Problem A. We are watching you!

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

（题目背景纯属虚构，如有雷同，~~不胜荣幸~~ We are watching you!）

请注意本题的空间限制。

作弊哥是一个资深代打高手，他经常在 OCASU、oaiqnal、CPCX 等各种编程比赛中替别人代写代码，赚取了丰厚的收入。在比赛过程中，作弊哥会将题目的正确代码 s_1 发送给作弊的参赛选手；这个参赛选手会在代码的 **开头** 随便加点没用的东西，构造出完整代码 s，就能提交。

作弊哥靠着这招屡试不爽。然而，这一次他不幸被官方盯上了。

作为官方的技术审查专家，小 A 需要对于选手提交的代码 s，分析 s 的任意一个后缀的代码风格有多大概率出自于作弊哥。为了方便检验代码的同时，尽量不影响评测机速度，小 A 构造了一个能接受所有 s 后缀的最小化确定型有限状态自动机（Deterministic Finite Automaton，DFA）。

接下来，小 A 按深度优先搜索的方法，遍历这个 DFA，并分析出状态 i 和作弊哥的代码风格有 c_i 点相似度。小 A 认为子串 s' 的相似度为 DFA 在输出 s' 的过程中，经过状态的最大相似度；而完整代码的相似度为其所有非空子串相似度的平均值。

现在，小 A 拿到了一些选手的完整代码 s，依此构建了最小化 DFA，并按深度优先搜索的遍历方法给出了 DFA 各状态的相似度。请你帮忙评估这些代码和作弊哥代码的相似度。

如果你不了解 DFA 以及 DFA 的深度优先遍历，请阅读补充提示。

形式化的，以下代码描述了上述过程以及需求，但由于过大的时间复杂度，需要你优化并使其可以通过本题。

```
#include <bits/stdc++.h>
using i64 = long long;
struct SAM {
    struct Node {
        int fa, len;
        std::array<int, 26> trans;
        Node() : fa{}, len{}, trans{} {}
    };
    std::vector<Node> t;
    SAM() : t(2) {}
    int New() {
        t.push_back(Node());
```

```
        return t.size() - 1;
    }
    int extend(int lst, int c) {
        int u = lst, v;
        if (trans(u, c)) {
            if (len(u) + 1 == len(v = trans(u, c))) {
                return v;
            }
            int x = New();
            len(x) = len(u) + 1, fa(x) = fa(v);
            t[x].trans = t[v].trans;
            for (fa(v) = x; u && trans(u, c) == v; trans(u, c) = x, u = fa(u));
            return x;
        }
        int x = New();
        len(x) = len(u) + 1;
        for(; u && !trans(u, c); trans(u, c) = x, u = fa(u));
        if (!u) {
            fa(x) = 1;
        } else if (len(u) + 1 == len(v = trans(u, c))) {
            fa(x) = v;
        } else {
            int w = New();
            len(w) = len(u) + 1, fa(w) = fa(v);
            t[w].trans = t[v].trans;
            for (fa(v) = fa(x) = w; u && trans(u, c) == v; trans(u, c) = w, u = fa(u));
        }
        return x;
    }
    int& fa(int x) { return t[x].fa; }
    int& len(int x) { return t[x].len; }
    int& trans(int x, int c) { return t[x].trans[c]; }
};

void solve() {
    std::string s;
    std::cin >> s;
    SAM sam;
    int lst = 1;
    for (auto c : s) {  // 请注意这里是建立后缀自动机
        lst = sam.extend(lst, c - 'a');
    }
    std::vector<int> c(sam.t.size());
    for (int i = 1; i < c.size(); i++) {
        std::cin >> c[i];
    }
    i64 ans = 0;
    for (int i = 0; i < s.size(); i++) {
        int now = c[1], x = 1;
        for (int j = i; j >= 0; j--) {
            x = sam.trans(x, s[j] - 'a');
            now = std::max(now, c[x]);
            ans += now;
        }
    }
    std::cout << ans << std::endl;
}
int main() {
    int T;
    std::cin >> T;
```

```
    while (T--) {
        solve();
    }
    return 0;
}
```

Input

本题包含多组测试数据。

第一行是一个正整数 T（$1 \le T \le 2 \times 10^5$），表示有 T 份完整代码。

接下来 T 组数据。第一行是一个由小写字母组成的字符串 S，表示选手提交的代码。保证字符串长度 $|S|$ 满足（$1 \le |S| \le 2 \times 10^5$）。

第二行由 m 个整数 $c_1, c_2, \cdots, c_m$（$1 \le c_i \le 2 \times 10^5$）构成。其中 m 表示小 A 构造的最小化 DFA 的状态数，c_i 表示这个 DFA 按深度优先搜索，访问到的第 i 个状态的相似度。

数据保证 $\sum |S| \le 2 \times 10^5$；数据保证对于每组的字符串 S，其所有后缀构成的最小化 DFA 状态数恰好为 m。

Output

输出包含 T 行，其中第 i 行表示第 i 份代码的相似度。

为了避免除法运算，对于每个提问 S，你只需要输出答案乘上 $\frac{|S| \cdot (|S|+1)}{2}$，答案可以保证这是个整数。

Example

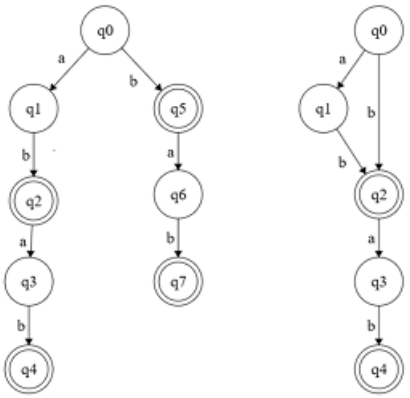
standard input	standard output
5 abb 1 1 3 1 2 oixcpc 1 1 4 5 1 4 1 9 tarjen 1 1 1 1 1 1 1 nanani 1 1 1 1 1 1 1 wildfire 1 1 1 1 1 1 1 1 1 1	13 109 21 21 36

Note

1. 确定型有限状态自动机（DFA）

确定型有限状态自动机（DFA）是一个状态机。它从固定的起始状态 q_0 出发，不断读入字符 c，并不断依此跳到后续状态；读入不同的字符将会跳转到不同的后续状态。如果读入完整个字符串后停在"接受状态"，我们认为 DFA 接受这个字符串；否则，认为 DFA 不接受这个字符串。特别的，如果 DFA 在某个状态 q 时读入字符 c 无后续状态，我们同样认为 DFA 不接受这个字符串。

如下图所示，带两个圆的状态为接受状态。左侧的 DFA 从初始状态 q_0 开始，读入 abab、bab、ab、b 后会分别停止于 q_4, q_7, q_2, q_5，均是接受状态；且读入其他字符串均不能进入接受状态。因此左侧的 DFA 可以接受 abab 的所有后缀；右侧的同理。



对于能识别相同字符串的所有 DFA，我们认为状态最少的 DFA 就是最小化 DFA。可以证明，不同的最小化 DFA 可以通过给状态重新编号，而变成相同的 DFA。

如该图所示，左右两侧的 DFA 均只能接受 abab、bab、ab、b 这四个字符串，因此两者等价。此外，可以证明，右侧的 DFA 是能识别这类字符串的最小化 DFA。

2. DFA 的深度优先遍历

DFA 的深度优先遍历将从起始状态 q_0 开始，每次选择当前状态未访问的、按字母表从小到大的下一个字符对应的边，直到遍历完所有状态。各状态的编号即为其被访问到的顺序。

如左侧的 DFA 的深度优先遍历顺序为 $q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7$；

右侧 DFA 的深度优先遍历顺序为 q_0, q_1, q_2, q_3, q_4。

Problem B. XCPC

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

在《XCPC》赛事系列中，有金、银、铜、铁四种奖牌，其价值分别为 4，3，2，1。在本题中，2 个铁牌可以合成 1 个铜牌，2 个铜牌可以合成 1 个银牌，2 个银牌可以合成 1 个金牌。

现在你有 n 个铁牌，你可以通过一番合成，将你拥有的奖牌数量改变，用四元组 (a_1, a_2, a_3, a_4) 分别表示金银铜铁奖牌的数量。

请回答以下 n 个问题，第 $i\,(1 \le i \le n)$ 个问题是：

* 初始时恰好有 n 个铁牌的情况下，最终有多少种不同的四元组 (a_1, a_2, a_3, a_4)，同时满足：（1）一共有 i 个牌子，即 $a_1 + a_2 + a_3 + a_4 = i$；（2）奖牌的价值之和大于等于 p，即 $4a_1 + 3a_2 + 2a_3 + a_4 \ge p$。

其中 p 通过输入给定。

两个四元组不同当且仅当它们存在某一位对应的数字不同。

Input

第一行输入两个整数 $n, p\,(1 \le p \le n \le 10^6)$，用空格相隔，分别表示初始有 n 个奖牌，以及问题要求的奖牌价值之和大于等于 p。

Output

输出一行 n 个整数，用空格相隔，第 $i\,(1 \le i \le n)$ 个数字表示第 i 个问题的答案。

Examples

standard input	standard output
8 7	0 0 1 2 2 1 1 1
10 8	0 0 1 2 2 2 2 1 1 1
12 1	0 1 2 2 3 3 2 2 2 1 1 1

Problem C. 中位数

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

小 A 拥有一个长度为 n 的数组 a，且 n 必定为奇数。

由于小 A 十分喜爱中位数，他将进行以下操作：每次选择数组中连续的三个数字，并将它们合并为其中位数，替换这三个数字。具体而言，每次选择任意一个位置 i（满足 $1 < i < n$），删除 a_{i-1}、a_i 和 a_{i+1}，并在该位置插入这三个数字的中位数。

小 A 将持续进行上述操作，直到数组中仅剩一个数字为止。整个过程共需进行 $\frac{n-1}{2}$ 次合并。他期望这个最终剩余的数字尽可能大。你的任务是帮助小 A 确定这个数字的最大可能值。

中位数的定义为：将一组长度为n的数组从小到大排序后，排名第$\lfloor \frac{n+1}{2} \rfloor$小的数字。

Input

本题包含多组测试数据。

第一行，包含一个整数 T（$1 \le T \le 10^6$），表示测试数据的组数。

对于每组数据:

第一行，包含一个整数 n（$1 \le n < 10^5$），且 n 必定为奇数。

第二行，包含 n 个整数 $a_1, a_2, \cdots, a_n$（$1 \le a_i \le 10^9$）。

数据保证 $\sum n \le 10^6$。

Output

对于每个测试用例，输出 $\frac{n-1}{2}$ 次合并以后剩下数字的最大值。

Example

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

Note

样例解释:

对于第四个样例而言，数组 $A = [\ 1\ 2\ 3\ 5\ 6\ 7\ 4\ ]$ 一种可行的方案是: $[\ \underline{1\ 2\ 3}\ 5\ 6\ 7\ 4\ ] \to [\ \underline{2\ 5\ 6}\ 7\ 4\ ] \to [\ \underline{5\ 7\ 4}\ ] \to [\ 5\ ]$。

其中 $\underline{a_{i-1}\ a_i\ a_{i+1}}$ 下划线选择的连续三个数字表示每次操作合并的对象。

Problem D. 二叉树

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 megabytes

墨菲特非常喜欢完美的树。在某一天，他决定用茂凯给的神秘种子自己种一棵树。

这颗种子拥有非常旺盛的生命力。将这颗种子种下后，如果我们把第一天种下的种子记为初始的叶子节点，那么从第二天开始，每个前一天长出来的叶子节点，都会长出两个新的叶子节点。换句话说，如果这颗树生长了 k 天，那么它就是一棵拥有 $2^k - 1$ 个节点的满二叉树。

由于其迅速的生长速度，墨菲特很快就能得到一棵拥有很多节点的满二叉树，他认为满二叉树是十分完美的。

茂凯也知道墨菲特非常喜欢树，因此他带来了一份礼物：由同样的神秘种子种出来的一棵树（大小不一定相同）。然而墨菲特已经有一颗这样的树了，两棵树的存放不太方便，于是他在两棵树之间连接了一条新的边，这样就是一棵树了。

显然，这样的树已经不是完美的二叉树了，因此墨菲特很快就把他丢在了角落。

过了很久很久之后，墨菲特又想起来了这棵由两棵满二叉树连接形成的新树，他想要重新把这棵树拆成两棵满二叉树，但是他已经忘记哪条边是他额外添加上去的了。

希望聪明的你可以帮助墨菲特解决这个问题。

形式化地，给定一棵树，你需要删除其中的一条边，使得分成的两棵树都是满二叉树，保证至少存在一个解。

*所有叶结点的深度均相同，且所有非叶节点的子节点数量均为 2 的二叉树称为满二叉树。

Input

输入包含多组测试数据。第一行包含一个正整数 T ($1 \le T \le 10^5$)，表示测试数据的数量。接下来是测试数据的描述。

每组测试数据的第一行包含一个正整数 n $(2 \le n \le 2^{17} - 2)$，表示墨菲特手上的树的节点个数。

接下来 $n - 1$ 行，每行两个由空格分开的正整数 u_i, v_i $(1 \le u_i, v_i \le n)$，表示树上的一条边 (u_i, v_i)。输入保证给定的边集构成一棵树。

另外，保证 $\sum n \le 10^6$ 。

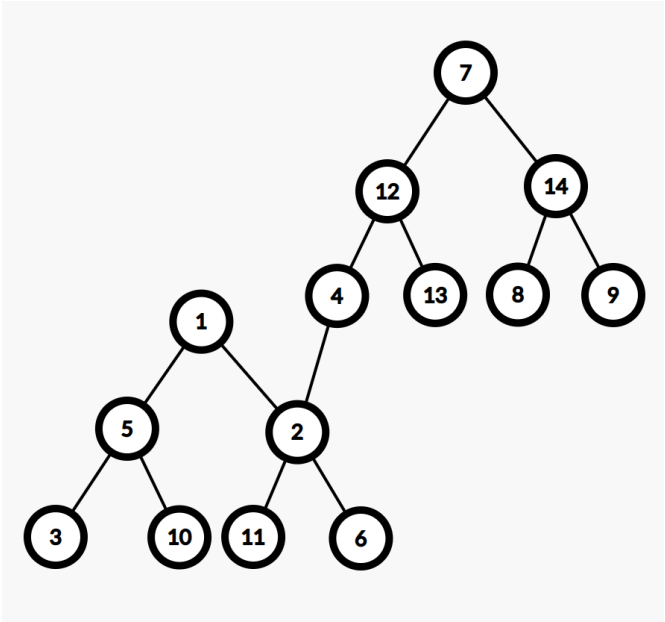
Output

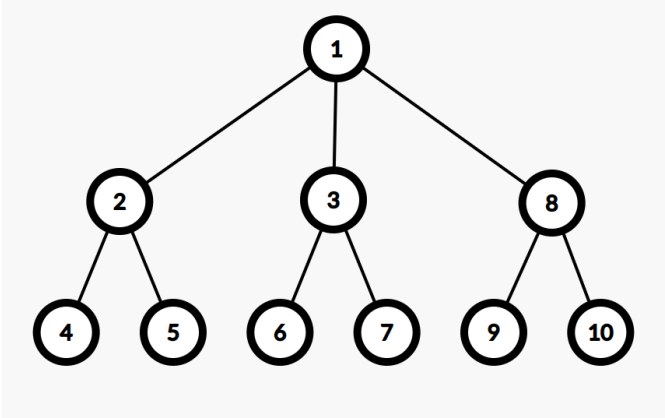
对于每一组测试数据，输出一行一个正整数 i，表示如果删除了给定边集中的第 i 条边 (u_i, v_i)，分成的两棵树都是满二叉树。保证这样的解一定存在。如果有多个可行的解，输出任意一个即可。

Example

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

Note





Images side by side example.

第二组样例的树如上图所示，可以发现，只有删除边 $(2,4)$ 是合法的。

第三组样例的树如下图所示，可以发现，删除边 $(1,2),(1,3),(1,8)$ 都是合法的，你可以输出任意一种。

Problem E. 卡牌游戏

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

小 A 和小 B 正在玩卡牌游戏。

有 $2n$ 张卡牌垒成一摞牌堆，自上而下的第 i（$1 \le i \le 2n$）张牌上标注了数字 a_i。发牌时，牌堆中的卡牌自上而下以 $1, 2, \ldots, 2n$ 编号，编号为奇数的卡牌将分给一位玩家，编号为偶数的卡牌将分给另一位玩家。这意味着，小 A 将会获得编号同为奇数或同为偶数的 n 张卡牌。

对于玩家而言，所得到的卡牌上的数字之和越大，游戏局面对他越有利。因此小 A 想最大化最坏情况下他所得到的卡牌数字之和。为了达到这个目的，小 A 可以对当前牌堆执行恰好一次以下操作：

- 从当前牌堆中抽出一张卡牌，并插回牌堆中的任意位置。注意发牌时的编号可能会发生变化。

例如，初始时牌堆中卡牌标注的数字自上而下依次是 $1, 2, 3, 4$，发牌时一位玩家将得到 $1, 3$，另一位玩家将得到 $2, 4$。小 A 可以选择抽出第二张卡牌，并将其插回最后一张卡牌后面，此时牌堆为 $1, 3, 4, 2$，发牌时一位玩家将得到 $1, 4$，另一位玩家将得到 $3, 2$。

你需要求出小 A 在执行恰好一次操作后，最坏情况下所得到的卡牌数字之和的最大值。

Input

本题包含多组测试数据。

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

对于每组数据：

第一行，一个正整数 n（$1 \le n \le 10^5$），表示每位玩家将得到的卡牌数量。

第二行，$2n$ 个正整数 $a_1, a_2, \ldots, a_{2n}$（$1 \le a_i \le 10^9$），表示当前牌堆中卡牌标注的数字。

对于每个测试点，保证 $\sum n$ 不超过 10^5。

Output

对于每组数据，输出一行，一个整数，表示在执行恰好一次操作后，最坏情况下小 A 所得到的卡牌数字之和的最大值。

Example

standard input	standard output
4 2 1 3 2 4 1 1000000000 1 3 1 1 2 3 5 8 4 1 2 4 8 16 32 64 128	5 1 9 106

Note

第一组样例即是题目描述中所给出的例子。发牌时，一位玩家的卡牌数字之和为 $1 + 4 = 5$，另一位玩家的卡牌数字之和为 $3 + 2 = 5$，可以证明这即是所能取得的最大值。

第二组样例中，无论如何操作，发牌情况都将是一位玩家得到 1，而另一位玩家得到 10^9。因此最坏情况下小 A 只能得到 1，即为答案。

Problem F. 帕累托前沿

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

给出 n 个二元组 (x_i, y_i)，你需要回答 q 个询问，每个询问给出闭区间 $[l, r]$，请回答满足以下条件的整数 j 数量:

- $l \le j \le r$
- 不存在 $l \le k \le r, k \ne j$，使得 $x_k \ge x_j$ 且 $y_k \ge y_j$

Input

第一行两个正整数 n, q（$1 \le n, q \le 10^6$），分别表示二元组数量和询问数量。

第二行 n 个非负整数 $x_1, x_2, \ldots, x_n$（$0 \le x_i \le 10^6$）。

第三行 n 个非负整数 $y_1, y_2, \ldots, y_n$（$0 \le y_i \le 10^6$）。

接下来 q 行，每行两个正整数 l, r（$1 \le l \le r \le n$），表示询问的区间。

Output

对于每个询问，输出一行一个整数表示答案。

Example

standard input	standard output
8 7 1 9 7 8 0 7 2 3 19 20 5 6 1 14 9 5 1 8 3 7 2 6 4 4 5 7 3 8 6 7	1 2 1 1 1 2 1

Note

对于询问 1，满足条件的整数为 2。

对于询问 2，满足条件的整数为 4、6。

对于询问 3，满足条件的整数为 2。

对于询问 4，满足条件的整数为 4。

对于询问 5，满足条件的整数为 6。

对于询问 6，满足条件的整数为 4、6。

对于询问 7，满足条件的整数为 6。

Problem G. 炒股高手

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

小 A 立志成为一名炒股高手。经过长时间的学习，他自信已掌握了丰富的理论知识，并精心挑选出了一支股票，决定在接下来的 n 个交易日内进行实战操作。

股票之神向小 A 投来了注视，他帮你预知了接下来 n 个交易日中这只股票的股价。由于股票之神擅长数学，所有价格均以自然对数的形式给出。也就是说，在第 i 天（$1 \le i \le n$），会得到一个正整数 a_i，表示当天该股票的价格为 e^{a_i} 元。需要注意的是，在本题中，小 A 可以购买非整数份的股票。

由于手头资金有限，小 A 决定公开向他人借款，称为"鸡债"。为了方便管理和收益核算，他规定每位出借人均提供固定的 e^k 元借款。

在这 n 天内，共有 m 位出借人愿意向小 A 提供鸡债。

对于第 i 位出借人，他会在第 s_i 天开盘前向小 A 提供 e^k 元，在第 t_i 天收盘后要求结算收益。在这段时间内，小 A 可自由使用这笔资金进行任意次数的买入与卖出操作。

你的任务是：对于每一笔鸡债，计算小 A 在最优操作下所能获得的最终总资产（即本金加收益）的自然对数值 k，并输出该整数。 换句话说，若小 A 最终通过操作将手中 e^k 元变为 e^w 元，则你应输出该整数 w。

Input

第一行包含两个正整数 n（$1 \le n \le 10^5$） 和 m（$1 \le m \le 10^5$），分别表示交易日的数量以及鸡债的数量。

第二行包含 n 个正整数 $a_1, a_2, \ldots, a_n$（$1 \le a_i \le 10^4$），其中第 i 天的股票价格为e^{a_i}。

第三行包含一个正整数 k（$1 \le k \le 10^9$），表示每份鸡债提供的借款金额为e^k。

接下来 m 行，每行包含两个整数 s_i 和 t_i（$1 \le s_i \le t_i \le n$），表示第 i 份鸡债的借款开始日和收益结算日。

Output

输出共 m 行，每行一个正整数 w_i，表示第 i 份鸡债的本金加收益总额为 e^{w_i} 元。

Example

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

Note

在样例中，一共有 6 个交易日，2 份鸡债。

每天的股价为 $e^3, e^2, e^4, e^5, e^3, e^6$ 。

鸡债的本金均为 e^2 。

第 1 份鸡债：第 2 天借出，在第 4 天归还。

第 2 份鸡债：第 3 天借出，在第 6 天归还。

对于第 1 份鸡债：三天的股价为 $[e^2, e^4, e^5]$ ，最优方案为第 2 天买入，第 4 天卖出。最终资产为 $e^2 \div e^2 \times e^5 = e^5$ 。

输出答案为 5 。

对于第 2 份鸡债：四天的股价为 $[e^4, e^5, e^3, e^6]$ ，最优方案为第 3 天买入，第 4 天卖出，第 5 天再买入，第 6 天再卖出。最终资产为 $e^2 \div e^4 \times e^5 \div e^3 \times e^6 = e^6$ 。

输出答案为 6 。

Problem H. 难以控制的滑板火箭

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

在一个 $n \times m$ 的 `01` 网格中，其中第 i 行第 j 列第元素为 $a_{i,j}$，若 $a_{i,j} = 1$ 则表示这个位置为空地，反之若 $a_{i,j} = 0$ 则表示这个位置上有障碍物。

现在小猫从 $(1, 1)$ 出发，想要去 (n, m)。

若小猫当前在 (x, y) 则**一次移动**后可以到 $(x-1, y)$、$(x+1, y)$、$(x, y-1)$、$(x, y+1)$、$(x-1, y-1)$、$(x+1, y-1)$、$(x$ 的位置上，注意不能移动到地图外，也不能走到障碍物上。即任意时候 $1 \le x \le n, 1 \le y \le m, a_{x,y} = 1$。

因为小猫使用了难以控制的滑板火箭，每一分钟都会移动 $[l, r]$ 次。

现在需要你求出小猫最少需要几分钟才能成功抵达终点（**必须要某一分钟的移动全部结束后小猫的位置在** (n, m) **才算成功抵达**），如果无论经过多久都不能成功抵达请输出 `-1`。

Input

第一行一个整数 t（$1 \le t \le 1000$），表示测试数据组数。

接下来对于每一组测试数据，第一行两个整数 n, m（$2 \le n, m \le 1000$），表示 `01` 网格的大小。

接下来一行包含两个整数 l, r（$1 \le l \le r \le 10^9$），表示在一分钟内移动次数的限制范围。

接下来 n 行，每行 m 个字符，表示网格的元素 $a_{i,j}$，字符仅会出现 `0` 或 `1`，且 $a_{1,1}$ 与 $a_{n,m}$ 一定为 1。

保证所有测试数据的 $n \times m$ 的和不超过 10^6。

Output

对于每一组测试数据输出一行，如果小猫能在有限时间内抵达 (n, m)，那么输出最少需要的分钟数，否则输出 `-1`。

Example

standard input	standard output
3 5 5 2 3 10000 01000 00110 11001 11111 7 8 3 3 10101000 01010100 10000100 01000010 00100100 00011010 00000001 7 8 4 4 10101000 01010100 10000100 01000010 00100100 00011010 00000001	2 3 3

Note

对于第一组样例：

在第一分钟 $(1,1) \to (2,2) \to (3,3) \to (3,4)$；

在第二分钟 $(3,4) \to (4,5) \to (5,5)$。

Problem I. 割点

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

给定一个正整数 n 和一个长度为 $n-2$ 的 01 序列 $a_2, a_3, \ldots, a_{n-1}$，要求你构造一个 n 个点的**无向简单连通图** G，使得：

- 点 1 是割点，点 n 不是割点。
- 对于每个 $1 < i < n$：
  若 $a_i = 1$，则点 i 在图 G 中是割点；
  若 $a_i = 0$，则点 i 在图 G 中不是割点。
- 图 G 中各顶点的度数满足：$\deg_1 \geq \deg_2 \geq \cdots \geq \deg_n$。

如果存在多种可行的图，输出任意一种；如果不存在满足条件的图，则输出 -1。

简单图的定义为：无重边（即任意一对点之间至多只有一条边）且无自环（即不存在一条边两端点相同）的图。

割点的定义为：删掉该点以及它连的边后，使得图连通块个数增加的点。

Input

本题包含多组测试数据。

第一行一个正整数 T（$1 \leq T \leq 500$），表示测试数据的组数。

对于每组数据：

第一行一个正整数 n（$4 \leq n \leq 1000$），表示图中点的个数。

接下来一个长度为 $n-2$ 的 01 串，表示序列 $a_2, a_3, \ldots, a_{n-1}$。

保证所有数据的 $\sum n \leq 2000$。

Output

对于每组数据：

若无解，输出 -1；

若有解，第一行输出 m （$0 < m \leq \frac{n(n-1)}{2}$），表示图的边数。接下来 m 行，每行输出两个正整数，第 i 行的两个正整数表示第 i 条边两个点的编号。若有多种满足题意的图，你可以输出任意一种。

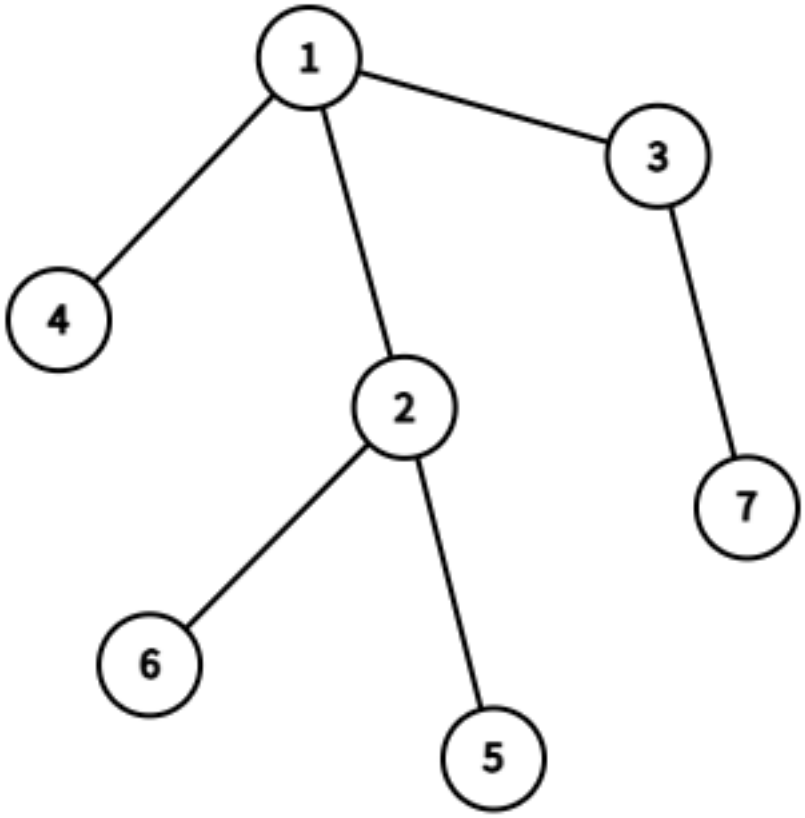
Example

standard input	standard output
2 4 11 7 11000	-1 6 1 2 1 3 1 4 2 5 2 6 3 7

Note

对于样例一，可以证明不存在满足题意的图。

对于样例二，图如下：



其中点 $1, 2, 3$ 是割点，$\deg_1 \sim \deg_7$ 分别为：$3, 3, 2, 1, 1, 1, 1$，符合题意。

Problem J. 构造大师贝贝

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

贝贝励志成为FJ-ACM中最强的构造选手，于是他每日苦练构造题。
为了检验贝贝的训练成果，宁宁同学提出了一道十分甚至九分困难的构造题来检验他的学习成果：
给定一个正整数 n，请在 100 次操作以内，将其变为一个完全平方数。每次操作的内容为如下：

- 选择一个当前数字 n 的正约数 x，即 $n \mod x = 0$；
- 每次选择的 x 需跟之前**任何一次**选择的都不同；
- 随后让 n 加上 x。

在整个操作过程中，n 不允许超过 10^{18}。
这可难坏了贝贝，于是他找到了无比聪明的你来解决这个问题。
其中，$n \mod x$ 表示 n 除以 x 的余数。

Input

本题包含多组评测用例。
第一行，包含一个整数 $T(1 \le T \le 10^3)$，表示评测用例组数。
接下来的 T 行，一行一个正整数 $n(1 \le n \le 10^{12})$。

Output

对于每组评测用例：
第一行，输出一个整数 $m(0 \le m \le 100)$，表示操作次数。
第二行，包含 m 个由空格分隔的不同的正整数，表示每次操作所加上的数字 x。
在整个操作过程中，你需要保证 n 不超过 10^{18}。

Example

standard input	standard output
2 2025 182	0 3 7 3 4

Note

对于第 1 个评测用例的说明如下：

- 因为 $2025 = 45 \times 45$ 原本就是平方数，故无需操作。

对于第 2 个评测用例的说明如下：

- 第一次操作：
  - 选择 $x = 7$，数字 7 是 182 的因子；

  - 令 $n := 182 + 7 = 189$。

- 第二次操作:
  - 选择 $x = 3$，数字 3 是 189 的因子且 3 不在之前选择的数字集合 $\{7\}$ 中;
  - 令 $n := 189 + 3 = 192$;
- 第三次操作:
  - 选择 $x = 4$，数字 4 是 192 的因子且 4 不在之前选择的数字集合 $\{7, 3\}$ 中;
  - 令 $n := 192 + 4 = 196$;
  - $196 = 14 \times 14$ 是完全平方数，结束。

其中 := 表示赋值符号。

Problem K. VERTeX

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

给定一棵有 n 个结点的树，结点依次以 $1, 2, \ldots, n$ 标号。第 i（$1 \le i \le n$）个结点有**正整数**权值 $b_i > 0$。对于连接结点 u 与 v 的树边，其边权为 $w_{uv} = b_u + b_v$。

现在给定树的形态与每条树边的边权，你需要判断是否存在满足条件的一组结点权值。若存在，则求出任意一组结点权值。

Input

第一行，一个正整数 n（$1 \le n \le 2 \times 10^5$），表示结点数量。

接下来 $n-1$ 行，每行三个整数 u, v, w_{uv}（$1 \le u, v \le n$，$1 \le w_{uv} \le 10^9$），表示一条连接结点 u 与 v 的边权为 w_{uv} 的树边。

Output

若满足条件的一组结点权值不存在，则输出一行一个字符串 `NO`。

否则，第一行输出一个字符串 `YES`，第二行输出 n 个正整数 $b_1, \ldots, b_n$，表示你求出的一组结点权值。你需要保证对于任意 $1 \le i \le n$ 有 $b_i \le 10^9$。

若存在多组满足条件的答案，输出任意一组均可。

Examples

standard input	standard output
5 1 2 5 1 3 4 2 5 7 3 4 2	YES 3 2 1 1 5
4 1 2 5 2 3 9 3 4 4	NO

Note

对于第一组样例，可以验证给出的权值满足条件。注意到 $w_{34} = b_3 + b_4 = 2$，因此 b_3 与 b_4 只能取 1，继而可以确定其他结点的权值。

对于第二组样例，注意到 $b_2 + b_3 = w_{23} = 9 = w_{12} + w_{34} = b_1 + b_2 + b_3 + b_4$，从而 $b_1 + b_4 = 0$，而这与 $b_1 > 0$ 且 $b_4 > 0$ 矛盾，因此不存在满足条件的结点权值。

Problem L. 众数

Input file:	standard input
Output file:	standard output
Time limit:	3 seconds
Memory limit:	512 megabytes

有 n 个整数构成的序列 a_i（下标 $1 \sim n$），对于序列的每个前缀 $a_1, a_2, \cdots, a_k$（$1 \le k \le n$），考虑选择其中一个非空下标集合 S，定义 $f(S) = \min\{a_i \mid i \in S\} + \max\{a_i \mid i \in S\}$。

求对于序列 a 的每个前缀，在其对应的 $2^k - 1$ 种情况下，$f(S)$ 的值的众数（出现最多的数），如果有多种数出现次数一样均为最多，则输出其中最大的数。

Input

第一行一个整数 t（$1 \le t \le 1000$），表示接下来有 t 组测试数据。

接下来对于每一组测试数据，第一行包含一个整数 n（$1 \le n \le 10^6$），表示整数的个数。

接下来一行包含 n 个整数表示 a_i（$1 \le a_i \le 10^9$）。

保证所有测试数据的 n 的总和不超过 10^6。

Output

对于每一组数据输出一行，包含 n 个由空格隔开的整数，依次表示每一个前缀的 $f(S)$ 的值的最大众数。

Example

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

Note

对于第一组样例：

前 1 个数中，2 出现了 1 次，最大众数为 2；

前 2 个数中，2 出现了 3 次，最大众数为 2；

前 3 个数中，2 出现了 3 次，5 出现了 3 次，8 出现了 1 次，最大众数为 5；

前 4 个数中，2 出现了 3 次，5 出现了 3 次，6 出现了 6 次，8 出现了 1 次，9 出现了 1 次，10 出现了 1 次，最大众数为 6；

前 5 个数中，2 出现了 7 次，5 出现了 7 次，6 出现了 14 次，8 出现了 1 次，9 出现了 1 次，10 出现了 1 次，最大众数为 6；

前 6 个数中，2 出现了 7 次，5 出现了 21 次，6 出现了 28 次，8 出现了 3 次，9 出现了 3 次，10 出现了 1 次，最大众数为 6。

Problem M. 致谢

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

亲爱的参赛者:

衷心感谢你参与第十二届福建省大学生程序设计竞赛暨福建邀请赛。

同时，向在比赛幕后默默付出的教练老师和赛事组织人员致以诚挚的谢意，正是因为你们的辛勤付出，比赛才能顺利进行。

此外，也要感谢历届福建省赛(FJCPC)的承办院校为该系列赛事所做出的巨大贡献。

接下来是历届福建省赛(FJCPC)的承办院校及其比赛时间的具体信息:

届数	比赛时间	承办院校	英文名	英文缩写
1	2010-5-23	福州大学	Fuzhou University	FZU
2	2011-5-15	福建师范大学	Fujian Normal University	FNU
3	2012-12-02	福州大学	Fuzhou University	FZU
4	2013-12-15	福州大学	Fuzhou University	FZU
5	2014-11-30	福建农林大学	Fujian Agriculture and Forestry University	FAFU
6	2015-12-20	华侨大学	Huaqiao University	HQU
7	2016-5-22	闽江学院	Minjiang University	MJU
8	2017-5-17	厦门理工学院	Xiamen University of Technology	XMUT
9	2018-6-10	泉州师范学院	Quanzhou Normal University	QNU
10	2019-5-19	集美大学	Jimei University	JMU
11	2024-5-26	福州大学	Fuzhou University	FZU

小A希望你能输出第n届FJCPC的承办院校英文缩写。

再次向所有为第十二届福建省大学生程序设计竞赛暨福建邀请赛奉献力量的每一个人表示最衷心的感谢！

也祝愿你能比赛中取得理想的成绩！

Input

输入一个正整数$n(1 \le n \le 11)$。

Output

输出第n届FJCPC的承办院校英文缩写。

Examples

standard input	standard output
1	FZU
2	FNU

Note

由表可知，第一届FJCPC承办校是福州大学，第二届FJCPC承办校是福建师范大学。