2023 icpc 国际大学生程序设计竞赛
亚洲区域赛（南京站）

正式赛

2023 年 11 月 5 日

试题列表

A	酷，昨日四次重现
B	交并比
C	原根
D	红黑树
E	延伸距离
F	等价重写
G	背包
H	谜题：问号
I	计数器
J	后缀结构
K	华丽收场
L	电梯
M	接雨水

本试题册共 13 题，20 页。
如果您的试题册缺少页面，请立即通知志愿者。

承办方

命题方

竞赛过程中访问非竞赛网页是违反竞赛规则的行为。
如果您有兴趣（我们很荣幸），
请在竞赛后扫描二维码。

Problem A. 酷，昨日四次重现

继 2018，2019，2020，2021 和 2022 年成功承办赛事之后，南京航空航天大学（NUAA）将连续第六年承办国际大学生程序设计竞赛（ICPC）。

在 2018 与 2019 年，“中二之力”队与“三个顶俩”队为清华大学赢得了冠军。在 2020，2021 与 2022 年，北京大学的“逆十字”队赢得三连冠。今年，将会有约 330 支队伍参与南京站的竞赛。本次竞赛将会颁发至多 33 项金奖，66 项银奖与 99 项铜奖（数字仅供参考）。让我们期待选手们出色的表现！

更棒的是，因为疫情已经结束，我们终于可以相聚南京参与这场精彩的比赛。我们想要感谢竞赛组委会与志愿者们的努力付出。感谢你们为本次竞赛做出的贡献！

2018 国际大学生程序设计竞赛亚洲区域赛（南京站）

在 2018 年的竞赛中，K 题《袋鼠谜题》要求选手为以下游戏构造一个操作序列：

> 谜题由一个 n 行 m 列的网格（$1 \le n, m \le 20$）组成，且有一些（至少 2 只）袋鼠位于网格中。玩家的目标是控制袋鼠并把它们聚集在同一个格子中。一些格子里有墙，袋鼠无法进入这些有墙的格子，而其它格子是空的。袋鼠可以从一个空格子移动到上，下，左，右相邻的另一个空格子中。
>
> 游戏开始时，每个空格子里都有一只袋鼠。玩家可以通过键盘上 U，D，L，R 四个按键控制袋鼠的移动。所有袋鼠会同时根据您按下的按键移动。
>
> 选手需要构造一个长度至多为 5×10^4 且由 U，D，L，R 组成的操作序列以达成目标。

在 2020 年的竞赛中，A 题《啊，昨日重现》要求选手构造一张输入地图，以证明以下代码并不是上述问题的解：

```cpp
#include <bits/stdc++.h>
using namespace std;
string s = "UDLR";
int main()
{
  srand(time(NULL));
  for (int i = 1; i <= 50000; i++) putchar(s[rand() % 4]);
  return 0;
}
```

在 2021 年的竞赛中，A 题《呀，昨日再次重现》同样要求选手为以下游戏构造操作序列：

> 本题中，网格中的每个格子都有恰好一只袋鼠。您需要构造一个仅由字符 'U'，'D'，'L' 和 'R' 组成的操作序列。在应用该操作序列后，所有袋鼠必须聚集在指定格子 (a, b) 中。操作序列的长度不能超过 $3(n-1)$。同往常一样，所有袋鼠会根据您的命令同时移动。

在 2022 年的竞赛中，A 题《停停，昨日请不要再重现》要求选手解决以下计数问题：

> 本题中，网格中的每个格子（除了一个格子是洞）都恰好有一只袋鼠。给定操作序列，所有走出网格外或踩到洞上的袋鼠都会被移除。给定所有操作后剩余的袋鼠数量，求有几个位置可能是洞。

在 2023 年的竞赛中，袋鼠题又回来啦！我们不知道为什么命题组的成员们那么喜欢袋鼠，但题目如下：

给定一张 n 行 m 列的网格，每个格子要么是洞，要么是空地。每个空地都恰好有一只袋鼠。

相似地，袋鼠可以被键盘上的 U，D，L，R 键控制。所有袋鼠会同时根据按下的按键移动。具体来说，对于一只位于第 i 行第 j 列的格子（用 (i, j) 表示）上的袋鼠：

1. 按键 U：它会移动到 $(i-1, j)$。
2. 按键 D：它会移动到 $(i+1, j)$。
3. 按键 L：它会移动到 $(i, j-1)$。
4. 按键 R：它会移动到 $(i, j+1)$。

如果一只袋鼠踩到了洞或者移动到了网格外面，它将被从网格上移除。如果完成一系列操作后（操作序列可以为空）恰有一只袋鼠留在网格上，那么这只袋鼠就是赢家。

您需要解决的问题是：对于每只袋鼠，判断是否存在一个操作序列使得它成为赢家。输出可能成为赢家的袋鼠总数。

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据：

第一行输入两个整数 n 与 m（$1 \le n, m \le 10^3$，$1 \le n \times m \le 10^3$）表示网格的行数和列数。

对于接下来 n 行，第 i 行输入一个长度为 m 的字符串 $s_{i,1}s_{i,2}\cdots s_{i,m}$，每个字符要么是 '.'（点号，ascii: 46）要么是 'O'（大写字母，ascii: 79）。如果 $s_{i,j}$ 是 '.' 则格子 (i, j) 是空地；如果 $s_{i,j}$ 是 'O' 则格子 (i, j) 是洞。

保证所有数据 $n \times m$ 之和不超过 5×10^3。

Output

每组数据输出一行一个整数，表示共有几只袋鼠满足存在一个操作序列使得它成为赢家。

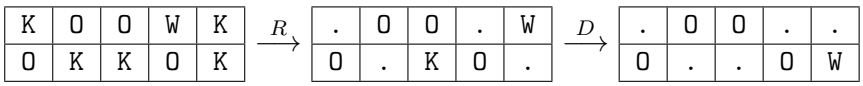
Example

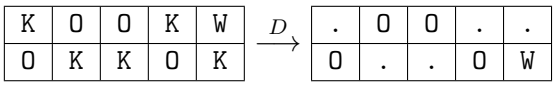
standard input	standard output
4 2 5 .OO.. O..O. 1 3 O.O 1 3 .O. 2 3 OOO OOO	3 1 0 0

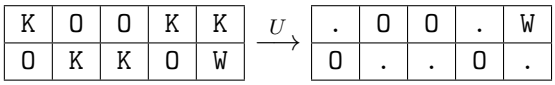
Note

样例数据解释如下。我们用 'W' 表示接下来成为赢家的袋鼠，用 'K' 表示其它袋鼠。

对于第一组样例数据，初始位于 $(1,4)$，$(1,5)$ 和 $(2,5)$ 的袋鼠可能成为赢家。以下展示可能的操作序列：







对于第二组样例数据，因为只有一只袋鼠，无需任何操作即可让它成为赢家。

对于第三组样例数据，因为任何操作都会让两只袋鼠同时被移除，所以没有任何可能的赢家。

对于第四组样例数据，因为没有袋鼠，所以没有任何可能的赢家。

Problem B. 交并比

交并比，又称雅卡尔指数或雅卡尔相似系数（法语原文 coefficient de communauté，由 Paul Jaccard 提出），是用于比较样本集的相似性与多样性的统计量。雅卡尔系数能够量度有限样本集合的相似度，其定义为两个集合交集大小与并集大小之间的比例：

$$J(A,B)=\frac{|A\cap B|}{|A\cup B|}$$

在计算机视觉中，交并比（IOU）是一个被广泛使用的统计量，用于衡量各类对象检测和分割算法。

与 x 轴和 y 轴对齐的矩形（即矩形的边与 x 轴或 y 轴平行）通常被称作“轴对齐矩形”，“轴对齐矩形边界框”（AABB），或“边界框”。另一方面，与 x 轴和 y 轴不一定对齐的矩形（即矩形的边与坐标轴可能成一个角度）通常被成为“旋转矩形”，“旋转矩形边界框”，或“方向矩形边界框”（OBB）。在计算机视觉与图像处理应用中，根据需要解决的问题，两种矩形都被广泛使用。

本题中，您需要找到一个轴对齐矩形（AABB），使得它与一个旋转矩形（OBB）的交并比最大。两个矩形之间的交并比定义为两个矩形交集的面积除以两个矩形并集的面积。

Input

有多组测试数据。第一行输入一个整数 T（$1\le T\le 10^4$）表示测试数据组数，对于每组测试数据：

第一行输入八个整数 x_1，y_1，x_2，y_2，x_3，y_3，x_4，y_4（$-10^9\le x_i,y_i\le 10^9$），其中 (x_i,y_i) 表示旋转矩形的第 i 个顶点的坐标。顶点以顺时针或逆时针顺序给定。保证旋转矩形的面积为正。

Output

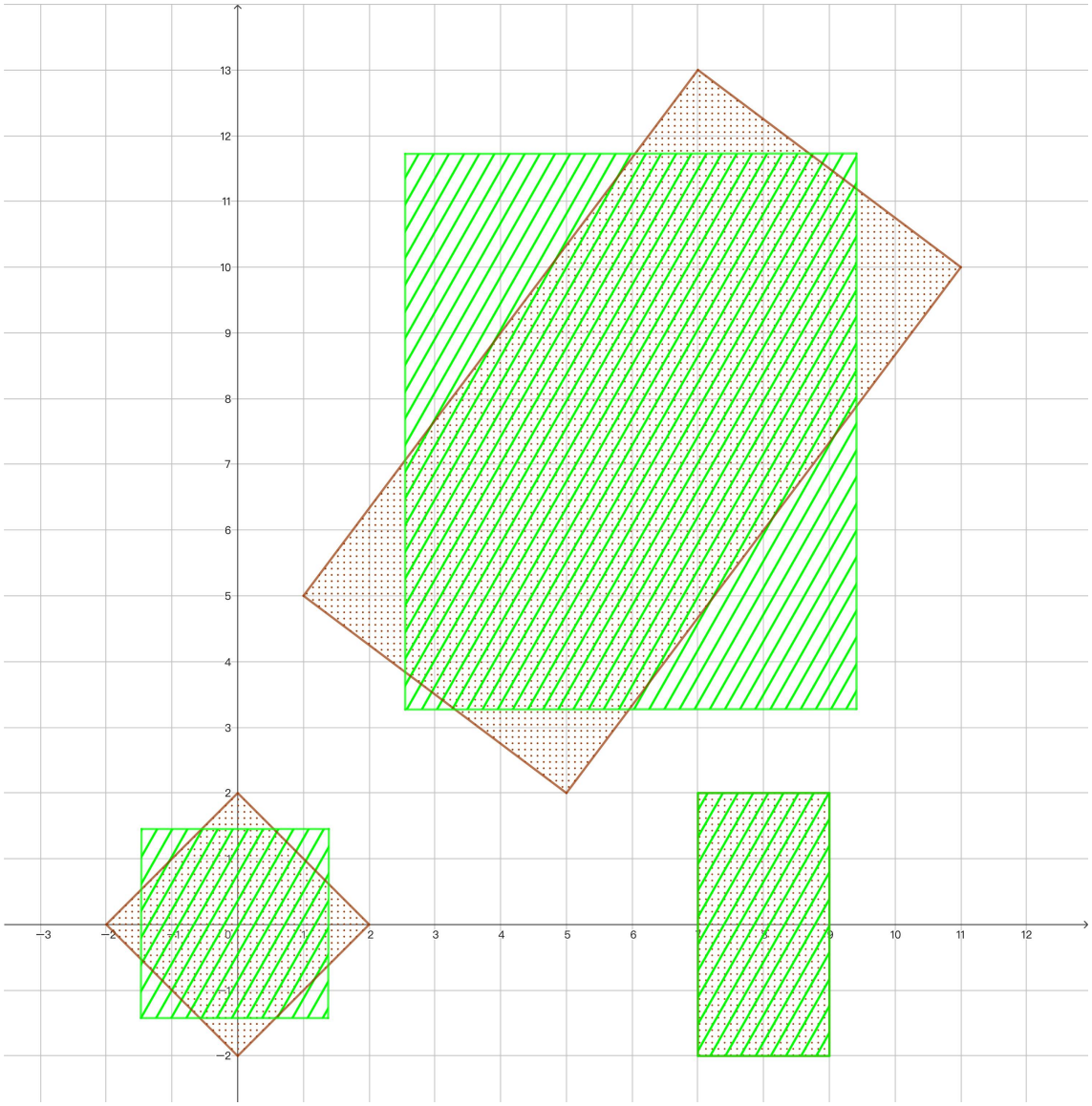
每组数据输出一行一个数表示旋转矩形和轴对齐矩形之间的最大交并比。如果相对误差或绝对误差不超过 10^{-9}，您的答案将被接受。

Example

standard input	standard output
3 0 2 2 0 0 -2 -2 0 7 -2 9 -2 9 2 7 2 7 13 11 10 5 2 1 5	0.70710678118654752 1 0.62384322483109367

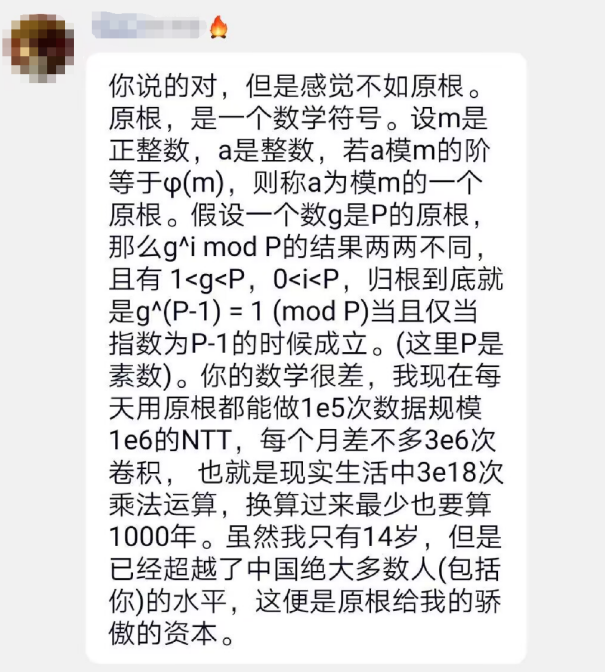
Note

样例数据解释如下。输入的矩形用点状阴影表示，最优的轴对齐矩形用斜线阴影表示。



Problem C. 原根

堡堡刚刚学会了数论中的原根。现在他正通过即时通信软件向小青鱼炫耀他的知识。



本图仅为娱乐，与题目本身无关。如果您看不懂中文，可以安全地跳过本图。

因为如果非负整数 g 是模 P 的原根（P 是质数），则 $g^{P-1} \equiv 1 \pmod{P}$，所以堡堡打算用表达式 `(g ^ (P - 1)) % P == 1` 来检查 g 是不是模 P 的原根。不幸的是，在大多数编程语言中（例如 C 和 C++），`^` 是按位异或（XOR）运算符，而不是次方运算符。小青鱼一下子就发现了这个错误，现在他开始思考起以下问题：

给定质数 P 和非负整数 m，有多少非负整数 g 满足 $g \leq m$ 且 $g \oplus (P-1) \equiv 1 \pmod{P}$？这里 $\oplus$ 是按位异或（XOR）运算符。

请帮助小青鱼解决这个问题。

Input

有多组测试数据。第一行输入一个整数 T（$1 \leq T \leq 10^5$）表示测试数据组数，对于每组测试数据：

第一行输入两个整数 P 和 m（$2 \leq P \leq 10^{18}$，$0 \leq m \leq 10^{18}$，P 是质数）。

Output

每组数据输出一行一个整数表示满足限制的 g 的数量。

Example

standard input	standard output
3 2 0 7 11 1145141 998244353	1 2 872

Problem D. 红黑树

有一棵 n 个节点的有根树，节点编号从 1 到 n，其中节点 1 为根。每个节点都有一个颜色， 要么是红色，要么是黑色。

称一个节点是好的，若每一条以该节点为起点，以该节点任意一个后代叶子节点为终点的简单路径中，都包含相同数量的黑色节点。称一棵树是完美的，若树中每个节点都是好的。

令 R_k 表示以节点 k 为根的子树。对于每个 $1 \le k \le n$，回答以下询问：如果您可以任意选择一些节点并改变它们的颜色（也就是说，把红色节点改成黑色，以及把黑色节点改成红色），至少需要选择几个节点才能让 R_k 变得完美。

请回忆：简单路径不会多次经过同一条边。

同时请回忆：以节点 k 为根的子树是一棵由节点 k 所有后代组成的有根树，并以节点 k 为根。请注意，每个节点都是它自己的后代。

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据：

第一行输入一个整数 n（$2 \le n \le 10^5$）表示树中节点的数量。

第二行输入一个长度为 n 的字符串 $s_1 s_2 \cdots s_n$（$s_i \in \{0, 1\}$）。若 $s_i = 0$ 则节点 i 是红色节点；若 $s_i = 1$ 则节点 i 是黑色节点。

第三行输入 $(n-1)$ 个整数 $p_2, p_3, \cdots, p_n$（$1 \le p_i < i$），其中 p_i 是节点 i 的父节点。

保证所有数据 n 之和不超过 10^6。

Output

每组数据输出一行 n 个由单个空格分隔的整数，其中第 i 个整数表示至少需要选择几个节点才能让 R_i 变得完美。

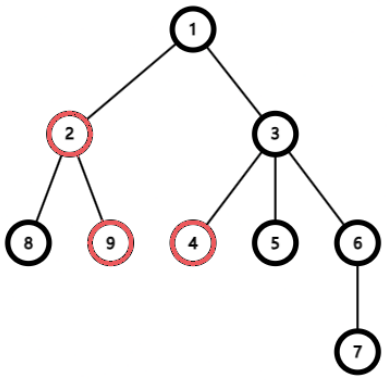
请不要在行末输出多余空格，否则您的答案可能会被认为是错误的！

Example

standard input	standard output
2 9 101011110 1 1 3 3 3 6 2 2 4 1011 1 1 3	4 1 2 0 0 0 0 0 0 2 0 0 0

Note

我们展示第一组样例数据。



为了让 R_1 变得完美，我们可以改变节点 2，4，6 和 9 的颜色。改变之后，所有从节点 1 到后代叶子节点的路径均包含 3 个黑色节点，所有从节点 2 到后代叶子节点的路径均包含 2 个黑色节点，所有从节点 3 到后代叶子节点的路径均包含 2 个黑色节点。由于节点 4 到 9 都只有一个后代叶子节点，所以它们一直都是好的。

为了让 R_2 变得完美，我们可以改变节点 8 的颜色。改变之后，所有从节点 2 到后代叶子节点的路径均包含 0 个黑色节点。由于节点 8 和 9 都只有一个后代叶子节点，所以它们一直都是好的。

Problem E. 延伸距离

有 $n \times m$ 个点排成了 n 行 m 列，相邻点之间被无向带权边连接。具体来说，令 (i,j) 表示位于第 i 行第 j 列的点，对于所有 $1 \le i,i' \le n$ 和 $1 \le j,j' \le m$，(i,j) 和 (i',j') 之间有边相连当且仅当 $|i-i'|+|j-j'|=1$。

堡堡的旅行将从第一列的任意一个点 $(p,1)$ 开始，到最后一列的任意一点 (q,m) 结束。对于每条边他都可以沿着这条边的两个方向走。一条路径的距离定义为这条路径经过的边权之和。在所有从第一列到最后一列的路径中，堡堡会选择最短的路径。

小青鱼希望堡堡享受这段旅行，于是他尝试在堡堡出发前增加一些边的权值。具体来说，小青鱼每次操作可以选择一条边，将它的权值增加 1。小青鱼希望在他修改后，堡堡旅行的距离恰好增加了 k。请帮助他求出最少需要进行多少次操作，并输出方案。

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据：

第一行输入三个整数 n，m 和 k（$1 \le n \times m \le 500$，$2 \le n,m \le 500$，$1 \le k \le 100$）表示行数，列数和最短路径距离的增加量。

对于接下来的 n 行，第 i 行输入 $(m-1)$ 个整数 $r_{i,1}, r_{i,2}, \cdots, r_{i,m-1}$（$1 \le r_{i,j} \le 10^9$），其中 $r_{i,j}$ 表示连接 (i,j) 和 $(i,j+1)$ 的边权。

对于接下来的 $(n-1)$行，第 i 行输入 m 个整数 $c_{i,1}, c_{i,2}, \cdots, c_{i,m}$（$1 \le c_{i,j} \le 10^9$），其中 $c_{i,j}$ 表示连接 (i,j) 和 $(i+1,j)$ 的边权。

保证所有数据 $n \times m$ 之和不超过 5×10^3。

Output

对于每组数据：

首先输出一行一个整数表示最少需要的操作次数。

接下来输出 n 行，第 i 行输出 $(m-1)$ 个由单个空格分隔的整数 $r'_{i,1}, r'_{i,2}, \cdots, r'_{i,m-1}$（$1 \le r'_{i,j} \le 2 \times 10^9$），其中 $r'_{i,j}$ 表示完成所有操作后连接 (i,j) 和 $(i,j+1)$ 的边权。

接下来输出 $(n-1)$ 行，第 i 行输出 m 个由单个空格分隔的整数 $c'_{i,1}, c'_{i,2}, \cdots, c'_{i,m}$（$1 \le c'_{i,j} \le 2 \times 10^9$），其中 $c'_{i,j}$ 表示完成所有操作后连接 (i,j) 和 $(i+1,j)$ 的边权。

如果有多种合法答案，输出任意一种。

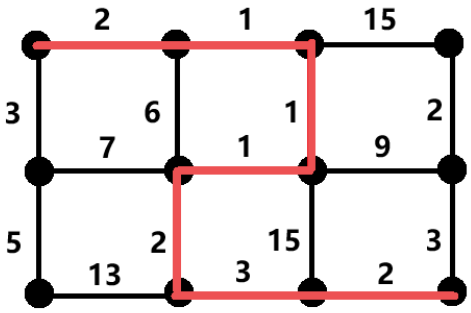
请不要在行末输出多余空格，否则您的答案可能会被认为是错误的！

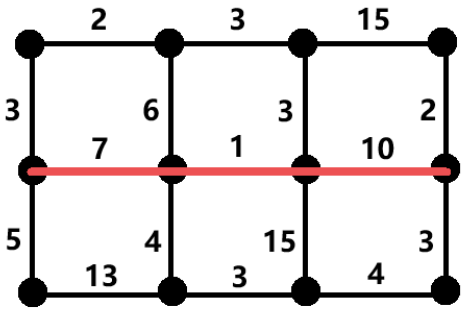
Example

standard input	standard output
2 3 4 6 2 1 15 7 1 9 13 3 2 3 6 1 2 5 2 15 3 3 3 3 1 1 2 2 3 3 1 1 1 2 2 2	9 2 3 15 7 1 10 13 3 4 3 6 3 2 5 4 15 3 4 4 1 3 2 3 3 1 1 1 2 2 2

Note

第一组样例数据如下图所示。左边的是原图，右边的是增加一些边的权值之后的图。每张图的最短路用红色线条表示。





Problem F. 等价重写

有一个长度为 m 的序列 A，所有元素均为 0。接下来我们将依次对 A 执行 n 个操作。第 i 个操作可记为 p_i 个不同的整数 $b_{i,1}, b_{i,2}, \cdots, b_{i,p_i}$，表示对于所有 $1 \le j \le p_i$，我们将把序列中第 $b_{i,j}$ 个元素的值改为 i。令 R 表示所有操作后的结果序列。

我们现在要求您重新排列这些操作，但保持最终的结果不变。更正式地，令 $q_1, q_2, \cdots, q_n$ 表示一个 n 的排列，且与 $1, 2, \cdots, n$ 不同。您将会依次对序列 A 执行第 q_1，q_2，...，q_n 个操作，最终的结果序列必须和 R 相同。您的任务就是找到这样的排列，或表明其不存在。

请回忆：一个 n 的排列是一个长度为 n 的序列，每个从 1 到 n（含两端）的整数在其中都恰好出现一次。令 $x_1, x_2, \cdots, x_n$ 和 $y_1, y_2, \cdots, y_n$ 为两个 n 的排列，我们称它们是不同的，若存在整数 k 满足 $1 \le k \le n$ 且 $x_k \ne y_k$。

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据：

第一行输入两个整数 n 和 m（$1 \le n, m \le 10^5$）表示操作的数量和序列的长度。

对于接下来 n 行，第 i 行首先输入一个整数 p_i（$1 \le p_i \le m$）表示第 i 个操作修改的元素数量。接下来输入 p_i 个不同的整数 $b_{i,1}, b_{i,2}, \cdots, b_{i,p_i}$（$1 \le b_{i,j} \le m$）表示被修改的元素下标。

保证所有数据 $(n+m)$ 之和不超过 2×10^6，且所有数据 p_i 之和不超过 10^6。

Output

对于每组测试数据：

如果存在所求的排列，首先输出一行 `Yes`。接下来在第二行输出 n 个由单个空格分隔的整数 $q_1, q_2, \cdots, q_n$ 表示答案。如果有多种合法答案，您可以输出任意一种。

如果不存在所求的排列，仅需输出一行 `No`。

请不要在行末输出多余空格，否则您的答案可能会被认为是错误的！

Example

standard input	standard output
3 3 6 3 3 1 5 2 5 3 2 2 6 2 3 3 1 3 2 2 3 1 1 3 2 2 1	Yes 3 1 2 No No

Note

对于第一组样例数据，按 $\{1, 2, 3\}$ 或 $\{3, 1, 2\}$ 的顺序执行操作，结果序列均为 $\{1, 3, 2, 0, 2, 3\}$。

Problem G. 背包

小青鱼，一位没有经验的商人，最近开了一家名叫“皇后有机珠宝”（QOJ）的店。这家珠宝店共有 n 枚宝石，其中第 i 枚售价为 w_i 元，美丽度为 v_i。进入商店之前，您准备了 W 元用来买下美丽度总和尽量高的宝石。

有趣的是，小青鱼的店今天正在促销。任何顾客都可以任选 k 枚宝石并免费获得它们。有了这样的机会，您很想知道，如果您使用最佳策略，用 W 元到底能获得美丽度总和多高的宝石。

请注意，每枚宝石独此一份，您不能多次获取同一枚宝石。另外，您无需花完所有的钱。

Input

每个测试文件仅有一组测试数据。

第一行输入三个整数 n，W 和 k（$1 \le n \le 5 \times 10^3$，$1 \le W \le 10^4$，$0 \le k \le n$），表示商店中宝石的总数，您拥有的金钱数以及您可以免费获得的宝石数量。

对于接下来 n 行，第 i 行输入两个整数 w_i 和 v_i（$1 \le w_i \le W$，$1 \le v_i \le 10^9$），表示第 i 枚宝石的售价和美丽度。

Output

输出一行一个整数表示答案。

Examples

standard input	standard output
4 10 1 9 10 10 1 3 5 5 20	35
5 13 2 5 16 5 28 7 44 8 15 8 41	129

Note

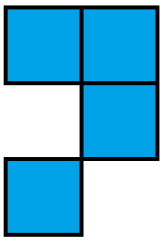
对于第一组样例数据，小青鱼的商店有 4 枚宝石，您可以免费获得其中 1 枚。一种最优策略是免费获取第一枚宝石，并购买第三和第四枚宝石。

宝石	售价 w_i	美丽度 v_i	操作
1	9	10	免费获取
2	10	1	/
3	3	5	购买
4	5	20	购买

所以答案是 $10 + 5 + 20 = 35$。

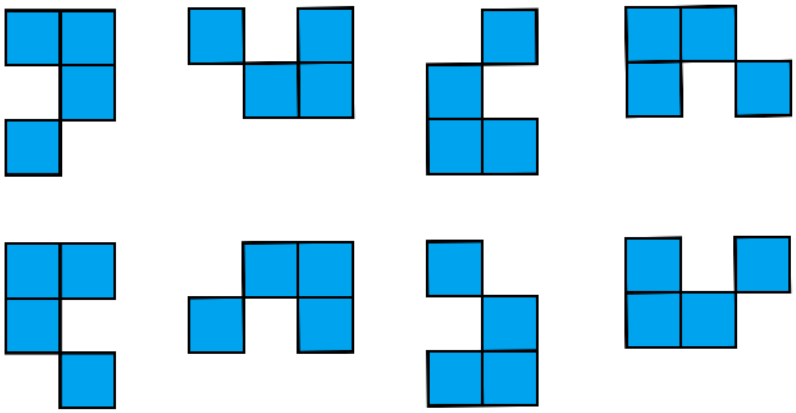
Problem H. 谜题：问号

星绘是一位谜题大师。今天，她正在玩一款名叫“问号填充”的谜题。谜题由一个 n 行 n 列的网格，以及许多问号拼图（QM 拼图）组成。一片 QM 拼图占据 4 个格子，如下图所示。



一片 QM 拼图（它看起来像一个问号，应该吧？）

整片 QM 拼图必须全部位于网格内部，拼图可以旋转 90 度的倍数或翻面。更精确地，共有 8 种 QM 拼图，如下图所示。



任意两片 QM 拼图不能占据同一个格子。谜题的目标是在 $n \times n$ 的网格中放入尽可能多的 QM 拼图。

星绘想知道您能否成功解开谜题。

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据：

第一行输入一个整数 n（$1 \le n \le 2 \times 10^3$）表示网格的大小。

保证所有数据 n^2 之和不超过 5×10^6。

Output

对于每组数据：

首先输出一行一个整数表示最多能在网格中放入几片 QM 拼图。

接下来输出 n 行。每一行包含由单个空格分隔的 n 个整数。第 i 行的第 j 个整数 $a_{i,j}$ 表示位于第 i 行第 j 列的格子属于第 $a_{i,j}$ 片 QM 拼图。如果 $a_{i,j}$ 为 0，则这个格子是空的，不属于任何 QM 拼图。

如果有多种合法答案，输出任意一种。

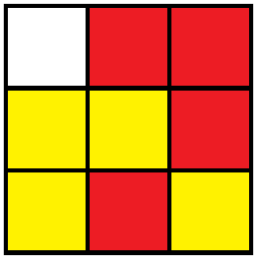
请不要在行末输出多余空格，否则您的答案可能会被认为是错误的！

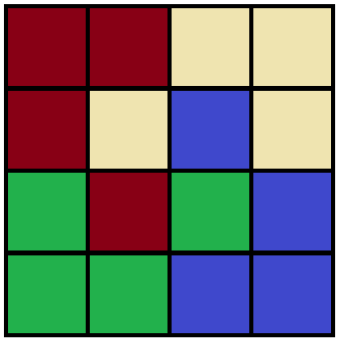
Example

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

Note

样例数据解释如下。





Problem I. 计数器

有一个计数器上有两个按钮，按下 “+” 按钮会让计数器的值增加 1，按下 “c” 按钮会让计数器的值变成 0。计数器一开始的值为 0。

某人对计数器进行了 n 次操作，每次操作是按下两个按钮中的某一个。给定 m 条已知信息，其中第 i 条已知信息可以用两个整数 a_i 和 b_i 描述，表示第 a_i 次操作后，计数器的值为 b_i。

问是否存在一种操作方式满足所有已知条件。

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据：

第一行输入两个整数 n 和 m（$1 \le n \le 10^9$，$1 \le m \le 10^5$）表示操作总数和已知信息总数。

对于接下来 m 行，第 i 行输入两个整数 a_i 和 b_i（$1 \le a_i \le n$，$0 \le b_i \le 10^9$）表示已知第 a_i 次操作后，计数器的值为 b_i。

保证所有数据 m 之和不超过 5×10^5。

Output

每组数据输出一行。若存在一种操作方式满足所有已知条件输出 `Yes`，否则输出 `No`。

Example

standard input	standard output
3 7 4 4 0 2 2 7 1 5 1 3 2 2 2 3 1 3 1 3 100	Yes No No

Note

对于第一组样例数据，按 “++cc+c+” 的顺序按下按钮即可满足所有已知条件。

对于第二组样例数据，按下 3 次按钮共有 8 种方式，如下表所述。

操作方式	第 2 次操作结果	第 3 次操作结果		操作方式	第 2 次操作结果	第 3 次操作结果
ccc	0	0		+cc	0	0
cc+	0	1		+c+	0	1
c+c	1	0		++c	2	0
c++	1	2		+++	2	3

没有任何操作方式满足所有已知条件。

对于第三组样例数据，按下 3 次按钮最多让计数器的值变成 3，不可能变成 100。

Problem J. 后缀结构

给定字符串 $u = u_1 \dots u_n$，令 $\mathrm{pre}(u, i)$ 表示前缀 $u_1 \dots u_i$。特别地，$\mathrm{pre}(u, 0)$ 是空字符串。

对于两个字符串 $u = u_1 \dots u_n$ 与 $v = v_1 \dots v_m$，令 $u + v$ 表示连接后的字符串 $u_1 \dots u_n v_1 \dots v_m$。

给定长度为 m 的字符串 $t = t_1 \dots t_m$ 和一棵有 $(n+1)$ 个节点的树 T，节点编号为 $0, 1, \dots, n$，其中节点 0 是根。每条边上都有一个字符。请注意，在本题中，字母表中可能会有多于 26 个字符。

考虑如下函数

$$f(i, j) = \max\{d(x) \mid s_x \text{ 是 } s_i + \mathrm{pre}(t, j) \text{ 的后缀}\}$$

其中 s_i 是从根到节点 i 的最短路径上所有字符连接而成的字符串，$d(i)$ 是从根到节点 i 的最短路径经过的边数。

您需要计算 $g_1, g_2, \dots, g_m$，其中 $g_j = \sum_{i=1}^{n} f(i, j)$。

请注意，s_0 是空字符串，空字符串是任何字符串的后缀。

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据：

第一行输入两个整数 n 和 m（$1 \le n, m \le 2 \times 10^5$）。

第二行输入 n 个整数 $p_1, p_2, \dots, p_n$（$0 \le p_i < i$），其中 p_i 表示节点 i 的父节点。

第三行输入 n 个整数 $c_1, c_2, \dots, c_n$（$1 \le c_i \le n$），其中 c_i 表示从节点 p_i 到节点 i 的边上的字符是字母表中第 c_i 个字符。保证对于所有 $i \ne j$，有 $p_i \ne p_j$ 或 $c_i \ne c_j$。

第四行输入 m 个整数 $t_1, t_2, \dots, t_m$（$1 \le t_i \le n$），其中 t_i 是字符串 t 中的第 i 个字符。

保证所有数据 n 之和与 m 之和均不超过 2×10^5。

Output

每组数据输出一行 m 个由单个空格分隔的整数 $g_1, g_2, \dots, g_m$。

请不要在行末输出多余空格，否则您的答案可能会被认为是错误的！

Example

standard input	standard output
2 11 3 0 1 2 0 4 5 4 6 0 9 10 1 3 2 2 1 3 4 1 3 2 1 3 2 4 5 16 0 0 0 1 4 1 2 3 2 2 2 1 3 3 2 1 3 2 1 3 2 2 1 1 2 1	17 26 22 8 5 5 5 5 5 5 5 5 5 5 5 5 5 10 5

Note

我们来计算第一组样例数据中的 $f(11,1)$ 和 $f(11,2)$ 以便您更好地理解。有 $s_{11} = \{3,2,1\}$，所以 $s_{11} + \mathrm{pre}(t,1) = \{3,2,1,3\}$。因为 $s_6 = \{2,1,3\}$ 是该字符串存在于树中的最长后缀，所以 $f(11,1) = d(6) = 3$。另外 $s_{11} + \mathrm{pre}(t,1) = \{3,2,1,3,2\}$，那么 $s_3 = \{1,3,2\}$ 是该字符串存在于树中的最长后缀，所以 $f(11,2) = d(3) = 3$。

Problem K. 华丽收场

在猪之国，《Slay the Pig》是当下最流行的肉鸽游戏。在游戏中，玩家们通过使用卡牌来对抗邪恶的土豆大魔王（Evil Potato Lord）。

游戏的主要规则如下：

1. 游戏开始时，玩家有一个起始手牌集合和一个自顶向下排列好的抽牌堆。
2. 在游戏的任意时刻，卡牌只会存在于玩家的手牌中或者抽牌堆里。
3. 玩家可以使用手牌中的卡牌，使用卡牌会先让该卡牌被丢弃，然后触发该卡牌的效果。
4. 玩家只有在上一张卡牌的所有效果都被触发完毕后，才可以使用下一张卡牌。

本题中，简单起见，我们只考虑抽牌这一种效果。 抽牌的规则如下：

1. 当使用可以抽牌的卡牌时，会按自顶向下的顺序将若干张卡牌从抽牌堆依次加入到手牌中。
2. 玩家有一个手牌数量上限 k，任意时刻玩家的手牌数量不能超过 k。
3. 当玩家试图抽牌时，如果玩家此时的手牌数量已经为 k，则不会将这张卡牌加入到手牌，而是将它直接从抽牌堆中丢弃，且不触发这张卡牌的任何效果。
4. 当玩家试图抽牌时，如果此时抽牌堆为空，则什么都不会发生。

在这个游戏中，以“华丽收场”这张卡牌为核心的卡组是最为强大的。因为一旦这张牌被使用，它会对所有敌人造成大量的伤害从而轻易地赢得游戏胜利。然而华丽收场也有着苛刻的使用条件，即使用时玩家的抽牌堆必须是空。也就是说，此时所有卡牌必须已经被使用，或被丢弃，或在玩家的手牌中。

鳖皇（Bie-Bot）是猪之国中仅次于 Mysterious Oscar 的最聪明的猪，他也在使用基于华丽收场的卡组。卡组由以下四种卡牌组成：

1. 华丽收场（Grand Finale）：游戏中最强大的卡牌，保证有且仅有一张华丽收场在鳖皇的卡组中。这张卡仅能在抽牌堆为空时被打出。
2. 快斩（Quick Slash）：使用这张卡之后可以从抽牌堆抽一张牌。
3. 后空翻（Backflip）：使用这张卡之后可以从抽牌堆抽两张牌。
4. 伤口（Wound）：一张状态牌，一旦这张牌在玩家的手牌中，就不能被使用。

在游戏开始时，鳖皇幸运地在他的起始手牌中获得了唯一一张华丽收场，并且鳖皇提前得知了他的抽牌堆自顶向下每一张牌分别是什么。现在，他的目标是成功使用华丽收场。 鳖皇想要知道，在最优策略下，达成目标所需的最小手牌数量上限 k。 作为猪之国第三聪明的玩家，您能帮帮鳖皇吗?

更正式地，给定一个长度为 n 的字符串 S_H 表示鳖皇的起始手牌，和一个长度为 m 的字符串 S_P 自顶向下地表示鳖皇的抽牌堆。 这两个字符串均由大写字母 'G', 'Q', 'B' 和 'W' 组成，分别表示起始手牌或抽牌堆对应位置的卡牌为华丽收场，快斩，后空翻和伤口。 鳖皇可以根据前文提到的规则使用这些卡牌。请输出鳖皇最终能成功使用华丽收场所需的最小手牌数量上限 k（$k \geq n$），或者声明不存在这样的 k。

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据:

第一行输入两个整数 n 和 m（$1 \leq n, m \leq 2500$）表示鳖皇的起始手牌数量以及抽牌堆的卡牌的数量。

第二行输入一个长度为 n 的字符串 S_H 表示鳖皇的起始手牌。字符串由大写字母 'G'，'Q'，'B'，'W' 组成。保证字符 'G' 仅在字符串 S_H 中出现一次。

第三行输入一个长度为 m 的字符串 S_P 自顶向下地表示鳖皇抽牌堆。字符串由大写字母 'Q'，'B'，'W' 组成。

保证所有数据 $(n+m)$ 之和不超过 5×10^4。

Output

每组数据输出一行一个整数，表示成功使用华丽收场需要的最小手牌数量上限 k（$k \geq n$）。如果无法使用华丽收场，输出 `IMPOSSIBLE`。

Example

standard input	standard output
2 2 6 BG BQWBWW 4 6 GQBW WWWWQB	3 IMPOSSIBLE

Note

以下使用"手牌/抽牌堆"字符串表示当前状况。对于第一组测试数据，一种可行的最优策略是:

1. BG/BQWBWW $\xrightarrow{B}$ BQG/WBWW
2. BQG/WBWW $\xrightarrow{Q}$ BWG/BWW
3. BWG/BWW $\xrightarrow{B}$ BWG/W（抽牌过程中会移除一个 'W'，因为此时手牌上限已满）
4. BWG/W $\xrightarrow{B}$ WWG/$\emptyset$（只会抽取一个 'W' 因为抽取第二张牌时抽牌堆为空）
5. WWG/$\emptyset$ $\xrightarrow{G}$ 成功使用华丽收场 !

Problem L. 电梯

有 n 组包裹需要被配送。第 i 组共有 c_i 个包裹，每个包裹的重量为 w_i（w_i 等于 1 或 2），并且需要被送到第 f_i 层。

有一台电梯，每趟能运送总重量不超过 k（k 是偶数）的包裹。电梯会从地面层出发，渐渐移动到这一趟所有包裹的目标楼层的最高层 h，最后返回地面层。这一趟运送将消耗 h 单位的电能。

更正式地，令 (w, f) 表示一个重量为 w，且目的地为第 f 层的包裹。一个由包裹组成的多重集合（一种可能含有重复元素的集合）$\mathbb{P}$ 能在同一趟被运送，若 $\sum\limits_{(w,f)\in\mathbb{P}} w \le k$。这一趟运送将消耗 $\max\limits_{(w,f)\in\mathbb{P}} f$ 单位的电能。

求将所有包裹运送到目的地最少一共需要多少单位的电能?

请注意，每一趟运送的包裹可以来自不同组，每一组包裹也可以分成多趟运送。您可以认为一共有 $\sum\limits_{i=1}^{n} c_i$ 个包裹需要被运送，只不过一些包裹可能有相同的重量以及相同的目的地。

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据：

第一行输入两个整数 n 与 k（$1 \le n \le 10^5$，$2 \le k \le 2 \times 10^{10}$，$k$ 是偶数）表示组数以及电梯的最大载重。

对于接下来 n 行，第 i 行输入三个整数 c_i，w_i 与 f_i（$1 \le c_i \le 10^5$，$w_i \in \{1, 2\}$，$1 \le f_i \le 10^5$）表示第 i 组包裹的数量，每个包裹的重量以及目的地。

保证所有数据中 n 之和不超过 3×10^5。

Output

每组数据输出一行一个整数，表示将所有包裹运送到目的地最少一共需要多少单位的电能。

Example

standard input	standard output
2 4 6 1 1 8 7 2 5 1 1 7 3 2 6 8 1200000 100000 1 100000 100000 1 12345 100000 2 100000 100000 2 12345 100000 1 100000 100000 1 12345 100000 2 100000 100000 2 12345	24 100000

Note

对于第一组样例数据，我们可以遵循以下策略：

- 第一趟运送包裹 $(2, 6)$，$(2, 6)$ 与 $(2, 5)$。这一趟消耗 6 单位的电能。

- 第二趟运送包裹 $(1,8)$，$(1,7)$，$(2,6)$ 与 $(2,5)$。这一趟消耗 8 单位的电能。
- 第三趟运送包裹 $(2,5)$，$(2,5)$ 与 $(2,5)$。这一趟消耗 5 单位的电能。
- 第四趟运送包裹 $(2,5)$ 与 $(2,5)$。这一趟消耗 5 单位的电能。

一共需要 $6+8+5+5=24$ 单位的电能。可以证明这是最少一共需要的电能。

对于第二组样例数据，所有包裹可以在同一趟被运送。

Problem M. 接雨水

有一张由长度为 n 的序列 $a_1, a_2, \cdots, a_n$ 表示的柱状图。柱状图上从左到右第 i 根柱子的高度为 a_i，宽度为 1。

我们会对该柱状图进行 q 次修改。第 i 次修改可以记为一对整数 (x_i, v_i)，表示我们会将第 x_i 根柱子的高度增加 v_i。

在每次修改之后，回答以下询问：如果下了一场大雨，雨水填满了柱状图上的每个坑洼，求这张柱状图中可以留存多少雨水。

更正式地，给定长度为 n 的整数序列 $a_1, a_2, \cdots, a_n$，第 i 次修改会将 a_{x_i} 的值增加 v_i。在每次修改之后，回答以下询问：令 $f_i = \max(a_1, a_2, \cdots, a_i)$ 以及 $g_i = \max(a_i, a_{i+1}, \cdots, a_n)$，计算

$$\sum_{i=1}^{n} (\min(f_i, g_i) - a_i)$$

Input

有多组测试数据。第一行输入一个整数 T 表示测试数据组数，对于每组测试数据：

第一行输入一个整数 n（$1 \le n \le 10^5$）表示柱状图中柱子的数量。

第二行输入 n 个整数 $a_1, a_2, \cdots, a_n$（$1 \le a_i \le 10^6$），其中 a_i 表示第 i 根柱子的初始高度。

第三行输入一个整数 q（$1 \le q \le 10^5$）表示修改的次数。

对于接下来 q 行，第 i 行输入两个整数 x_i 和 v_i（$1 \le x_i \le n$，$1 \le v_i \le 10^6$）表示第 i 次修改将第 x_i 根柱子的高度增加了 v_i。

保证所有数据 n 之和与 q 之和均不超过 10^6。

Output

每次修改输出一行一个整数表示柱状图中可以留存多少雨水。

Example

standard input	standard output
2 6 1 2 3 4 5 6 2 1 2 3 3 5 100 10 1 10 100 1 3 100	1 4 180