Problem A. 逆序对染色

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

我们知道逆序对的定义：对于一个长度为 n 的排列 $a_1, a_2, \ldots, a_n$，如果有 $i < j$ 且 $a_i > a_j$，那么称 a_i, a_j 为一组逆序对。一个长度为 n 的排列满足 1 到 n 恰好各出现一次。

求逆序对的个数什么的对你来说应该不是什么难题了。现在给你一个长度为 n 的排列 $a_1, a_2, \ldots, a_n$，对于每一对满足 $i < j$ 且 $a_i > a_j$ 的逆序对，在大小为 $n \times n$ 的网格图上把第 i 行第 a_j 列的格子染色。请问最终的网格图上染色的连通块有几个？连通是指，如果两个被染色的格子有一条公共边，那么我们认为这两个格子是连通的。

Input

第一行一个整数 n（$1 \le n \le 3 \times 10^5$），表示排列的长度。

第二行 n 个整数 $a_1, a_2, \ldots, a_n$，表示一个长度为 n 的排列。

Output

输出一个整数，表示最终的网格图上的连通块的个数。

Examples

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

Problem B. 连接召唤

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

你现在身处一片魔法大陆，并且你身边现在有着五种不同种类的精灵，第 i 种精灵的能力值是 i。

这时有个精灵召唤师带着一只能力值为 6 的精灵经过。这个能力值为 6 的精灵看起来十分强大，因此你连忙询问他如何获得这种精灵并从他那获得了一种名为连接召唤的方法。

连接召唤：每次选择一些能力值为 1 到 5 的精灵，对于每只精灵，你可以选择赋权为 1 或者赋权为 x（x 为其能力值），然后得到**一只**能力值为这些精灵赋权和的精灵，但由于魔力限制，赋权和不能超过 6。

你现在想知道自己手上的这些精灵能够连接召唤出多少只能力值为 6 的精灵。

Input

第一行一个整数 T（$1 \le T \le 10^5$），表示数据组数。

接下来 T 行，每行五个整数 a_i（$0 \le a_i \le 10^9, \sum a_i \le 10^9$），表示你现在具有的能力值为 i 的精灵数量。

Output

对于每组数据，输出一行一个整数，表示你能通过连接召唤得到能力值为 6 的精灵的最大数目。

Example

standard input	standard output
5 3 3 3 3 3 2 3 4 5 1 1 2 3 4 5 2 2 0 0 0 0 3 0 0 3	7 7 7 1 3

Problem C. 黑白立方格

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

给定一个大小为 $N \times M \times L$ 的立方格，其中每个格点 (i, j, k) 要么是白色要么是黑色，格点的坐标均为正整数。设翻转格点 (i, j, k) 颜色的代价为 $a_{i,j,k}$。求翻转格点颜色的最小代价，使得左下角 $(1, 1, 1)$ 为黑色，右上角 (N, M, L) 为白色，且任意两个不同格点 (i_1, j_1, k_1) 和 (i_2, j_2, k_2) 至少满足以下条件之一。

- $i_1 > i_2$
- $j_1 > j_2$
- $k_1 > k_2$
- (i_1, j_1, k_1) 是黑色的。
- (i_2, j_2, k_2) 是白色的。

Input

第一行包含三个整数 N, M, L（$1 \le N, M, L \le 5 \times 10^3$，$2 \le N \times M \times L \le 5 \times 10^3$），表示网格的大小。

接下来 $L \times N$ 行中，每行包含一个长度为 M 的字符串，该字符串只由 `B` 和 `W` 组成。第 $(k-1) \times N + i$ 行的第 j 个字符是格点 (i, j, k) 的初始颜色，如果这个字符为 `B`，则表示格点颜色为黑色，否则为白色。

接下来 $L \times N$ 行中，每行包含 M 个不大于 10^5 的非负整数。第 $(k-1) \times N + i$ 行的第 j 个整数表示翻转格点 (i, j, k) 颜色的代价。

Output

输出一行一个整数，表示最小代价。

Example

standard input	standard output
2 2 2 WW WW BB BB 1 1 1 1 2 2 2 2	5

Problem D. L 型覆盖

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

给定一个 n 行 m 列的网格，行按从上到下的顺序从 1 到 n 编号，列按从左到右的顺序从 1 到 m 编号。现在有任意多个 1×1 L 型（如下图所示），你需要使用这些 L 型覆盖网格，使得只有第 1 行第 m 列的单元格（位于右上角的单元格）没有被覆盖，且其他单元格仅被恰好一个 L 型覆盖。你需要判断是否可以进行满足条件的覆盖，如果可以，输出一种覆盖方案。

Input

第一行一个整数 T（$1 \le T \le 10^4$），表示测试数据组数。

对于每组数据，一行两个整数 n, m（$2 \le n, m \le 500$），表示要覆盖的网格大小。

保证所有测试数据 $n \times m$ 之和不超过 10^6。

Output

对于每组数据，如果无法进行满足条件的覆盖，输出一行 `No`。

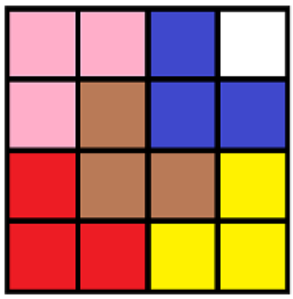
否则，先输出一行 `Yes`，再输出 n 行，每行一个长为 m 的字符串，表示一种覆盖方案。字符串中仅包含 `UDLRC.` 六种字符，第 i 行第 j 个字符表示第 i 行第 j 列网格的覆盖情况。字符 `.` 表示单元格没有被覆盖，对于输出的覆盖方案，应仅有一个 `.` 且位于第 1 行第 m 列。字符 `C` 表示 L 型的中心（即题目描述中的 L 型的左下角）。字符 `UDLR` 分别表示这个单元格的上、下、左、右单元格是覆盖这一单元格的 L 型的中心，你应保证除右上角单元格外，每个单元格仅被一个 L 型覆盖。

Example

standard input	standard output
2 4 4 2 3	Yes CLD. UDCL DCLD CLRC No

Note

对于第一组数据，一种覆盖方法如下图。



Problem E. L 型覆盖检查器

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

本题中，你需要实现上一题的输出检查器。

给定一个 n 行 m 列的网格，行按从上到下的顺序从 1 到 n 编号，列按从左到右的顺序从 1 到 m 编号。现在有任意多个 1×1 L 型（如下图所示），你已经使用了一些 L 型覆盖网格，但你不知道你是否正确地覆盖了网格。

你的覆盖方案可以表示为 n 行长度为 m 的字符串。字符串中仅包含 `UDLRC.` 六种字符，第 i 行第 j 个字符表示第 i 行第 j 列网格的覆盖情况。字符 `.` 表示单元格没有被覆盖，如果覆盖方案中只有一个 `.`，且这个 `.` 位于第 1 行第 m 列，则这个覆盖方案是正确的，否则是错误的。字符 `C` 表示 L 型的中心（即上图中的 L 型的左下角）。字符 `UDLR` 分别表示这个单元格的**上**、**下**、**左**、**右**单元格是覆盖这一单元格的 L 型的中心。如果所有 L 型都是完整的，且每个单元格都仅被一个 L 型覆盖，则这个覆盖方案是正确的，否则是错误的。

给出一些覆盖方案，请判断这些方案是否是正确的。

Input

第一行一个整数 T（$1 \le T \le 10^4$），表示测试数据组数。

对于每组数据，第一行两个整数 n, m（$2 \le n, m \le 500$），表示网格大小。

接下来 n 行，每行一个长为 m 的字符串，字符串仅由 `UDLRC.` 六种字符组成。第 i 行字符串的第 j 个字符表示网格中第 i 行第 j 列的覆盖情况。

保证所有测试数据 $n \times m$ 之和不超过 10^6。

Output

对于每组数据，如果覆盖方案正确，输出一行 `Yes`，否则输出 `No`。

Example

standard input	standard output
2 4 4 CLD. UDCL DCLD CLRC 2 3 DRC CLU	Yes No

Problem F. 小球进洞：平面版

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

小球进洞（*Isoball*）是一款富有挑战性的趣味游戏，本题中我们考虑这款游戏的平面版。给定一个圆的半径和初始位置，平面上有一个矩形，你的目标是让这个圆位于矩形内部。为此，你确定了这个圆的移动方向（用一个向量表示）。现在你想知道，如果圆按这个方向移动，是否存在一个时刻（包括最初）可以达成目标。

对于一个圆心在 (p, q)，半径为 r 的圆，称其位于左下角为 (l_x, l_y)，右上角为 (r_x, r_y) 的矩形内部，当且仅当 $\forall p \in \{(x, y) \mid (x-p)^2 + (y-q)^2 \le r^2\}$，都有 $l_x \le p_x \le r_x$ 且 $l_y \le p_y \le r_y$，其中 p_x, p_y 分别为点 p 的横纵坐标。

Input

第一行一个整数 T（$1 \le T \le 10^4$），表示测试数据组数。

对于每组数据，第一行五个整数 x, y, r, v_x, v_y（$-10^6 \le x, y, v_x, v_y \le 10^6$，$1 \le r \le 10^6$），分别表示初始时圆心的横纵坐标，圆的半径和移动方向。保证 v_x 和 v_y 不同时为 0。

第二行四个整数 l_x, l_y, r_x, r_y（$-10^6 \le l_x, l_y, r_x, r_y \le 10^6$），分别表示矩形左下角的横纵坐标和右上角的横纵坐标。保证 $l_x < r_x$ 且 $l_y < r_y$。

Output

对于每组数据输出一行，如果可以达成目标，输出 `Yes`，否则输出 `No`。

Example

standard input	standard output
5 0 0 1 1 0 2 -2 6 2 0 0 1 1 0 2 0 6 2 0 0 1 1 1 1 1 3 3 0 0 1 -1 -1 1 1 3 3 0 0 1 -1 1 -5 -5 5 5	Yes No Yes No Yes

Problem G. 函数查询

Input file:	standard input
Output file:	standard output
Time limit:	2 seconds
Memory limit:	256 megabytes

给定 n 个整数 $x_1, x_2, \ldots, x_n$。现有 q 次询问，每次询问给定一个函数 $f(x) = (a \oplus x) - b$，请判断是否存在 $1 \le i < n$ 满足 $f(x_i) \cdot f(x_{i+1}) \le 0$，如果存在请输出一个满足条件的 i，否则输出 -1。

$a \oplus b$ 表示 a 异或 b。

Input

第一行两个整数 n, q（$2 \le n \le 3 \cdot 10^5$，$1 \le q \le 3 \cdot 10^5$）。

第二行 n 个整数 $x_1, x_2, \ldots, x_n$（$0 \le x_i \le 10^9$）。

接下来 q 行，每行两个整数 a, b（$0 \le a, b \le 10^9$）。

Output

输出 q 行，表示对每个询问的回答。

Example

standard input	standard output
5 6 3 5 1 2 4 0 2 1 1 2 3 3 2 4 2 5 8	2 3 2 1 4 -1

Problem H. GG 和 YY 的石子游戏

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

GG 和 YY 正在用一堆 n 个石子玩游戏。GG 和 YY 轮流操作，GG 先操作。在每一轮中，玩家可以从石子堆中移走 1 或 2 个石子。不能操作的一方输。两位玩家都想获胜，并得到尽可能多的石子。假设 GG 和 YY 都采用最优策略。请确定胜者，并回答胜者最后获得的石子数 v。

Input

第一行一个整数 T（$1 \le T \le 10^4$），表示测试数据组数。

对于每组数据，每行一个整数 n（$1 \le n \le 10^{12}$），表示石子数量。

Output

对于每组数据，如果 GG 获胜，输出 "0 v"，否则输出 "1 v"（不包含引号）。

Example

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

Problem I. 集装箱调度

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

在一个繁忙的港口，每天有大量的集装箱需要堆放到货船的甲板上。为了方便起见，你可以将甲板和集装箱都视作边平行于坐标轴的矩形。甲板是一个左下角为 $(0,0)$ 右上角为 (l,h) 的大矩形区域。

你是集装箱的调度员，需要安排每个集装箱放到甲板上的位置。理论上讲，你需要合理调度使得空间利用率最大化。但你觉得你工资拿得太少了，于是在调度方法上偷懒。你给出的调度方法是:

- 由于限高，集装箱只能摆一层，之间不能有重叠。
- 由于你没有经费买可以旋转的吊臂，集装箱在放置时不能旋转。
- 集装箱遵循先来后到的顺序，即排在前面的集装箱先放，排在后面的集装箱后放，不改变集装箱的顺序。如果过程中有集装箱放不下，就直接把它扔了。
- 每次选择一个集装箱放置时，优先选择能放下它的位置中左下角 x 坐标最小的。如果有多个这样的候选位置，则选择 y 坐标最小的。

一共有 n 个集装箱，按顺序给出每个集装箱 x 轴方向的长度和 y 轴方向的宽度，请你判断每个集装箱能否放下，如果能则给出其左下角的位置。

Input

第一行三个整数 n,l,h（$1 \le n \le 50, 1 \le l,h \le 10^9$），分别表示集装箱的个数以及甲板区域的长度和宽度。

接下来 n 行每行两个整数 x,y（$1 \le x,y \le 10^9$），表示每个集装箱 x 轴方向的长度和 y 轴方向的宽度。

Output

按顺序每行对应一个集装箱，如果该集装箱能放下，输出空格分隔的两个整数，表示其左下角的位置坐标。否则，输出 -1。

Example

standard input	standard output
4 10 10 5 5 6 6 4 7 10 2	0 0 -1 5 0 0 7

Problem J. 罗马数字

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

罗马数字中有七个字符 I，V，X，L，C，D，M，分别表示 1，5，10，50，100，500，1000。

现在你将使用罗马数字与阿拉伯数字，在十进制下混合表示一个正整数，例如 X2 $= 10 \times 10^1 + 2 \times 10^0 = 102$，IV $= 1 \times 10^1 + 5 \times 10^0 = 15$。

对于每种数字，在表示中使用一次有一定的花费。求一个正整数使用用混合表示时的最小花费。

Input

第一行一个正整数 T（$1 \le T \le 2 \cdot 10^3$），表示数据组数。

对于每组数据，第一行一个正整数 n（$1 \le n \le 10^{18}$），表示要表示的正整数。

第二行 10 个正整数 $c_0, c_1, \ldots, c_9$（$1 \le c_i \le 10^7$），分别表示在混合表示中使用一次阿拉伯数字 0 到 9 的花费。

第三行 7 个正整数 $c_{\mathtt{I}}, c_{\mathtt{V}}, \ldots, c_{\mathtt{M}}$（$1 \le c_i \le 10^7$），分别表示在混合表示中使用一次罗马数字 I 到 M 的花费。

Output

对于每组数据输出一行一个整数，表示在混合使用罗马数字与阿拉伯数字的情况下，使用十进制表示这个正整数所需的最小花费。

Example

standard input	standard output
5 102 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 112 1 5 5 1 1 1 1 1 1 1 1 1 1 1 1 1 1 150 1 1 1 1 1 1 1 1 1 1 5 5 5 5 5 5 5 114514 10 5 5 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1919810 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	2 7 3 6 6

Note

102 可以用 X2，I02 和 102 表示，其中花费最小的表示为 X2，所需花费为 2。

112 可以用 I12，1I2，II2 和 112 表示，其中花费最小的表示为 II2，所需花费为 7。

150 可以用 XL，5C，10L 等表示，其中花费最小的表示为 150，所需花费为 3。

Problem K. 元素反应

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

你现在有 m 种元素，第一种元素的名称为 a，第二种为 b，以此类推。你将使用这些元素来打怪。

具体的，你有一个长度为 n 的元素使用序列，你会依次使用这些元素攻击怪物。假设当前元素种类为 x，如果怪物身上没有挂上元素，那么将会给怪物挂上元素 x；如果怪物身上已经有某种元素 y，那么两种元素会发生反应，产生 $a_{y,x}$ 的伤害，并且反应之后怪物身上只剩下元素 x，同类元素间也会进行反应。

但是现在某些种类的元素可能有了惰性，即这些元素攻击怪物时不会产生任何反应，也不会挂在怪物身上，会被直接忽略掉。

但你并不知道哪些种类元素有惰性，所以你想知道对于每种元素是否有惰性的 2^m 种情况，分别求出这时候元素序列造成的伤害之和。

Input

第一行输入两个整数 m, n（$1 \le n \le 10^5, 1 \le m \le 17$）。

接下来 m 行每行 m 个整数，其中第 i 行第 j 个数表示第 i 种元素在前，第 j 种元素在后时的反应伤害 $a_{i,j}$（$0 \le a_{i,j} \le 10^8$）。

接下来一行长度为 n 的字符串，表示给定的元素序列。保证字符串中只出现前 m 个小写英文字母。

Output

输出一行 2^m 个整数，其中第 i 个数表示，将所有元素惰性视为 1，非惰性设为 0，按照元素种类编号从小到大在二进制位从低到高排列得到的二进制数为 $i-1$ 时元素序列造成的伤害之和。

Examples

standard input	standard output
3 4 1 2 3 4 5 6 7 8 9 abca	15 6 10 0 6 0 1 0
3 10 1 2 3 4 5 6 7 8 9 acbabccbac	47 42 32 27 17 10 2 0

Problem L. 毛肚下清汤?

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

九宫格学校的同学现在正在为吃火锅时什么菜能下什么锅展开激烈讨论。

现在有红汤和清汤两种锅底。有 n 种菜，编号为 1 到 n。第 i 种菜在红汤和清汤中煮熟的时间分别为 a_i, b_i（保证所有 a_i，b_i 均两两不同）。根据讨论结果，他们规定了这些菜能否下在红汤与清汤中。

九宫格学校的同学吃火锅的方式比较特别。他们会先等到所有 n 种菜都上齐，然后决定每种菜要放到红汤锅底中还是清汤锅底中，如果这种菜既可以放在红汤锅底，又可以放在清汤锅底，他们会选择将这种菜放在煮熟时间最短的锅底中。讨论完后，将所有要放到红汤锅底的菜一起全部下入红汤锅，同时将所有要放到清汤锅底的菜一起全部下入清汤锅。由于讨论过程消耗了大量的体力，等到某种菜煮熟时，他们会立刻将这种菜全部捞出并吃掉。

九宫格学校的同学想知道从红汤和清汤中的捞出来的菜分别是什么，请你按捞出时间先后输出答案。

Input

第一行一个整数 n（$2 \le n \le 10^5$），表示菜的种类数。

接下来 n 行每行四个整数，分别为在红汤中煮熟的时间 a_i，在清汤中煮熟的时间 b_i，能否煮在红汤中 c_i，能否煮在清汤中 d_i。其中 $1 \le a_i, b_i \le 2 \times n$，$c_i, d_i \in \{0, 1\}$。保证每个菜至少可以煮在一个锅中，且对于任意 $1 \le i, j \le n$，保证 $a_i \ne a_j$，$b_i \ne b_j$ 且 $a_i \ne b_j$。

Output

输出共两行。

第一行先输出一个整数 k，表示煮在红汤中的菜数目，接下来 k 个整数，表示红汤中按煮熟时间排序的菜的编号，用空格分隔。

第二行先输出一个整数 c，表示煮在清汤中的菜数目，接下来 c 个整数，表示清汤中按煮熟时间排序的菜的编号，用空格分隔。

Example

standard input	standard output
8 9 11 0 1 1 8 0 1 15 7 1 0 3 13 1 1 6 12 0 1 16 5 1 0 4 2 1 0 10 14 1 0	5 4 7 8 3 6 3 2 1 5