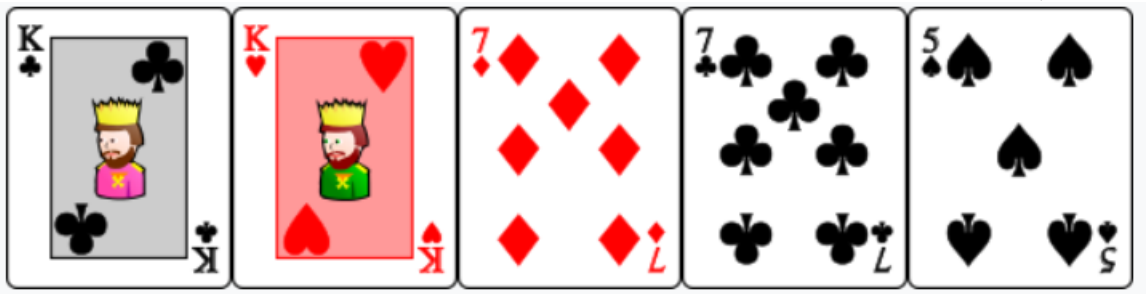
- Highcard: 单张牌按点数比较。牌按 $a_1a_2a_3a_4a_5$ 排列，使得 $a_1 > a_2 > a_3 > a_4 > a_5$。（$a_i$ 表示第 i 张牌的点数。）



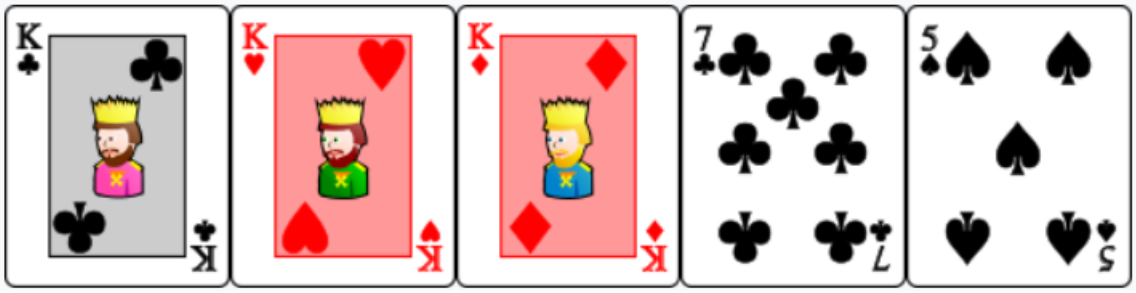
- Pair: 一对。牌按 $a_1a_2a_3a_4a_5$ 排列，使得 $a_1 = a_2$，$a_3 > a_4 > a_5$，且 $a_1 \neq a_3$，$a_1 \neq a_4$，$a_1 \neq a_5$。



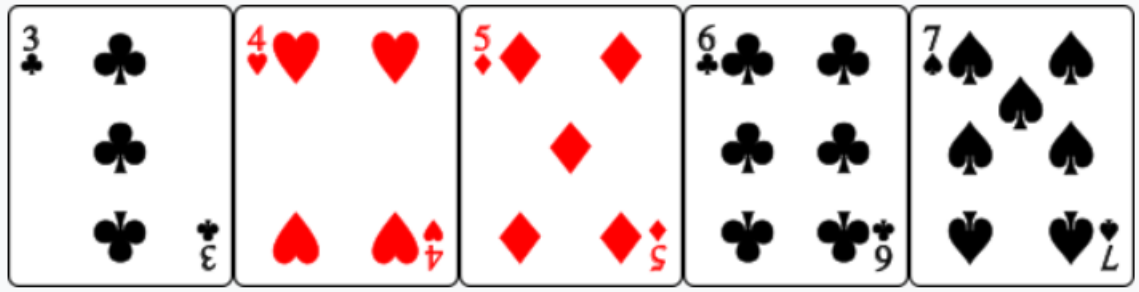
- Two pairs: 两对。牌按 $a_1a_2a_3a_4a_5$ 排列，使得 $a_1 = a_2$，$a_3 = a_4$，$a_1 > a_3$，且 $a_1 \neq a_5$，$a_3 \neq a_5$。



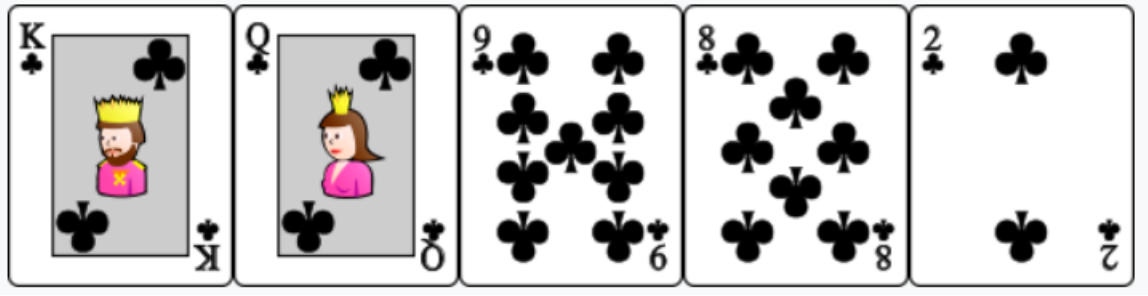
- Three of a kind: 三条。牌按 $a_1a_2a_3a_4a_5$ 排列，使得 $a_1 = a_2 = a_3$，$a_4 > a_5$，且 $a_1 \neq a_4$，$a_1 \neq a_5$。



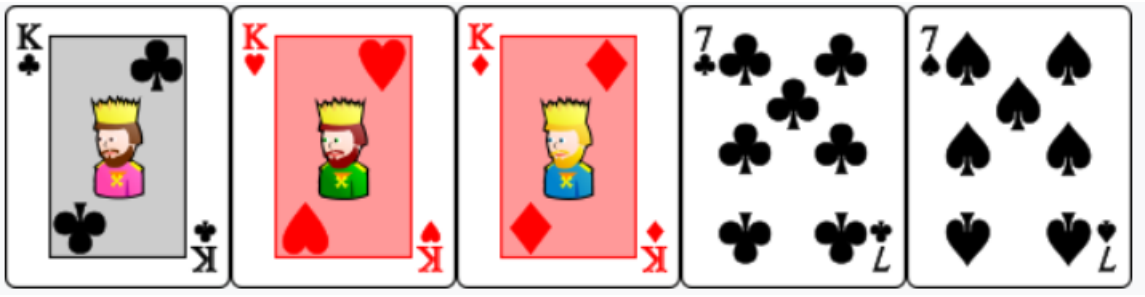
- Straight: 顺子，五张牌点数连续递增。牌按 $a_1a_2a_3a_4a_5$ 排列，使得对所有 $1 \leq i \leq 4$，都有 a_i 比 a_{i+1} 恰好大一张点数。特别地，如果 a_5 是 A，那么 a_4 可以是 2。在这种情况下，A 被视为比 2 小一张点数。



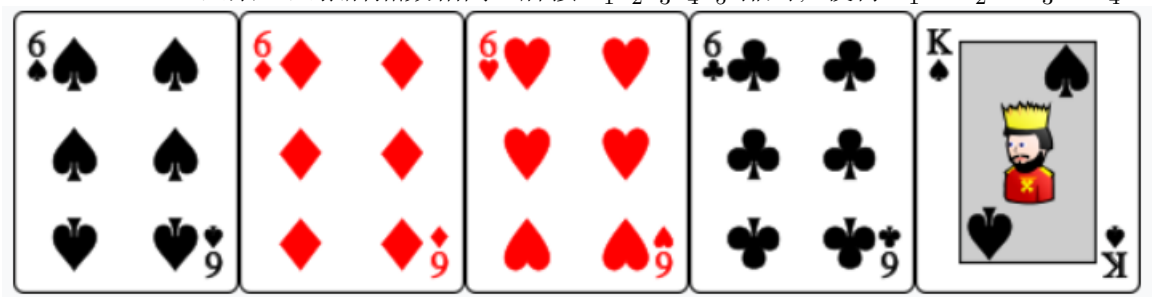
- Flush: 同花。五张牌花色相同。牌按 $a_1a_2a_3a_4a_5$ 排列，使得五张牌花色相同且 $a_1 > a_2 > a_3 > a_4 > a_5$。



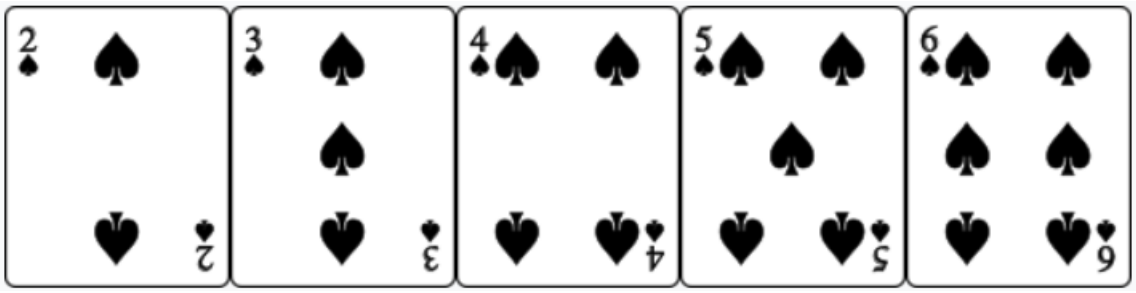
- Full house: 葫芦。由三条和一对组成。牌按 $a_1a_2a_3a_4a_5$ 排列，使得 $a_1 = a_2 = a_3$，$a_4 = a_5$。



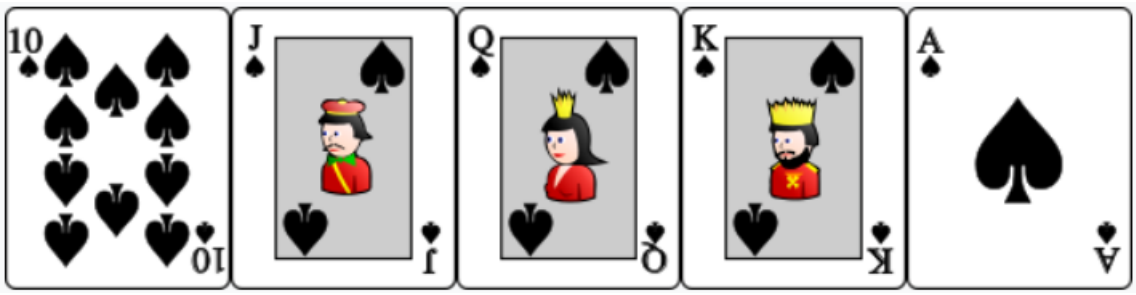
- Four of a kind: 四条。四张牌点数相同。牌按 $a_1a_2a_3a_4a_5$ 排列，使得 $a_1 = a_2 = a_3 = a_4$。



- Straight flush: 同花顺。同花的顺子。牌按 $a_1a_2a_3a_4a_5$ 排列，使得五张牌花色相同，且对所有 $1 \le i \le 4$，都有 a_i 比 a_{i+1} 恰好大一张点数。特别地，如果 a_5 是 A，那么 a_4 可以是 2。在这种情况下，A 被视为比 2 小一张点数。



- Royal flush: 皇家同花顺。从 10 到 A 的同花顺。牌按 $a_1a_2a_3a_4a_5$ 排列，使得 a_1, a_2, a_3, a_4, a_5 分别是同一花色的 A、K、Q、J、10。



比较两手牌时，首先比较两手牌的牌型。例如，如果一手牌是 Four of a kind，另一手牌是 Full house，那么 Four of a kind 一定胜过 Full house。

如果两手牌的牌型相同，则比较牌点大小。我们会按照上面描述的方式排列牌，然后逐张比较。先比较第一张牌，如果某手牌的第一张牌点数更大，则该手牌获胜。如果两手牌第一张牌点数相同，则比较第二张牌，依此类推。如果每个位置上的牌点数都相同，则无人获胜。牌的花色永远不影响胜负。例如，♣ 5、♢ 5、♡ 5、♠ 2、♣ 2 可以击败 ♢ 3、♠ 3、♡ 3、♢ A、♡ A。因为它们都是 Full house，所以先比较三条部分的点数。

Input

输入的第一行包含一个整数 T ($1 \le T \le 10^4$)，表示测试数据组数。对于每组测试数据：

仅有一行包含八个字符串 $c_1, c_2, c_3, c_4, p_1, p_2, p_3, p_4$，它们之间用单个空格分隔，分别表示你今晚打老虎的四张明牌，以及法国赌神皮尔克松的四张明牌。

对于每张牌，对应字符串的第一个字符表示点数。可能的点数为 `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, `T`, `J`, `Q`, `K`, `A`。其中 `T` 表示 10。第二个字符表示花色：`C` 表示梅花，`D` 表示方块，`H` 表示红桃，`S` 表示黑桃。

保证每组测试数据中每张牌最多只出现一次。

Output

对于每组测试数据，输出一行。如果你有必胜策略，输出 "`WoYaoYanPai`"（不含引号）；如果法国赌神有必胜策略，输出 "`GeiWoCaPiXie`"（不含引号）；如果双方都没有必胜策略，最终平局，输出 "`PaiMeiYouWenTi`"（不含引号）。

Example

standard input	standard output
3 AS KH KD AC AH KS KC AD 2D 3C 3D 2C AH QH JH 2H 4C 6H KH 9H 5H 6C 7H 9S	PaiMeiYouWenTi WoYaoYanPai GeiWoCaPiXie

Note

对于第一个样例，双方都可以把自己的暗牌变成 `Q`，组成 "AAKKQ" 的两对牌型，最终平局，因此双方都没有必胜策略。

对于第二个样例，你总是可以把自己的暗牌变成任意一张 `3`，组成 "33322" 的葫芦，而法国赌神最多只能组成同花，所以你有必胜策略。

对于第三个样例，法国赌神可以把自己的暗牌变成任意一张 `8`，组成 "56789" 的顺子，而你最多只能组成一对，所以法国赌神有必胜策略。

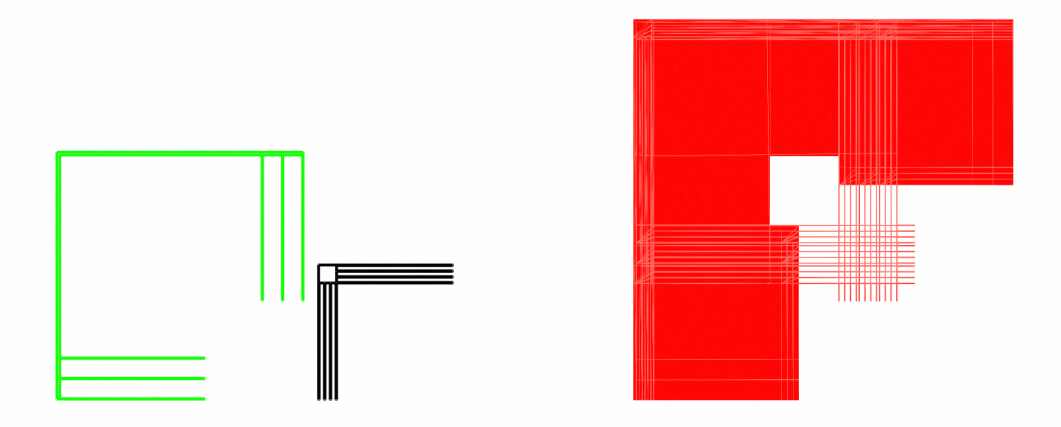
Problem K. 几何教科书

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	1024 megabytes

对于平面上的两个点集 $A, B \subseteq \mathbb{R}^2$，$A$ 和 B 的闵可夫斯基和定义为：

$$A + B = \{\, a + b \mid a \in A,\ b \in B \,\}$$

给定平面上两个简单多边形 P 和 Q，你需要求出 $P + Q$ 的面积。请注意，每个多边形均指其内部与边界所构成的闭区域（即填充后的多边形）。



Oks, E. (2005). Minkowski Sums of Simple Polygons. Tel-Aviv University Thesis, Fig. 1.2.

Input

输入的第一行包含一个整数 T $(1 \le T \le 10)$，表示测试数据组数。对于每组测试数据：

第一行包含两个整数 n 和 m $(3 \le n, m \le 30)$，分别表示两个简单多边形的顶点数量。

接下来 n 行，每行包含两个整数 x 和 y $(-100 \le x, y \le 100)$，给出 P 的顶点坐标 (x, y)。

接下来 m 行，每行包含两个整数 x 和 y $(-100 \le x, y \le 100)$，给出 Q 的顶点坐标 (x, y)。

保证多边形的顶点按逆时针顺序给出。多边形是简单的，即其顶点是不同的，并且多边形的两条边没有交叉或接触，除了相邻的边在其公共顶点处接触。此外，没有两条相邻的边是共线的。

保证所有测试数据中 n 的总和以及 m 的总和都不超过 30。

Output

对于每组测试数据，输出一行包含一个实数，表示 $P + Q$ 的面积。

如果你的答案的绝对误差或相对误差不超过 10^{-6}，则将被视为正确。形式化地，设你的输出为 a，标准答案为 b，当且仅当 $\frac{|a-b|}{\max(1,|b|)} \le 10^{-6}$ 时，你的输出会被接受。

Example

standard input	standard output
2 3 3 0 0 1 0 0 1 0 0 1 0 0 1 10 3 0 0 10 0 10 9 9 8 9 1 1 1 1 9 8 9 9 10 0 10 0 0 2 0 0 2	2.000000000000 103.500000000000

Problem L. 子串的子串

Input file:	standard input
Output file:	standard output
Time limit:	5 seconds
Memory limit:	1024 megabytes

给定一个长度为 n 的字符串 $S = s_1 s_2 \ldots s_n$ 和一个长度为 n 的序列 $a_1, a_2, \ldots, a_n$。给出 q 个询问，每次询问给定一个字符串 t:

如果一个区间 $[l, r]$ $(1 \le l \le r \le |S|)$ 满足 t 是 $S[l, r] = s_l s_{l+1} \ldots s_r$ 的子串，那么称这个区间为好区间。

求出所有好区间中 $f(l, r) = a_l + a_{l+1} + \ldots + a_r$ 的最大值，以及所有好区间的 $f(l, r)$ 的和对 $998\,244\,353$ 取模后的结果。

保证存在至少一个好区间。

Input

输入第一行两个整数 n $(1 \le n \le 10^5)$ 和 q $(1 \le q \le 3 \times 10^5)$，表示字符串和序列的长度，以及询问的个数。

输入第二行一个长度为 n 的字符串 S。

输入第三行 n 个整数 $a_1, a_2, \ldots, a_n$ $(-10^9 \le a_i \le 10^9)$，表示给定的序列。

接下来 q 行，每行包含一个字符串 t，表示一次询问，保证存在至少一个好区间。

保证 t 的总长度不超过 3×10^5，所有字符串都只由小写英文字母构成。

Output

输出 q 行，每行包含两个整数，分别表示一次询问中 $f(l, r)$ 的最大值，以及所有好区间的 $f(l, r)$ 的和对 $998\,244\,353$ 取模后的结果。

Example

standard input	standard output
8 2 abbabaab 3 -1 4 -1 5 -9 2 -6 bab aa	10 13 3 998244301

Note

对于第一组询问，能取到 $f(l, r)$ 最大值的好区间是 $[1, 5]$。

对于第二组询问，能取到 $f(l, r)$ 最大值的好区间是 $[1, 7]$。