Problem A. 时间复杂度

输入文件: standard input
输出文件: standard output
时间限制: 1 second
空间限制: 256 megabytes

在比赛中你可能会遇见复杂度上的问题，例如一个算法程序运行时间过长，所以在实现算法前，我们会估算算法的时间复杂度。

关于复杂度，前人有很丰富的理论，这里我们只用最简洁的方式介绍时间复杂度。同一个算法在不同的计算机上运行的速度会有一定的差别，并且实际运行速度难以在理论上进行计算，实际去测量又比较麻烦，所以我们通常考虑的不是算法运行的实际用时，而是算法运行所需要进行的基本操作的数量。

影响时间复杂度的最主要因素是循环，我们通常可以通过估算指令循环进行的次数来得到一种时间复杂度。如以下程序

```
int k=0;
for (int i=0;i<N;++i)
    for (int j=0;j<M;++j)
        ++k;
```

根据乘法原理，其中 $++k$ 的指令被执行了 $N*M$ 次，我们用 $O(NM)$ 表示其时间复杂度，其中 $O(x)$ 表示近似。

通常来说，如果时间复杂度达到了 $O(10^8)$ 则实际评测程序会运行约 $1s$ 。本次新手赛的题目大多是以 $1s$ 为时间复杂度上限，如果你的算法达到了 $O(10^8)$ 则仍有机会通过，但是如果达到了 $O(10^9)$ 及以上则估计要考虑重新设计算法或者改善常数部分了。

注意数据的输入输出会消耗比较大的时间，例如 C 语言中使用 scanf 能读入约 10^6 个整数，printf 类似。

C++ 的 cin/cout 与 Python 的 input/print 开销会更大，所以如果这些方法输入可能会超时，则请转用 C 语言的 scanf 输入。

Input

本题没有输入数据。

本题需要你阅读下列代码，在已知变量 n 的范围为 $[1, 10^5]$ 后，判断哪些代码不会超出 $1s$ 的时间复杂度限制。

```
//ID of this code is A
int k=0;
for (int i=1;i<n;++i)
    for (int j=i+1;j<=n;++j)
        ++k;
```

```
//ID of this code is B
for (int i=1;i<n;++i)
```

```
    for (int j=1;j*j<=n;++j)
        ++k;
```

```
//ID of this code is C
int k=0;
for (int i=1;i<n;++i)
    for (int j=1;i*j<=n;++j)
        ++k;
```

```
//ID of this code is D
for (int i=1;i<n;++i) {
    int j=i;
    while (j>0) {
        if (j%2==0) j=j/2;
        else j-=1;
    }
}
```

```
//ID of this code is E
long long fib(int n) {
    if(n <= 1) return 1;
    return fib(n - 1) + fib(n - 2);
}

int main() {
    scanf("%d", &n);
    printf("%lld", fib(n));
}
```

Output

输出一行字符串，按编号顺序升序输出不会超时的代码的编号。例如如果你认为编号 A,B,D 不会超时，则输出

```
ABD
```

其中编号为大写字母，编号间没有用来间隔的字符，字符串前后没有空格。

Problem B. 寻觅

输入文件: standard input
输出文件: standard output
时间限制: 1 second
空间限制: 128 megabytes

慧青学者协会 (Savvy Young Scholar Union) 聚集了一群朝气蓬勃的青年学者。他们经常举行集会讨论前沿理论问题。下周二是协会的创办 001 周年纪念日，为此协会理事会决定举办盛大的庆典。

理事长分配给秘书小嵩一个任务：给协会成员发送庆典的邀请函。在编写好得体的文书和导入协会名录后，打印机被小嵩一键启动，一张张邀请函被制作出来。然而，不同寻常的是，慧青学者协会的打印机解锁了一定程度的自我意识，它会把正常的数字编号加密成一串长度为 n 的英文字符串 S。每次正常编号加 1，对这个字符串的影响是修改某个特定位置的字符 (当然也有可能全串维持不变)。

协会总共有 $m+1$ 个成员，故总共制作了 $m+1$ 张邀请函。小嵩观察到了第 1 张邀请函的加密串为 S_1。之后，小嵩又发现第 $i+1$ 张邀请函对应的加密串 S_{i+1} 是将 S_i 中第 x_i 个字符改成 ch_i。

收到邀请函的成员十分兴奋，聪明的他/她一下子就注意到了那串神奇的字符串。出于强烈的集体认同感，收到邀请函的那一刻，成员就会数邀请函上的加密串 S 有多少个子串为 "sysu"，并在协会群发送自己的伟大发现。

聪明的你也是协会的一员，请试着预测所有人的消息吧。

Input

第一行包含两个整数 $n(1 \le n \le 10^6)$ 和 $m(1 \le m \le 10^4)$，分别表示字符串 S 的长度和协会成员数 -1 。

第二行给出一个长度为 n 的字符串 S_1，其中字符均为小写英文字母。

接下来的 m 行，每 i 行包含一个整数 $x_i(1 \le x_i \le n)$ 和一个字符 ch_i(保证为小写英文字母) ，描述 S_i 到 S_{i+1} 的修改。

Output

输出 $m+1$ 行，每行一个整数，表示收到第 i 张邀请函的成员统计子串 "sysu" 的出现次数的结果。

Example

standard input	standard output
6 5 sysusy 3 c 3 s 4 y 6 u 3 s	1 0 1 0 1 1

Note

子串：串中任意个**连续**的字符组成的子序列称为该串的子串。

Problem C. 声望

输入文件: standard input
输出文件: standard output
时间限制: 1 second
空间限制: 256 megabytes

单鸭山大学在推进其发展战略中，规划了一系列重要项目，旨在提升学校在不同学术领域的国际声望。每个项目完成后，会对特定领域带来直接的声望提升。目前，学校已经明确了所有项目完成后各学术领域最终期望达到的声望值。

已知单鸭山大学共有 n 个待推进项目，涉及 m 个不同的学术领域。对于每个学术领域 j（$1 \leq j \leq m$ ），有一个目标声望值 T_j，表示该领域在所有项目完成后期望达到的声望水平。

每个项目 i（$1 \leq i \leq n$）具有一个长度为 m 的声望增量向量 $P_i = [p_{i,1}, p_{i,2}, ..., p_{i,m}]$，表示项目 i 完成后对 m 个学术领域各自带来的声望提升值。项目之间没有特定的优先级或权重，只需按照某种顺序依次推进，使得所有项目的声望增量之和能使所有学术领域达到其目标声望值。

现在要求计算在满足所有学术领域最终声望目标的前提下（要严格等于目标声望，不能多，也不能少），单鸭山大学可以采用多少种不同的项目推进方案。请注意，两种方案相同当且仅当两个方案推进相同的项目。

Input

第一行两个正整数 n, m（$n \leq 20, m \leq 100$）。

第二行包含 m 个整数，第 j 个整数 T_j（$0 \leq j \leq m$），表示第 j 个学术领域期望达到的最终声望值。

接下来 n 行，每行包含 m 个整数。第 i 行的第 j 个整数 $p_{i,j}$（$0 \leq p_{i,j} \leq 100$），表示项目 i 完成后对第 j 个学术领域带来的声望提升值。

Output

输出一个整数，表示单鸭山大学在满足所有学术领域声望目标的前提下，可采用的不同项目推进方案数。

Example

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

Note

一种方案是推进项目 1,2; 另一种方案是推进项目 3. 两种方案都恰好使得最终的声望向量为 [5, 2, 3]

Problem D. 区区整除

输入文件: standard input
输出文件: standard output
时间限制: 1 second
空间限制: 16 megabytes

小 S 和小 Y 是情同手足的好哥们，经常互相考验对方问题。当对方回答不上来时，他们就会有强烈的优越感。今天的数学课上，老师强哥在黑板上写上，“x 整除 y，记作 $x|y$，等价于 $\exists k \in \mathbb{Z}, y = k \cdot x$ ……”

下课后，小 S 得意地走到小 Y 的书桌边，拍了拍小 Y 肩膀，“刚刚上课的时候, 我想到了一道关于整除的题，你肯定不会做。”

小 Y 登时站了起来，插起了腰，怒目圆睁, “嘿，区区整除，难不倒我!!! 你放马过来。”

小 S 从裤兜掏出一张皱巴巴的草稿纸，其上赫然写道：有三个正整数 x, y, z，满足 $\begin{cases} x > y > z \\ x|y+z \\ y|x+z \\ z|x+y \end{cases}$，

若已知 $d = x - z$ ，请据此构造合法的 x, y, z 或指出不存在方案。

此时的小 Y 汗流浃背了，赶紧想办法推脱一下，“我上个厕所应该就想出来了。你别得意的太早。”

在去厕所的路上，小 Y 撞见了你，马上像抓到救命稻草一般，一五一十地告诉了你题目，请求聪明的你的帮助。

Input

输入一个正整数 $d(1 \le d \le 10^9)$，表示 $x - z$ 的值

Output

若存在方案，请输出任意一种。输出一行三个整数 x, y, z (以空格相隔)。

若无解，输出 “Impossible”(不包含引号)。

Examples

standard input	standard output
3	Impossible
4	6 4 2

Problem E. 采糖果

输入文件: standard input
输出文件: standard output
时间限制: 4 seconds
空间限制: 512 megabytes

小 S 和小 Y 正在参加 SYSU（Super Yummy -Suger Union-）的百年庆典。在庆典上，有这么一个活动：

给定一个 $n*m$ 的网格，每个格子里有一个糖果，糖果有一定的美味度（可以为负）。小 S 和小 Y 合作采糖果，小 S 从 $(1,1)$ 走向 (n,m)，每一步向下或向右走；小 Y 从 $(1,m)$ 走向 $(n,1)$，每一步向下或向左走。**每个格子里面的糖果只能被采一次**，而小 S 和小 Y 希望能获得最大的美味度总和。

然而小 S 和小 Y 只是两个喜欢吃糖果的女孩子，于是她们找到了博学、审问、慎思、明辨、笃行的你，你能想办法帮她们夺得头奖吗?

Input

第一行输入一个正整数 $t(1 \le t \le 10^3)$，表示数据组数。

对于每组数据，第一行输入两个正整数 $n,m(1 \le n,m \le 3 \times 10^3)$，表示网格的长宽。

接下来 n 行，每行 m 个整数，表示对应格子中糖果的美味度 $a_{i,j}(|a_{i,j}| \le 10^4)$。

保证对于每个测试点，$\sum n, \sum m$ 均不超过 3×10^3 。

Output

对于每组数据，输出一行一个整数表示答案。

Example

standard input	standard output
2 3 3 1 2 3 1 2 3 1 2 3 3 3 -1 -2 -3 -1 -2 -3 -1 -2 -3	18 -13

Problem F. 绝对霸主

输入文件: standard input
输出文件: standard output
时间限制: 3 seconds
空间限制: 4 megabytes

在 SYSU 举办百年校庆的喜庆氛围中，一场悄无声息的数字之争悄然展开！SYSU 的校友们化身为超级英雄，每个人都有一个数字代号。数字代号聚集在一起，组成了一支强大的数字联盟。但是，传闻中有一个数字代号出现的频率超过了其他所有代号的出现频率之和，成为了数字联盟中的绝对霸主！

作为 SYSU 聪明绝顶的数字学家，你的任务是揭示这个数字联盟的绝对霸主！

Input

输入第一行包含一个整数 n，表示超级英雄的数量。($n \leq 10^7$)

第二行包含 n 个整数，代表每位超级英雄的数字代号。

所有数字代号均在 1 到 10^9 之间。

Output

输出一个数字，表示绝对霸主的数字代号。

Examples

standard input	standard output
6 10 10 10 10 10 10	10
5 8 8 4 8 3	8

Note

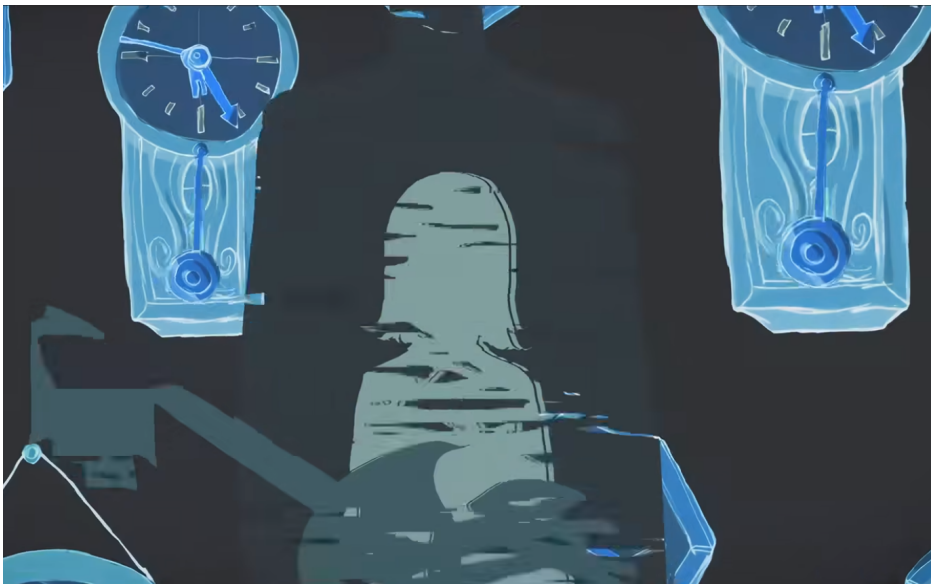
“int a[N]” 的空间占用为 $4N$ Byte $= \dfrac{4N}{2^{20}}$ MB

Problem G. 咬合秒针

输入文件:	standard input
输出文件:	standard output
时间限制:	3 seconds
空间限制:	256 megabytes

沉入灰烬　咬合秒针
淌于白日梦中
渐渐支离破碎
但是毁坏不了
也不允许我停住
只能不明「就里」地继续前进了
互相理解的轮回里即使一件都不存在
见了面「对不起。」这种的也不要丢还给我

— ずっと真夜中でいいのに。,《秒針を🄵む》



在时光之海中有一座拥有无限刻度的神秘钟盘，在其上有 n 根魔法时针。第 i 根时针会于 t_i 时刻出现，以刻度 x_i 为起点，循着特定的轨迹，直至抵达它的终点——刻度 y_i，当时针到达它的终点后便会消失。当 $y_i > x_i$ 时，时针将朝刻度增大的方向旋转；当 $x_i > y_i$ 时，时针将朝刻度减小的方向旋转。

所有时针的运动都是严格按照每个时刻移动一个刻度的速度旋转，而在这个钟盘上，当两根方向**相反**的时针在某一个时刻重合时，便会发生一次 **咬合秒针**。当然，时针在出现或者消失的时刻与方向相反的时针重合也是可以发生 **咬合秒针**的。

此刻，你被赋予了一项使命：请计算，这钟盘之上将会发生多少次 **咬合秒针**。

Input

第一行给出一个正整数 n $(1 \leq n \leq 5 \times 10^5)$。

接下来 n 行每行包括一个正整数 t_i, x_i, y_i $(1 \leq t_i, x_i, y_i \leq 10^9. \quad x_i \neq y_i)$。

Output

输出一行包括一个整数，表示将会发生多少次 **咬合秒针**。

Examples

standard input	standard output
5 1 1 5 2 9 1 3 1 10 2 6 2 2 1 4	4
3 1 1 7 4 8 9 5 9 5	2

Problem H. 罗德岛战役

输入文件: standard input
输出文件: standard output
时间限制: 1 second
空间限制: 256 megabytes

一大批整合运动的领袖带领整合运动进军罗德岛

而你，doctor，作为罗德岛的战术指挥官，率领着干员们与整合运动的领袖对抗。

擒贼先擒王，所以如何击溃整合运动的领袖们显得极为重要。

每一个整合运动的领袖，具有生命值、防御 d 、法抗 s 三个属性，要在有限的时间内击溃他，就要选择一些干员，使干员们在这段时间内造成的伤害 (实际造成的物理伤害 + 法术伤害 + 真实伤害) 大于等于敌方领袖的生命值。

于是，你每次都会从未被选中的干员中选择一名加入编队，或者从已选中的干员中将一名干员移除编队，并计算以目前编队内的干员，能否在有限时间内击杀这个敌方领袖。在将干员加入编队的时候，你会知道干员的代号以及这位干员在有限时间内能进行多少次攻击，每次攻击的物理伤害 a 、法术伤害 b 和真实伤害 c 分别是多少。计算敌人的防御 d 和法抗 s 后，实际造成的物理伤害为 $\max(a-d, \lfloor a*5\% \rfloor)$ ，法术伤害为 $\max(\lfloor b*\frac{100-s}{100} \rfloor, \lfloor b*5\% \rfloor)$ ，真实伤害为 c 。

对于不同的敌方领袖，你会选择不同的编队。因此在结束一个敌方领袖的编队选择后，你会先清空编队，再选择对抗下一个敌方领袖的编队。

$\lfloor x \rfloor$ 表示不大于 x 的最大整数，比如 $\lfloor 1.5 \rfloor = 1, \lfloor -1.5 \rfloor = -2$。

Input

第一行一个整数 T，表示整合运动的领袖个数 ($1 \le T \le 10^4$)

对于每一个敌方领袖，

第一行三个整数 h, d, s ($1 \le h \le 10^{14}$, $0 \le d \le 10^5$, $0 \le s \le 100$)，表示敌方领袖的生命值、防御和法抗。

第二行输入一个整数 m，表示添加或移除干员的操作次数。(保证 $1 \le \sum m \le 10^5$)

接下来 m 行，每行先第一个整数 opt。

若 $opt = 1$,表示将添加一名干员。随后输入一个仅由字母和数字组成的字符串 str_i 和 4 个整数 n_i, a_i, b_i, c_i ($n_i, a_i, b_i, c_i \in [0, 10^5]$)，$str_i$ 表示干员代号 ($|str_i| \le 50$)，n_i, a_i, b_i, c_i 分别表示在有限时间内能够进行的攻击次数，每次攻击的物理伤害、法术伤害和真实伤害。如果编队中已经有相应干员代号的干员，忽略这次添加干员的操作直接输出答案。

若 $opt = 2$ ，表示将移除一名干员，随后输入一个仅由字母和数字组成的字符串 str_i ，表示要移除的干员的代号。若编队中没有以这个字符串为代号的干员，忽略这次移除干员的操作直接输出答案。

Output

对于每一个领袖，共 m 行表示每次操作（可能被忽略）后现在的编队能否在有限的时间内击杀整合运动领袖。如果能，输出 *Yes*, 否则输出 *No*。(不区分大小写)

Example

standard input	standard output
1 50000 3000 80 12 1 Amiya 20 0 1000 2000 1 Amiya 20 0 1000 2000 1 Degenbrecher 10 5000 0 0 1 KirinRYato 16 1265 633 0 2 Texas 2 Amiya 1 Eyjafjalla 27 0 1875 0 1 Amiya 20 0 1000 2000 2 Degenbrecher 2 Degenbrecher 2 Eyjafjalla 1 Eyjafjalla 27 0 1875 0	No No Yes Yes Yes No No Yes Yes Yes No Yes

Problem I. 第三心脏

输入文件:	standard input
输出文件:	standard output
时间限制:	1 second
空间限制:	256 megabytes

每当一起回家时
分离以后脆弱的我冒出想法
没有和你道别是因为
不想你一个人被寂寞和黑暗吞噬
那颗冰冷的第三心脏
一直默默地在看着我们两人

—第三心脏

回家的路上有 n 个地点，由 m 条双向道路将其连接，第 i 条道路连接地点 x_i 与 y_i ，最开始每条道路都是黑暗的，你需要选择任意条 **不相交** (任意两条路径没有重复地点) 的 **路径**，将上面所有的道路点亮，让任意两个地点之间都 **存在**一条路径不超过 k 条 **未被点亮**的道路。

对于任意两个地点，保证可以通过双向道路连接。

路径：无重复节点的节点序列，相邻节点存在一条双向道路将它们相连。

Input

第一行输入三个正整数 n, m, k （$1 \le n \le 2000, 1 \le m \le 2500, 20 \le k \le n$ ）。

接下来 m 行，第 i 行输入两个正整数 x_i, y_i 表示有一条地点连接 x_i 与 y_i 的道路。

不存在任意两条道路连接的地点完全相同，不存在道路两端连接同一个节点，对于任意两个地点，保证可以通过双向道路连接。

Output

如果无解第一行输出一个 -1 。

否则第一行输出一个正整数 T 表示选择的路径条数。

接下来 T 行第一行一个正整数 w 表示该路径的点数，接下来 w 个正整数依次表示路径上的若干个点。

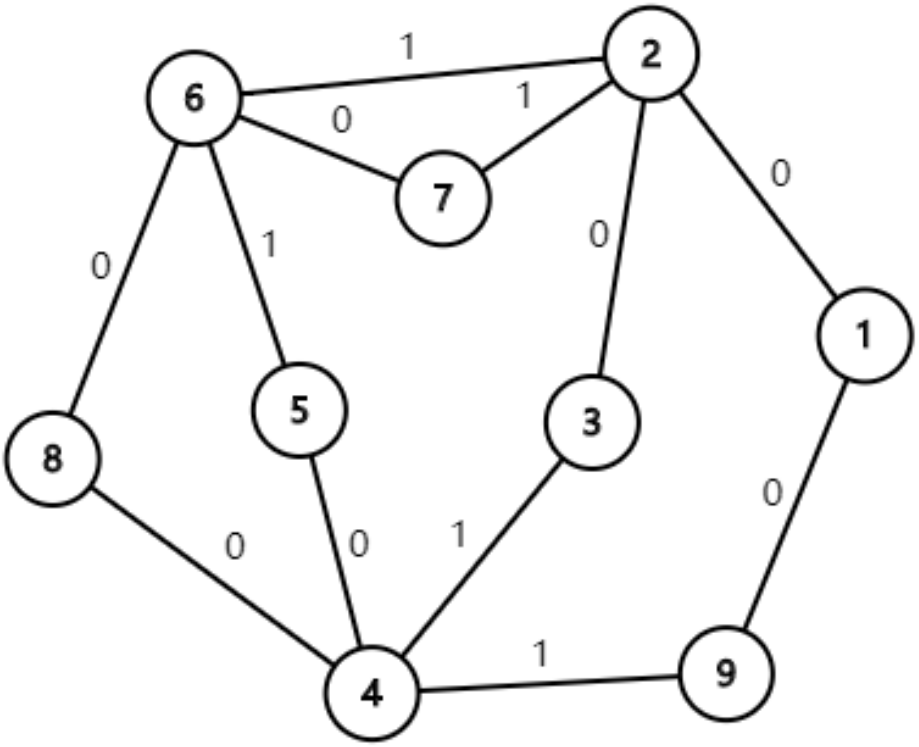
如题，你需要保证没有输出重复的点。

Example

standard input	standard output
9 12 20 1 2 1 9 2 3 2 6 2 7 3 4 4 5 4 8 4 9 5 6 6 7 6 8	2 5 5 4 8 6 7 4 3 2 1 9

Note

根据样例的输出，先绘制出连通图，先把所有道路的权值设为 1 ，然后把路径 $5 \to 4 \to 8 \to 6 \to 7$ 与路径 $3 \to 2 \to 1 \to 9$ 上的所有道路权值设为 0 得到下图。



不难发现图中所有点的最短距离都不超过 1 ，满足 $\leq k$ 的条件, 且选出的路径没有重复的部分。

Problem J. 回忆之树

输入文件: standard input
输出文件: standard output
时间限制: 1 second
空间限制: 256 megabytes

你知道吗？此段记忆，此篇关于光明与纷争的故事……

以一粒装满了感情的种子被埋进土中为始。

然后，在被珍视和被唾弃的回忆的共同滋养下，开始生根发芽。

再然后，它便茁壮成长，执拗地想去到更高的地方，一如这执拗的时间。

再然后呢，它就长成了一棵……BFS 树???

Arcaea 的世界是一张无向连通图。风云变幻，世界的法则不断更迭。回忆之树仍然挺立，但回忆之源却已然模糊……

现在给出 Arcaea 世界的构造（即一张无向连通图 G），再给出回忆之树的构造（即 G 的一棵生成树 T），有多少个节点可能是回忆之源呢？（即满足 T 是以该节点为根的一棵 BFS 树）

【名词解释】

生成树：一个连通无向图的生成子图，同时要求是树。也即在图的边集中选择 $n-1$ 条，将所有顶点连通。

BFS：BFS 全称是 Breadth First Search，中文名是宽度优先搜索，也叫广度优先搜索。所谓宽度优先，就是每次都尝试访问同一层的节点。如果同一层都访问完了，再访问下一层。

BFS 树：在 BFS 过程中，通过记录每个节点从哪个点访问而来，可以建立一个树结构，即为 BFS 树。BFS 的起点即为树根。

Input

第一行一个正整数 $t(1 \le t \le 10^4)$，表示数据组数。

对于每组数据，第一行输入两个正整数 $n, m(1 \le n \le 2 \times 10^5, n-1 \le m \le \min(\frac{n(n-1)}{2}, 3 \times 10^5))$，表示无向连通图 G 的点数和边数。

接下来 m 行，每行两个正整数 $i, j(1 \le i, j \le n, i \ne j)$，表示给定的无向连通图 G 中，i, j 间存在一条无向边。

接下来 $n-1$ 行，每行两个正整数 $i, j(1 \le i, j \le n, i \ne j)$，表示给定的生成树 T 中，i, j 间存在一条无向边。

保证不存在重边，所有数据组中 n 的总和不超过 2×10^5，m 的总和不超过 4×10^5。

Output

对于每组数据，首先输出一行一个整数 c，表示合法的节点数目。

接下来一行**从小到大输出 c 个正整数，表示合法节点的编号。注意，若 $c = 0$，也需要输出一个空行。**

Example

standard input	standard output
2 7 7 1 2 1 3 2 4 2 5 3 6 3 7 2 3 1 2 1 3 2 4 2 5 3 6 3 7 5 5 1 2 2 3 4 3 4 5 5 2 1 2 2 3 3 4 4 5	1 1 2 3 4

Problem K. 骤雨

输入文件: standard input
输出文件: standard output
时间限制: 1 second
空间限制: 256 megabytes

让这崩落的世界得到修复，完好如初。

让雨点和雪花在这片它们从未光顾过的荒芜土地上降下。

让毫无生气的死海再次满盈，让沉寂许久的波涛再度翻涌。

让新的法则一锤定音，就此挥之不去。

2024 年 11 月 12 日，漩涡滚滚，巨浪滔天。本只是来参加百年庆典的你，却意外卷入这场风暴之中...

所谓的法则呢，就是这么个东西：

称一个长度为 m 的整数序列 c 是一个 V-diagram，当且仅当 $m \geq 3$，且存在一个下标 $1 < i < m$ 满足：

- 对所有 $1 \leq j < i$，有 $c_j > c_{j+1}$。
- 对所有 $i < j \leq m$，有 $c_j > c_{j-1}$。

给定每个位置的取值范围，你能否造出一个长度为 n 的 V-diagram，平息这一切呢?

Input

第一行一个正整数 t（$1 \leq t \leq 5 \times 10^4$），表示数据组数。

对于每组数据，第一行输入一个正整数 n（$3 \leq n \leq 2 \times 10^5$），表示序列长度。

接下来 n 行，每行两个正整数 l, r（$1 \leq l \leq r \leq 10^9$），表示每个数的取值范围。

保证所有数据组中 n 的总和不超过 2×10^5。

Output

若存在合法方案，则输出一行 `YES`，再输出一行 n 个数，表示构造的一组方案；

若不存在合法方案，只需输出一行 `NO`。

`YES` 与 `NO` 不区分大小写。

Example

standard input	standard output
2 5 8 10 7 10 6 10 5 10 4 10 4 4 5 4 5 4 5 4 5	YES 10 7 8 9 10 NO

Problem L. 交通建造

输入文件: standard input
输出文件: standard output
时间限制: 2 seconds
空间限制: 256 megabytes

Jack 是掌管 Minecraft 的神，他在 Minecraft 中创建了一个国家。这个国家有 m 个省，其中第 i 个省有 n_i 个市。Jack 已经在每个省建造了 $n_i - 1$ 条铁路来使各市联通。

建造完省内交通后，Jack 决定开始建造各省之间的交通。为了节约成本，Jack 将只建造 $m - 1$ 条铁路，每条铁路连接两个来自不同省的市，并最终将各省各市通过铁路联通。

但是，这样建立交通也存在一个问题：从一个市到达另一个市，可能需要经过很多段不同的铁路，也就是说要转很多次车。Jack 想知道，如果任意建造这些铁路，在建成各省之间的交通后，最远的两个市之间的铁路个数的最大值和最小值。由于 Jack 在 Minecraft 中的计算机还在建造中，所以他把这个问题交给了你。

Input

第一行包含一个整数 $T(1 \le T \le 10^4)$ ，表示数据组数。

每组数据的第一行包含一个整数 $m(1 \le m \le 10^6)$，表示有 m 个省。

对于每个省，第一行包含一个整数 $n(n \ge 1)$ ，表示这个省有 n 个市。

接下来 $n - 1$ 行，每行包含两个整数 u, v ，表示这个省的 u 市和 v 市之间有一条铁路连接。

对于所有数据，保证 $\sum n \le 10^6$ 。

Output

对于每组数据，

第一行一个整数，表示在建成各省之间的交通后，最远的两个市之间的铁路个数的最大值。

第二行一个整数，表示在建成各省之间的交通后，最远的两个市之间的铁路个数的最小值。

Example

standard input	standard output
2 2 10 1 4 7 8 10 6 2 6 6 5 3 6 6 8 9 6 4 6 6 1 3 5 6 1 2 5 4 3 4 3 8 5 3 6 8 4 6 1 6 4 2 7 1 3 6 3 1 3 3 2 4 1 3 4 2 2 1	10 6 11 5

Note

对于样例的第一组数据，在 1 省的 7 市和 2 省的 2 市之间建造铁路能使最远的两个市之间的铁路个数取得最大值，这样 1 省的 1 市和 2 省的 6 市就是距离最远的两个市，他们之间的铁路个数为 10。

在 1 省的 6 市和 2 省的 4 市之间建造铁路能使最远的两个市之间的铁路个数取得最小值，这样 1 省的 7 市和 2 省的 2 市就是距离最远的两个市，他们之间的铁路个数为 6。

注意：本题输入量较大，请注意使用效率较高的读入方式。

Problem M. 计算

输入文件: standard input
输出文件: standard output
时间限制: 1 second
空间限制: 256 megabytes

已经没有什么好怕的了。

— 《魔法少女小圆》

这是一个存在魔法的世界。世界上存在极少数的少女会与 QB 签订魔法契约，成为“魔法少女”以对抗邪恶的魔女保护世界。

逸仙书院校庆即将到来，小 C 想和同伴一起庆祝这个日子，正当他们在准备的时候，未知的敌人突然出现在了这里，企图制造混乱。这时 QB 出现在了他的面前，她希望小 C 和自己签订契约，成为“魔法少女”来保护她的朋友和这里的一切。作为交换，小 C 可以实现一个愿望。

正巧小 C 正烦恼于某门科目的结课作业，作业只有一道问题。问题是这样的：小 C 每次会收到一串信息, 信息由数字和 '|'、'&'、' ^ ' 三种运算符构成，数字连续长度不超过 5 ，特别的，'^' 后面的指数最多只有一位，且数值 ≤ 3 ，保证不会出现运算符相邻情况，并保证 '^' 运算符不会连续出现，即 `3^3^3` 是错误的。

小 C 需要 QB 帮他得到这一串信息运算后的结果，由于 QB 数学不好，于是她找到了人类世界中为数不多的朋友——你，并向你求助，希望你能够帮帮她。

Input

仅包含一行一个字符串 $S(|S| \leq 2 \times 10^6)$，表示 QB 收到的一串信息。(保证 S 中的字符由数字,'|','&','^' 构成。)

Output

输出仅一行, 包含一个数字, 表示你的答案。

Examples

standard input	standard output
1&1^2|1^1	1
11451&1919|810	1851

Note

注意, 在本题中，'|','&' 是位运算符，但 '^' 是乘方运算符，具有右结合性。

优先级: 乘方 (^)> 按位与 (&)> 按位或 (|)。

Problem N. 不为人知的位运算童谣

输入文件: standard input
输出文件: standard output
时间限制: 1 second
空间限制: 256 megabytes

言语都是过剩多余
这梦境也将持续下去
若要由我来谈论爱的话
那一切都已在这首歌中
谁也不知道的这个故事
似乎又不小心哼唱出来了

鹅妈妈有一段童谣，它是一个长度为 n 的严格单调递增非负整数序列 $\{a_n\}$ (即 $\forall i \in [1, n-1], a_i < a_{i+1}$)。

鹅妈妈打算用一个非负整数 x 去保存这段童谣，序列会保留为 $\{a_n \text{ and } x\}$ 。

因为 x 越大越容易被人忘记，所以鹅妈妈希望找到最小的一个**非负**整数 x ，使得 $\{a_n \text{ and } x\}$ 依然是一个**严格单调递增**的序列 (即 $\forall i \in [1, n-1], (a_i \text{ and } x) < (a_{i+1} \text{ and } x)$) 。

其中 and 操作表示 **按位与**，如果你不了解定义可以阅读下列定义：

将两个整数作为二进制数，对二进制表示中的每一位逐一运算，只有对应的两个二进位都为 1 时，结果位才为 1 。在 C++ 中, 可以使用 “&” 运算符进行按位与。

Input

第一行给出一个正整数 n ($2 \le n \le 10^5$) 。

接下来一行给出 n 个非负整数表示 a_i ($0 \le a_i < 2^{31}$) 。

Output

输出一行一个非负整数表示满足条件最小的 x 。

Example

standard input	standard output
5 0 2 5 114 514	534

Note

对于样例中，所有的 a_i and x 分别是：$0, 2, 4, 18, 514$ 。可以证明，不存在更小的 x 满足约束条件。