兰州大学第一届“飞马杯”程序设计竞赛

题目

2021.05.29

规则	ACM 个人赛
语言	C/C++、Java、Python
题目	12
时长	5 小时

为了响应教育部提出“新工科”教育战略，培养和展示我大学生分析、解决问题和计算机编程的能力，鼓励和培养创新思维，丰富校园学术气氛，造就具有综合素质的面向 21 世纪的计算机人才，兰州大学将于 2021 年 05 月 29 日举办兰州大学第一届“飞马杯”程序设计竞赛，面向兰州大学全体学生。

(感谢兰州大学信息科学与工程学院马俊老师提供的技术支持)

Accepted Wrong Answer Runtime Error Time Limit Exceeded
Memory Limit Exceeded Compile Error

目录

A - ★★比赛新机制★★

时间限制： 1000ms
空间限制： 256MB

题目描述

最近，$Allen$ 教授研发了一种新的比赛机制，这种比赛机制是 $ACPC$ ($Asia\ Collegiate\ Programming\ Contest$) 赛制的扩展，简称 $AUPC$ ($Asia\ University\ Programming\ Contest$)。参赛者为个人参赛(即每人一台电脑)，且成绩分为两部分，过题数和罚时。每道题目的罚时 s 与比赛已经开始的时间(按分钟计) a 以及该题错误的提交次数(不包括编译错误) b 有关，即 $s = a + b \times 20$。

另外，新赛制规定，每位参赛者必须按一定的顺序循环做题，直到解出全部题目或比赛结束。在比赛开始之前，每位参赛者可以任意选择第一个要做的题目和方向：正序或者逆序，但是中途不能跳题或者改变方向。例如，现在有 A, B, C, D, E 五道题目，参赛者可以选择第一个要做的题目 C，然后按正序 C, D, E, A, B 的顺序做题，或者按逆序 C, B, A, E, D 的顺序做题，而 C, D, B, A, E 或 C, E, A, B, D 都是不被允许的。

$Alice$ 水平非常高，她总是可以做出所有的题目。现在她想知道，如果已知解决每道题需要花费的时间，那么解决 n 道题的最少罚时是多少。

我们认为比赛时间足够 $Alice$ 解决所有题目，在比赛开始的瞬间 $Alice$ 就开始做题并且所有题目都是一次提交便成功通过。

输入

第一行一个整数 T $(1 \le T \le 5 \times 10^5)$，表示测试用例的数量。

对于每组测试用例，第一行一个整数 n $(1 \le n \le 5 \times 10^5)$，表示题目的数量；

接下来一行 n 个整数，第 i $(1 \le i \le n)$ 个整数 A_i $(1 \le A_i \le 5 \times 10^5)$，表示解决第 i 道题需要花费的时间(按分钟计)。

对于全部测试用例，保证 $\sum n \le 5 \times 10^5$。

输出

对于每组测试用例，输出一行一个整数表示答案。

样例

样例输入	样例输出
2 5 2 1 5 3 4 10 5 9 3 7 8 1 10 4 6 2	36 277

提示

在第一组测试用例中，$Alice$ 按题目编号 $2, 1, 5, 4, 3$ 的顺序做题，总罚时最少，为 $1 + 3 + 7 + 10 + 15 = 36$。

在第二组测试用例中，总罚时最少为 $5 + 7 + 13 + 17 + 27 + 28 + 36 + 43 + 46 + 55 = 277$。

B - ★★体育课排队★★

时间限制： 4000ms
空间限制： 256MB

题目描述

今天体育老师终于没有生病了，同学们欢快地提前来到操场准备上体育课。

将操场视为一个二维平面，现在每个同学都在整数坐标表示的位置上玩耍，起初有可能有多位同学处于同一位置。为了避免老师再次被气倒，同学们需要在上课之前分为 m 队，并且每队要按 x 轴正方向整齐地排列，其中第 i 队队首的位置为 (sx_i, sy_i)，排队全部完成之后每支队伍里同学们的位置必须是连续的且不能有多位同学处于同一位置(排队过程中没有该限制)。

例如第 i 队要有 s_i 位同学，则应按任意的顺序分别排在 $(sx_i, sy_i), (sx_i + 1, sy_i), \cdots, (sx_i + s_i - 1, sy_i)$ 这 s_i 个位置上。

在排队开始之前，所有同学会一直停在初始位置。当排队开始时，所有的同学会在这一时刻同时开始移动，在每秒钟，每位同学可以向上下左右四个方向中的任意方向移动一个单位长度，或者留在原地。例如当前位置为 (x, y)，则下一秒可以移动到 $(x, y), (x + 1, y), (x - 1, y), (x, y + 1), (x, y - 1)$ 其中的一个位置。

Bob 身为班级的一员，他想让同学们尽可能多地玩耍，但是也要保证同学们能在体育课开始的瞬间排好队。因此他想知道，在采取最优策略的情况下，至少需要提前多少秒开始排队。当然，同学们可以自由选择自己要去的队伍。

(忽略同学们之间的碰撞体积)

输入

第一行一个整数 $T\ (1 \le T \le 1000)$，表示测试用例的数量。

对于每组测试用例，第一行两个整数 $n, m\ (1 \le m \le n \le 1000)$，分别表示学生的数量和要分的队数；

接下来 n 行，第 $i\ (1 \le i \le n)$ 行两个整数 $x_i, y_i\ (-1000 \le x_i, y_i \le 1000)$，表示第 i 位学生的位置；

接下来 m 行，第 $i\ (1 \le i \le m)$ 行三个整数 $s_i, sx_i, sy_i\ (1 \le s_i \le n, -1000 \le sx_i, sy_i \le 1000)$，分别表示第 i 队要排的人数和队首位置。

对于每组测试用例，保证队首的纵坐标各不相同，且 $\sum s = n$。

对于全部测试用例，保证 $\sum n \le 1000$。

输出

对于每组测试用例，先在第一行输出一个整数表示排好队所需的最短时间；

接下来 m 行，第 $i\ (1 \le i \le m)$ 行 s_i 个整数，表示第 i 队从队首到队尾每个位置学生的编号。

如果有多种答案，你可以输出其中任意一种。

样例

样例输入	样例输出
2 5 1 0 5 5 1 -2 0 3 1 2 2 5 -1 0 10 2 -990 -132 513 41 32 241 274 582 -980 -891 -982 -529 -414 290 721 -565 925 -966 -368 -773 5 554 0 5 -563 4	5 3 1 2 4 5 1333 7 2 4 8 9 3 5 6 1 10

提示

在第一组测试用例中，五位同学可按 $3,1,2,4,5$ 的顺序排成一队，所需时间最短，为 5。

在第二组测试用例中，所需最短时间为 1333。

C - ★★生命的游戏★★

时间限制： 1000ms
空间限制： 256MB

题目描述

生命游戏($Conway's\ Game\ of\ Life$)是英国数学家约翰·何顿·康威在 1970 年发明的一款“零玩家游戏”。

生命游戏的“棋盘”是一个 $n \times n$ 的网格，每个网格代表一个“细胞”，每个细胞有两种状态：“生”或“死”。生命游戏每隔一段时间进行一次演变，称为一个“回合”。

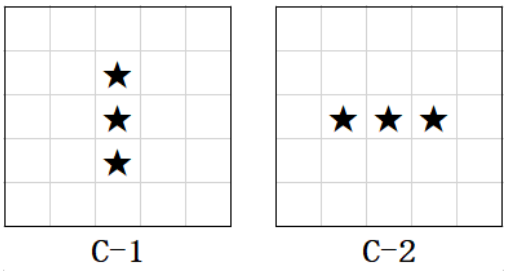
在生命游戏中，每个细胞在一个新回合的生与死只取决于它在上一个回合时周围 8 个细胞的存活状态，具体的规则如下：

- 如果一个死亡的的细胞周围恰好有 3 个活的细胞，那么下一个回合时，这个细胞的状态将转为“生”
- 如果一个存活的的细胞周围活细胞的数量大于 3 或小于 2，那么下一个回合时，这个细胞的状态将转为“死”
- 其他情况下，细胞的存活状态不变

特别地，我们规定第 i 行第 j 列的细胞为 (i,j)，且在“棋盘”边缘的细胞在位置上与另一侧循环相连。例如细胞 (n,n) 的右侧是细胞 $(n,1)$，下方是细胞 $(1,n)$。

$Cindy$ 特别喜欢生命游戏，她常常能够在看似毫无规律的生命游戏中发现许多有趣的现象。例如对于某些初始状态，会在若干回合的演变之后再次回到最初的状态，即所有对应位置的细胞状态完全相同，我们便认为该初始状态存在周期。

例如，图 $C-1$ 在进行一次演变之后会变成图 $C-2$，再进行一次演变后又会重新变为 $C-1$，其周期为 2。(其中 ★ 代表“生”)



对于一个给定的初始状态，$Cindy$ 想知道在 k 回合之内(包括第 k 回合)该状态是否存在周期。

输入

第一行一个整数 $T\ (1 \le T \le 100)$，表示测试用例的数量。

对于每组测试用例，第一行两个整数 $n,k\ (1 \le n,k \le 100)$，分别表示棋盘的大小和回合次数；

接下来 n 行，一个 $n \times n$ 的 01 矩阵，表示初始状态。其中 1 表示存活，0 表示死亡。

对于全部测试用例，保证 $\sum n, \sum k \le 100$。

输出

对于每组测试用例，输出一行一个字符串，如果存在周期，先在第一行输出 `YES`，然后在第二行输出第一次出现周期的回合数；如果不存在周期，直接输出一行 `NO`。

样例

样例输入	样例输出
2 5 2 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 5 50 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1	YES 2 NO

提示

在第一组测试用例中，第一次出现周期的回合数为 2。

在第二组测试用例中，在 50 回合内初始状态不会出现周期。

D - ★★飞马祝福语★★

时间限制： 3000ms
空间限制： 256MB

题目描述

在兰州大学第一届“飞马杯”程序设计竞赛中，我们收到了很多来自同学们的祝福信。我们认为祝福信的任意子序列都是一条祝福语，子序列为原字符串中删去若干字符，剩余字符相对位置不变形成的序列。例如 `ac`，`abcd` 均是 `abcd` 的子序列，而 `ca` 则不是 `abcd` 的子序列。

如果一条祝福语与 `FeiMa` 完全相同(包括大小写)，我们称这条祝福语为“飞马祝福语”。例如，`FeiMa` 是“飞马祝福语”，而 `Feima`，`FeiMaa` 则不是。

然而，令我们非常震惊的是，出现不明身份人员混入命题组，私自重启已关闭的服务器，并且在服务器上多次修改祝福信，被发现后及时制止。为了挽回损失并且更好地感受来自同学们的祝福，现在需要统计祝福信在每次修改之后依然包含多少条“飞马祝福语”。

由于数量可能很大，请对 998244353 取模后作为答案输出。

输入

第一行一个整数 $T\ (1 \leq T \leq 10^5)$，表示测试用例的数量。

对于每组测试用例，第一行两个整数 $n, q\ (1 \leq n, q \leq 10^5)$，分别表示祝福信的长度和修改次数；

接下来一行，一个长度为 n 的字符串 s，表示收到的祝福信；

接下来 q 行，第 $i\ (1 \leq i \leq q)$ 行两个整数 $l_i, r_i\ (1 \leq l_i \leq r_i \leq n)$ 和一个英文字母 x_i，表示不明身份人员将字符串 s 的区间 $[l_i, r_i]$ 中所有字符修改为 x_i。

对于全部测试用例，保证祝福信中仅包含大小写英文字母，且 $\sum n, \sum q \leq 10^5$。

输出

对于每组测试用例的每次修改，输出一行一个整数表示答案。

样例

样例输入	样例输出
1 15 2 FeimaFeiMaFeima 4 6 M 7 10 a	12 15

提示

在第一次修改之后，祝福信变为 `FeiMMMeiMaFeima`，其中包含 12 条飞马祝福语。

在第二次修改之后，祝福信变为 `FeiMMMaaaaFeima`，其中包含 15 条飞马祝福语。

E - ★★序列大团结★★

时间限制： 1000ms
空间限制： 256MB

题目描述

$Ellis$ 热爱集体，向往团结。她喜欢一切团结的事物，例如团结的班级，团结的家庭，甚至是团结的序列。

$Ellis$ 认为，如果对于所有 $1 \leq x \leq \max\{E_i\}$，都恰好满足以下条件之一：

- x 没有在序列 E 中出现过
- x 是 k 的倍数
- x 在序列 E 中的出现次数是 k 的倍数

那么就称序列 E 是团结的，其“团结因子”为 k。

同时，$Ellis$ 认为，对于一个“团结因子”为 k 的序列，其“团结度”为“团结因子”同样为 k 的不同非空连续子序列的数量。

一个序列的连续子序列为原序列中下标连续的任意子序列，即在原序列中截取了一段。例如 $[2,3,4],[1,2,3,4,5]$ 均是 $[1,2,3,4,5]$ 的连续子序列，而 $[1,3,5]$ 则不是。不同的连续子序列指的是从原序列中截取的位置不同，即使不同子序列中所包含数的大小和顺序可能完全相同。

$Ellis$ 想知道，对于一个给定的序列 E 和“团结因子” k，它的“团结度”是多少。

输入

第一行一个整数 $T\ (1 \leq T \leq 10^5)$，表示测试用例的数量。

对于每组测试用例，第一行两个整数 $n, k\ (1 \leq k \leq n \leq 10^5)$，分别表示序列 E 的长度和“团结因子”；

接下来一行 n 个整数，第 $i\ (1 \leq i \leq n)$ 个整数为 $E_i\ (1 \leq E_i \leq 10^9)$。

对于全部测试用例，保证序列 E 在一定程度上随机生成，且 $\sum n \leq 10^5$。

输出

对于每组测试用例，输出一行一个整数表示答案。

样例

样例输入	样例输出
2 5 2 1 1 2 3 3 10 3 1 1 2 3 3 6 2 2 1 9	6 10

提示

在第一组测试用例中，团结因子为 2，团结非空连续子序列有 $[2],[1,1],[3,3],[1,1,2],[2,3,3],[1,1,2,3,3]$，团结度为 6。

在第二组测试用例中，团结度为 10。

F - ★★飞马分隔符★★

时间限制： 1000ms
空间限制： 256MB

题目描述

$Ford$ 非常喜欢 `FeiMa`，如果一个字符串中存在一个子序列为 `FeiMa`，$Ford$ 就会异常兴奋，然而这可能会导致一些无法预料的后果。

子序列为原字符串中删去若干字符，剩余字符相对位置不变形成的序列。例如 `ac`，`abcd` 均是 `abcd` 的子序列，而 `ca` 则不是 `abcd` 的子序列。

为了防止意外发生，你需要将这个字符串 s 按顺序划分为若干部分 $s_1, s_2, \ldots, s_k$，且 $s_1 + s_2 + \cdots + s_k = s$（$+$ 表示字符串的连接），使得每一部分都不包含子序列 `FeiMa`。而每次划分都需要一个“分隔符”，每个“分隔符”都要去商店购买。

现在，你想知道最少需要购买多少个分隔符。

输入

第一行一个整数 T $(1 \leq T \leq 10^5)$，表示测试用例的数量。

对于每组测试用例，第一行一个整数 n $(1 \leq n \leq 10^5)$，表示字符串的长度；

接下来一行一个长度为 n 的字符串 s。

对于全部测试用例，保证字符串 s 中仅包含大小写英文字母和数字且 $\sum n \leq 10^5$。

输出

对于每组测试用例，输出一行一个整数表示答案。

样例

样例输入	样例输出
2 18 FeiMa20210529FeiMa 5 MaFei	2 0

提示

在第一组测试用例中，可将 s 分为 `Fei`、`Ma20210529Fei`、`Ma`，最少需要 2 个分隔符。

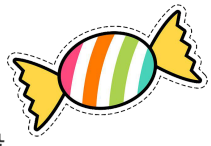
在第二组测试用例中，不需要分隔符。

G - ★★糖果魔法阵★★

时间限制： 2000ms
空间限制： 256MB

题目描述

$Gemma$ 是一名女巫，她擅长使用各种各样的魔法。似乎大部分女巫都喜欢糖果，当然，$Gemma$ 也不例外。



每天，$Gemma$ 都会用她的魔力催动若干次糖果魔法阵，从而得到吃不完的糖果。

具体来说，每次用魔力催动糖果魔法阵获得的糖果数与放入魔法阵中的糖果数有关。假设放入魔法阵中的糖果数为 $x\ (x>0)$，那么用魔力催动一次魔法阵后获得的糖果数为 $\frac{x^4+12x^2+4}{4x(x^2+2)} \bmod 998244353$（原先放入的 x 颗糖果已经作为催动魔法阵的代价永远消失了）。

特别地，如果 $x=0$，那么便意味着此次催动魔法阵获得的糖果数也为 0。

为了得到尽可能多的糖果，$Gemma$ 每次都会把她当前所有的糖果放入魔法阵。尽管有些时候这种做法并不是最优的，但谁让这个女巫傻乎乎的呢。

另外，$Gemma$ 的魔力是有限的，每次催动魔法阵都会消耗她一定的魔力，并且，她必须在每晚通过睡“魔法觉”来恢复她的魔力。

具体来说，假设从第一天开始到第 m 天结束时她总共催动了 k 次魔法阵，且此时拥有的糖果数量为 x，那么在“魔法觉”之后，第 $m+1$ 天她的魔力会恢复为 $(116195172^{4^k}-882049183^{4^k})mx \bmod 998244353$。在任何时刻，魔力都不会为负，并且睡“魔法觉”不会消耗任何糖果。

$Gemma$ 为了糖果不停地努力，只要当天还有足够的魔力，她就会一直催动魔法阵，直到剩余魔力不足以再催动一次魔法阵，她才会去睡她的“魔法觉”。

现在 $Gemma$ 想知道，经过她的不懈努力，在第 n 天结束时，她会拥有多少颗糖果。在一开始，她只有 1 颗糖果。

输入

第一行三个整数 $s,c,q\ (1\le s,c\le 10^9, 1\le q\le 1000)$，分别表示初始魔力，每次催动魔法阵消耗的魔力和询问次数；

接下来 q 行，第 $i\ (1\le i\le q)$ 行一个整数 $n_i\ (1\le n_i\le 2\times 10^7)$，表示询问第 n_i 天结束时拥有的糖果数。

对于全部询问，保证 $\sum n\le 2\times 10^7$。

输出

对于每次询问，输出一行一个整数表示答案。

样例

样例输入	样例输出
10 6 2 1 10	915057325 678018512

提示

在第一次询问中，第一天会催动一次魔法阵，第一天结束时拥有 $\frac{17}{12} \bmod 998244353 = 915057325$ 颗糖果。

在第二次询问中，第 10 天结束时拥有 678018512 颗糖果。

H - ★★温暖的力量★★

时间限制： 1000ms
空间限制： 256MB

题目描述

小时候，$Helen$ 曾感受到一份与众不同的温暖。那份温暖，蕴含着令人重生的力量。即便过去了很久很久，她依然无法忘却。

随着时间的流逝，这份温暖被随机分成了 x $(x > 1)$ 份。由于温暖之间的羁绊，每份温暖分到的力量都是质数，并且力量总和与原来相同。

为了重新寻回这份温暖，必须知道 x 最大可能为多少。但是 $Helen$ 思考了很久，也未能如愿。

作为 $Helen$ 的好友，你不想再看到她为此忧伤。因此你需要帮助 $Helen$ 找到这个最大的 x 值。

输入

第一行一个整数 T $(1 \leq T \leq 10^5)$，表示测试用例的数量。

对于每组测试用例，输入一行一个整数 s $(1 \leq s \leq 10^6)$，表示这份温暖最初蕴含的力量。

输出

对于每组测试用例，输出一行一个整数，如果 x 有解，则输出 x，否则输出 -1。

样例

样例输入	样例输出
2 4 5	2 2

提示

在第一组测试用例中，温暖最多可以被分为 2 份，蕴含的力量分别为 $2, 2$。

在第二组测试用例中，温暖最多可以被分为 2 份，蕴含的力量分别为 $2, 3$。

I - ★★平形四边行★★

时间限制： 1000ms
空间限制： 256MB

题目描述

$Isabella$ 非常喜欢四边形，尤其是较为特殊的平行四边形。

众所周知，两组对边分别平行且相等的四边形是平行四边形。$Isabella$ 可以很轻松地从平面中找到 4 个不同的点，使它们能够围成一个平行四边形，但是她并不满足于此。如果是给定的若干个点，再想从中找到满足条件的 4 个点，则会变得十分棘手。$Isabella$ 是一个不喜欢麻烦的人，她定义了一种新规则，两组对边分别相等的图形称为“平形四边行”，即使四个顶点可能共线甚至重合。这看起来似乎少了一些限制，但 $Isabella$ 依然相信这并不是一个简单的问题。

$Isabella$ 想知道，在平面内给定的若干个点中，是否存在 4 个不同的点能够构成“平形四边行”。(即使是两个坐标完全相同的点，只要它们的下标不同，$Isabella$ 就认为这两个点是不同的。例如，即使 $x_i = x_j, y_i = y_j$，但只要 $i \neq j$，那么这两个点就被认为是不同的)

输入

第一行一个整数 n $(1 \leq n \leq 10^5)$，表示平面中点的个数；

接下来 n 行，第 i 行两个整数 x_i, y_i $(-1000 \leq x_i, y_i \leq 1000)$，表示第 i 个点的坐标。

输出

如果存在满足条件的四个点，先在第一行输出 `YES`，然后在第二行依次输出这四个点的下标(没有顺序)，如果有多种答案，你可以输出其中任意一种；如果不存在满足条件的四个点，直接输出一行 `NO`。

样例

样例输入	样例输出
6 270 620 -844 929 326 848 129 -821 242 344 -809 -158	NO
10 -832 178 -988 974 -534 819 -24 -561 -990 -387 -788 755 -980 -698 -692 -887 -976 997 -312 -372	YES 4 7 8 10

提示

在样例一中，不存在可以围成一个“平形四边行”的四个点。

在样例二中，$4, 7, 8, 10$ 四个点可以围成一个“平形四边行”。

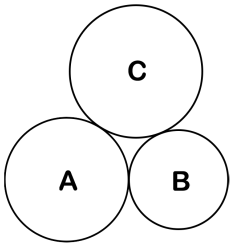
J - ★★翻滚吧硬币★★

时间限制： 1000ms
空间限制： 256MB

题目描述

$Jack$ 是一个喜欢动脑的孩子，在他的生日这天，他收到了一份神奇的礼物：三枚硬币以及一封信。

信中描述了这样一个情景：任取两枚硬币固定在二维平面上，并让它们恰好相切，用第三枚硬币沿着它们形成的边界进行翻滚，即时刻保证与至少一枚已固定的硬币相切。这样这枚运动的硬币在翻滚了一定的圈数之后，一定会回到原点，即恰好绕了一周。（如此一来，便会出现三种情况：固定 A, B，让 C 运动；固定 A, C，让 B 运动；固定 B, C，让 A 运动）



信里还说，如果 $Jack$ 能求出运动的硬币翻滚的圈数最少是多少，那么他将会收获一份更大的礼物。$Jack$ 非常想要这份大礼，但是又不知道如何解决这个难题，因此他向你求助问题的答案。

输入

第一行一个整数 T $(1 \leq T \leq 10^5)$，表示测试用例的数量。

对于每组测试用例，输入一行三个整数 R_1, R_2, R_3 $(1 \leq R_1, R_2, R_3 \leq 10^9)$，分别表示三枚硬币的半径。

输出

对于每组测试用例，输出一行一个实数表示答案。只要你的答案与标准答案的绝对或相对误差不超过 10^{-9} 则被认为正确。

在形式上，若你的答案是 a，标准答案是 b，那么当且仅当 $\dfrac{|a-b|}{\max(1,|b|)} \leq 10^{-9}$ 时，你的答案才会被判定为正确。

样例

样例输入	样例输出
2 1 2 3 5 5 5	1.841387941165222 2.666666666666667

提示

在第一组测试用例中，固定前两枚硬币，用第三枚硬币绕其边界翻滚一周，圈数最少，约为 1.841387941165222。

在第二组测试用例中，固定其中任意两枚硬币，用第三枚硬币绕其边界翻滚一周，圈数最少，约为 2.666666666666667。

K - ★★快乐苹果树★★

时间限制： 1000ms
空间限制： 256MB

题目描述

在果园里，生长着一棵苹果树。这是一棵与众不同的苹果树，它可以根据它感受到的快乐结出不同数量的苹果。

这棵苹果树为 T，树上有 n 个结点，根节点为 1，每个节点都有一个快乐值。在初始时，所有结点的快乐值都是 0。当一个结点的快乐值达到一定大小时，这个结点就会在秋天来临时结出一个苹果。

为了让这棵苹果树尽可能多地结出苹果，园丁 $Kathy$ 决定为它浇水、施肥、修理枝叶等，这样会使得它某些结点的快乐值上升。另外，$Kathy$ 还会不定时地询问某个结点的当前快乐值，从而得知这个结点能否结出苹果。

具体来说，$Kathy$ 会对苹果树进行以下两种操作：

- $1\ F\ K$：从点集 $\{u \mid u \in T, lca(u,F)=F\}$ 中等概率地随机选择一个结点 u，然后再从点集 $\{v \mid v \in T, lca(u,v)=F\}$ 中等概率地随机选择一个结点 v，使得结点 v 获得 K 的快乐值(这里 $lca(a,b)$ 代表 a,b 的最近公共祖先，每个结点的快乐值是累加的)
- $2\ F$：询问结点 F 快乐值的期望

作为园丁，$Kathy$ 可以很轻松地完成第一种操作，却难以求出第二种操作的结果，于是她寻求你的帮助。

为了消除浮点误差，每次询问你只需要告诉 $Kathy$ 相应结点快乐值的期望对 998244353 取模的结果即可。

输入

第一行一个整数 $T\ (1 \le T \le 10^5)$，表示测试用例的数量。

对于每组测试用例，第一行两个整数 $n, q\ (1 \le n, q \le 10^5)$，分别表示树的结点个数和操作次数；

接下来 $n-1$ 行，第 $i\ (1 \le i \le n-1)$ 行两个整数 $u_i, v_i\ (1 \le u_i, v_i \le n)$，表示结点 u 与结点 v 相连；

接下来 q 行，第 $i\ (1 \le i \le q)$ 行三个整数 $1\ F_i\ K_i\ (1 \le F_i \le n, 1 \le K_i \le 10^9)$，表示进行操作一；或者两个整数 $2\ F_i\ (1 \le F_i \le n)$，表示进行操作二。

对于全部测试用例，保证给出的树均是合法的，且 $\sum n, \sum q \le 10^5$。

输出

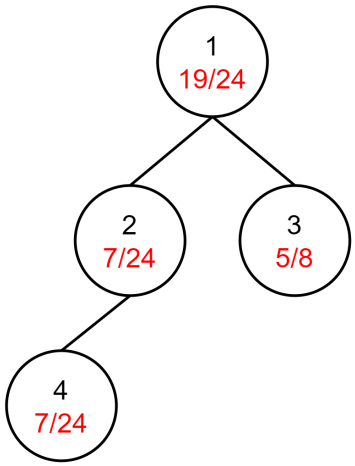
对于每组测试用例的每次操作二，输出一行一个整数表示答案。

样例

样例输入	样例输出
1 4 5 1 2 1 3 2 4 1 1 2 2 1 2 2 2 3 2 4	540715692 41593515 374341633 41593515

提示

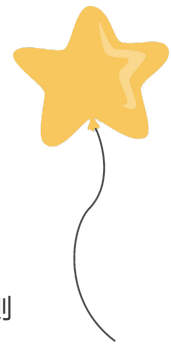
对于第一组测试用例，在进行操作一之后，四个结点快乐值的期望分别为 $\frac{19}{24}, \frac{7}{24}, \frac{5}{8}, \frac{7}{24}$，取模后分别为 $540715692, 41593515, 374341633, 41593515$。



L - ★★星星收集者★★

时间限制： 1000ms
空间限制： 256MB

题目描述



$L.St$ 和 $L.Ts$ 是一对回文好兄弟，他们都喜欢收集 ★。为了比一比谁的收集能力更强，$L.Ts$ 提议进行一场比赛。

$L.Ts$ 准备了 n 个盒子，从左向右依次排列，每个盒子里都放了若干个 ★(可能为 0)。$L.St$ 和 $L.Ts$ 轮流进行操作($L.St$ 先手)，每次操作要在剩下的盒子中从左边数第一个盒子开始按顺序取走若干个盒子(第一个盒子必须取走，且取走的盒子位置必须是连续的)。当所有盒子被取完时，游戏结束。

两人分别计算各自取走的盒子中的 ★ 的总数量，总数量更接近 x 的人获胜；如果相同，则 $L.St$ 获胜。例如，$L.St$ 收集了 a 颗 ★，$L.Ts$ 收集了 b 颗 ★，若 $|a-x| \le |b-x|$，则 $L.St$ 获胜；否则 $L.Ts$ 获胜。

作为发起比赛的人，$L.Ts$ 想要保证自己能够获胜。在比赛开始之前，$L.Ts$ 可以选取若干盒子，并分别额外加入若干颗 ★(可以为 0，不同盒子额外加入的 ★ 数量可以不同)。

但由于害怕额外加入的 ★ 总数量太多被发现，进而被取消成绩，所以 $L.Ts$ 想知道最少需要额外加入多少颗 ★ 才能保证自己获胜(两人均采取最优策略)。

输入

第一行两个整数 n, x $(1 \le n \le 10^5, 0 \le x \le 10^9)$，分别表示盒子的数量和评判标准 x。

接下来一行 n 个整数，第 i $(1 \le i \le n)$ 个整数 L_i $(0 \le L_i \le 10^5)$，表示从左向右第 i 个盒子中 ★ 的初始数量。

输出

输出一行一个整数表示答案。

样例

样例输入	样例输出
3 5 3 4 4	0
5 7 1 2 0 3 2	7

提示

对于样例一，可分为三种情况：

- $L.St$ 取走 3，那么 $L.Ts$ 可以取走 4，之后 $L.St$ 不得不取走最后一个 4，这样 $L.St$ 收集了 7 颗 ★，而 $L.Ts$ 收集了 4 颗 ★，由于 $|7-5|>|4-5|$，于是 $L.Ts$ 获胜。
- $L.St$ 取走 $3,4$，那么 $L.Ts$ 取走最后一个 4，这样 $L.St$ 收集了 7 颗 ★，而 $L.Ts$ 收集了 4 颗 ★，由于 $|7-5|>|4-5|$，于是 $L.Ts$ 获胜。
- $L.St$ 取走 $3,4,4$，这样 $L.St$ 收集了 11 颗 ★，而 $L.Ts$ 收集了 0 颗 ★，由于 $|11-5|>|0-5|$，于是 $L.Ts$ 获胜。

因此 $L.Ts$ 始终拥有必胜策略，不需要加入额外的 ★。

对于样例二，$L.Ts$ 可以在第 1 个盒子中额外加入 3 颗 ★，在第 3 个盒子中额外加入 4 颗 ★，最少需要额外加入 7 颗 ★，这时自己拥有必胜策略。