Problem A. Entangled Coins

输入文件: 标准输入
输出文件: 标准输出
时间限制: 1s
内存限制: 512MB

给定 n 枚有两面（朝下或朝上）的硬币，其中有 s 枚朝上，其余的朝下。

你可以操作硬币**任意次（包括零次）**；在每次操作中，你应任选**恰好** k 枚硬币进行翻转（朝上变为朝下，反之亦然）。

你的目标是将朝上的硬币数量从 s 变为 t。输出所需的**最少**操作次数或报告无解。

输入格式

第一行含测试用例的数量 t ($1 \leqslant t \leqslant 2 \times 10^5$)。测试用例格式如下：

每个测试用例仅占一行，含四个整数 n, k, s, t ($1 \leqslant k \leqslant n \leqslant 10^9, 0 \leqslant s, t \leqslant n$)，含义如上。

输出格式

对于每个测试用例，输出仅占一行：

如果有解，输出一个整数表示最少的操作次数；否则输出 -1 表示无解。

样例

标准输入	标准输出
8 8 3 4 7 9 7 1 0 16 15 1 0 4 2 3 3 6 6 2 4 7 6 2 5 98257693 98257692 24 43850682 98257693 98257692 24 43850681	1 5 15 0 1 -1 43850658 -1

Problem B. Ice Pigeon vs. Fire Pigeon: Extra Training!

输入文件: 标准输入
输出文件: 标准输出
时间限制: 5s
内存限制: 1024MB

自从冰鸽教会了火鸽 KMP 算法后，火鸽深受启发，立志要成为像冰鸽一样的字符串大师。

火鸽问冰鸽如何才能在字符串问题上达到高水平，冰鸽回答说：加训!

冰鸽每天都在字符串问题上刻苦加训。

今天他正在解决一个这样的问题：

给定一个长度为 n 的字符串 S。你需要精确地执行以下操作 k 次：

1. 对于第 i 次操作 $(1 \leqslant i \leqslant k)$，选择 S 的一个子串作为 S_i。

$S[l_i, r_i]$ 是 S 的子串。(即，从位置 l_i 开始到位置 r_i 结束的 S 的连续子串，其中 $1 \leqslant l_i \leqslant r_i \leqslant n$)。此外，空字符串也是 S 的一个子串。

2. 将选择的子串 $S_i(1 \leqslant i \leqslant k)$ 按顺序拼接起来，形成一个新字符串 $S_1 + S_2 + ... + S_k$。

设 X 是通过这种方式拼接 k 个子串可以形成的不同字符串的数量。输出 X 对 998244353 取模的结果。

输入格式

第一行包含两个整数 n 和 k $(1 \leqslant n \leqslant 5 \times 10^5, 1 \leqslant k \leqslant 10^9)$。

第二行包含一个长度为 n 的字符串 S，仅由大小写英文字母组成。

注意，大写和小写字母被视为不同的字符。

输出格式

输出一个整数，表示通过拼接 k 个子串（按第 1 次到第 k 次操作的顺序选择）可以形成的不同字符串的数量，结果对 998244353 取模。

样例

标准输入	标准输出
2 2 ab	12
3 3 abb	96
2 10 Aa	28656
9 6 IcePigeon	811212467

提示

对于第一个样例，”ab” 的子串有：”” (空字符串)、”a”、”b” 和”ab”。

(1) “” + “” = “”

(2) “” + “a” = “a”

(3) “” + “b” = “b”

(4) “” + “ab” = “ab”

(5) “a” + “” = “a”

(6) “a” + “a” = “aa”

(7) “a” + “b” = “ab”

(8) “a” + “ab” = “aab”

(9) “b” + “” = “b”

(10) “b” + “a” = “ba”

(11) “b” + “b” = “bb”

(12) “b” + “bab” = “bab”

(13) “ab” + “” = “ab”

(14) “ab” + “a” = “aba”

(15) “ab” + “b” = “abb”

(16) “ab” + “ab” = “abab”

在上述十六个字符串中，只有十二个是不同的：

“”,“a”,“b”,“ab”,“aa”,“aab”,“ba”,“bb”,“bab”,“aba”,“abb”,“abab”

Problem C. Array Deletion Game

输入文件: 标准输入
输出文件: 标准输出
时间限制: 3s
内存限制: 512MB

Alice 和 Bob 在一个长度为 N 的数组 A 上玩一个游戏，数组中所有元素都是正整数。规则如下：

1. 玩家轮流行动，Alice 先手。
2. 在每个回合，玩家可以：
   - 移除数组最左边的元素，或
   - 移除数组最右边的元素。
3. 如果在一名玩家移动后，剩余元素的和 $\leqslant s$，则该玩家输掉游戏。

给定初始数组，你需要处理 Q 次查询。对于每次查询给出的不同 s，判断 Alice 是否有必胜策略。

输入格式

第一行包含一个整数 N $(1 \leqslant N \leqslant 10^5)$，表示数组的长度。
第二行包含 N 个整数 A_i $(1 \leqslant A_i \leqslant 10000)$，表示数组的元素。
第三行包含一个整数 Q $(1 \leqslant Q \leqslant 10^5)$，表示查询的次数。
接下来的 Q 行，每行包含一个整数 s $(1 \leqslant s < \sum A)$，表示当前查询的阈值。

输出格式

对于每次查询 s，输出一行：

- 如果 Alice 有必胜策略，输出 “Alice”。
- 否则，输出 “Bob”。

样例

标准输入	标准输出
5 1 3 5 7 9 3 10 15 20	Alice Alice Bob

Problem D. Prime XOR Permutation

输入文件: 标准输入
输出文件: 标准输出
时间限制: 2s
内存限制: 512MB

给定一个整数 N，你需要构造一个从 0 到 $N-1$ 的整数排列 P，使得对于所有 $1 \leqslant i < N$，$P_i \oplus P_{i+1}$ 都是一个质数。$\oplus$ 表示按位异或操作。

输入格式

第一行含一个整数 T $(1 \leqslant T \leqslant 2 \times 10^5)$，代表测试用例的数量。

每组测试用例仅含一个整数 N $(1 \leqslant N \leqslant 10^6)$。

保证所有测试用例 N 的总和不超过 10^6。

输出格式

对于每组测试用例：

- 如果存在某个满足条件的的排列，输出一行，包含 N 个由空格分隔的整数，表示该排列 P。
- 如果不存在这样的排列，输出 -1。

样例

标准输入	标准输出
2 4 5	3 1 2 0 4 1 3 0 2

Problem E. Mysterious XOR Operation

输入文件: 标准输入
输出文件: 标准输出
时间限制: 1s
内存限制: 512MB

我们定义一种特殊的神秘异或操作 $\oplus_m$，规则如下：
对于两个数 a 和 b，首先计算它们的常规异或结果 $c = a \oplus b$。然后按以下方式处理 c 的二进制表示：

1. 从最低有效位到最高有效位扫描 c 的二进制表示
2. 初始化一个计数器 $count = 0$
3. 对于每一位：
   - 如果该位是 1：
     - 将 $count$ 加 1
     - 如果 $count$ 是奇数，保留该位
     - 如果 $count$ 是偶数，清除该位（设为 0）
   - 如果该位是 0，保持不变

示例：$(101001)_2 \oplus_m (10010)_2 = (101001)_2$
给定一个长度为 N 的数组 A，计算所有无序对 (i, j)（其中 $i \neq j$）A_i 和 A_j 的神秘异或结果之和。形式化地，其可以被表示为 $\sum_i^N \sum_{j>i}^N A_i \oplus_m A_j$。

输入格式

第一行包含一个整数 N $(2 \leqslant N \leqslant 10^5)$。
第二行包含 N 个整数 A_i $(0 \leqslant A_i \leqslant 10^8)$。

输出格式

输出一个整数，表示所有无序对的神秘异或结果之和。

样例

标准输入	标准输出
3 5 3 9	8

提示

$5 \oplus_m 3 = 2$
$5 \oplus_m 9 = 4$
$3 \oplus_m 9 = 2$
所以答案等于 $4 + 2 + 2 = 8$

Problem F. Grid Survival

输入文件: 标准输入
输出文件: 标准输出
时间限制: 2s
内存限制: 512MB

Alice 和 Bob 在一个有 n 行 m 列的棋盘上用一个棋子玩游戏。游戏规则如下：

- 称第 i 行第 j 列的格子位置为 (l, c) ，当 l 和 c 的奇偶性相同时该格子被涂为**白色**，不同时则被涂为**黑色**。
- 棋子必须始终位于某个格子内；换而言之，棋子的位置也可以用一个整数对 (l, c) 表示，表示棋子在第 l 行第 c 列的格子内，且总有 $1 \leqslant l \leqslant n, 1 \leqslant c \leqslant m$。
- 棋盘上预先给定了 k 个特殊格，第 i 个特殊格的位置为 (l'_i, c'_i) ，并拥有一个正整数权值 w_i 。
- 游戏会持续若干回合（从第 1 回合开始）：
  - 初始时，棋盘为空，棋子尚未放置；
  - 在第一回合，Bob 在白色与黑色间选择一个颜色 col ，且对于每一个（第 i 个）特殊格，他可以以 w_i 的代价**激活**该格子，或不消耗代价且**不激活**该格子；
  - 在第二回合，Alice 选择颜色为 col 的任意一个格子放置棋子；
  - 在第三回合及之后的所有奇数回合，Bob 必须将棋子沿行移动 1 个单位，即从位置 (l, c) 移动到 $(l+1, c)$ 或 $(l-1, c)$。注意移动后仍有 $1 \leqslant l \leqslant n$，且 Bob 不能在该回合不移动棋子。
  - 在第四回合及之后的所有偶数回合，Alice 必须将棋子沿列移动 1 个单位，即从位置 (l, c) 移动到 $(l, c-1)$ 或 $(l, c+1)$。类似地，移动后仍有 $1 \leqslant c \leqslant m$，且 Alice 不能在该回合不移动棋子。
- 无论何时，一旦棋子落在任何一个**被激活**的格子，即判 Alice 输 Bob 获胜；否则如果游戏持续 10^{100} 回合（或永远进行下去），判 Bob 输 Alice 获胜。

假设两名玩家都足够聪明，并给定棋盘的参数（n 和 m）和各特殊格的参数（l'_i, c'_i 和 w_i）；对于每个被选择的颜色，请计算 Bob 为获胜激活若干特殊格所需的最小总代价，或报告 Bob 不可能获胜。

注意：你需要回答多组询问。

输入格式

第一行包含测试用例的数量 t ($1 \leqslant t \leqslant 2 \times 10^5$) 。测试用例格式如下：

第一行包含三个整数 n, m, k ($2 \leqslant n, m \leqslant 10^6, 0 \leqslant k \leqslant \min(2 \times 10^5, n \times m)$)，含义如上；

接下来 k 行，第 i 行包含三个整数 l'_i, c'_i ($1 \leqslant l'_i \leqslant n, 1 \leqslant c'_i \leqslant m$) 和 w_i ($1 \leqslant w_i \leqslant 10^9$) ，表示第 i 个特殊格子所在的行和列及权值；保证在同个测试用例中，没有任何两个特殊格子在同一位置 ($\forall\ 1 \leqslant i < j \leqslant k \Rightarrow l'_i \neq l'_j \vee c'_i \neq c'_j$)。

且保证每个测试文件中 k 的总和不超过 2×10^5 ($\sum k \leqslant 2 \times 10^5$)。

输出格式

对于每个测试用例，答案独占一行，含两个整数，依次代表选择颜色为白色/黑色的答案：若 Bob 可能获胜则输出最小总代价，不可能则输出 -1 。

样例

标准输入	标准输出
7 2 3 4 1 1 52 1 2 33 2 1 47 2 2 95 2 3 2 1 2 90 1 3 30 5 5 4 4 2 30 4 4 10 2 4 40 2 2 20 6 6 4 6 2 10 6 4 10 1 3 10 1 5 10 5 6 4 5 2 10 5 4 10 1 3 10 1 5 10 11 5 5 2 1 1 2 3 1 6 3 1 10 3 1 10 5 1 2 2 0	33 33 90 90 40 30 -1 40 40 40 5 -1 -1 -1

Problem G. Line of Sight

输入文件: 标准输入
输出文件: 标准输出
时间限制: 8s
内存限制: 512MB

SATSKY 将月球猫猫的故事改编为了一个几何算法问题，以纪念与他的队友们共度的难忘岁月。
形式化地：

- 给你一个 n 边形 P，其每个顶点 $P_{1\sim n}$ 的坐标均为整数，并按**逆时针**顺序给出；
- 再给你两个整数坐标点 A 和 B，保证它们在多边形内部（**不含边界**）；
- 称“点 X 能（在 P 中）从点 Y 被看到”**当且仅当**连接两点的整个线段，**两个端点除外**，严格位于 P 的内部（自然地，也不与 P 的任何边相交）；
- 你需要求出：
  - 对于多边形的每个顶点（i 从 1 到 n），它（P_i）能否从点 A 被看到？
  - 对于多边形的每个顶点（i 从 1 到 n），它（P_i）能否从点 B 被看到？
  - 是否存在一个在多边形内部（**不含边界**）的点 C，可以同时看到 A 和 B？

输入格式

第一行包含测试用例的数量 t ($1 \leqslant t \leqslant 2 \times 10^5$) 。测试用例格式如下：
第一行包含一个整数 n ($3 \leqslant n \leqslant 2 \times 10^5$)，表示 P 的顶点数；
第二行包含四个整数 x_A, y_A, x_B, y_B，表示点 A 和点 B 的 x 和 y 坐标；
接下来 n 行，第 i 行包含两个整数 x_{P_i}, y_{P_i}，表示点 P_i 的 x 和 y 坐标。
保证：

- $P_{1\sim n}$ 按**逆时针**顺序给出，且必然构成一个多边形；
- 对于 A、B 以及 P 的每个顶点 ($P_{1\sim n}$)，所有坐标满足 $|x|, |y| \leqslant 10^9$；
- 每个测试文件中 n 的总和不超过 10^6 ($\sum n \leqslant 10^6$)。

强调：

- P 中相邻的边可能共线，不相邻的边也一样；
- A、B 和 $P_{1\sim n}$ 中的任意三点可能在同一条直线上，A 和 B 甚至可能重合；
- 你选择的 C 点的坐标可以是**小数**。

输出格式

对于每个测试用例，输出三行答案：
第一行，输出 n 个用空格分隔的整数：如果点 A 能看到 P_i，则第 i 个整数为 1，否则为 0；
第二行，输出 n 个用空格分隔的整数：如果点 B 能看到 P_i，则第 i 个整数为 1，否则为 0；
第三行，如果存在满足上述要求的点 C，输出字符串“Yes”（不含引号）；否则输出字符串“No”（不含引号）。

样例

标准输入	标准输出
2 8 -1 1 1 -1 2 -1 2 1 1 2 -1 2 -2 1 -2 -1 -1 -2 1 -2 13 1 1 6 1 0 1 1 0 2 1 3 1 4 0 5 0 6 0 7 1 6 2 5 1 4 1 3 2 1 2	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 Yes 1 1 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 1 1 1 1 1 1 0 0 0 No

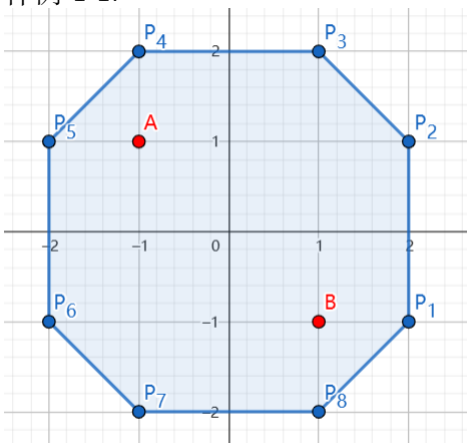
标准输入	标准输出
4 8 -2 0 2 0 -3 -1 -1 -1 -1 1 1 1 1 -1 3 -1 3 2 -3 2 8 -2 0 2 0 -3 -1 -1 -1 -1 1 1 1 1 -1 3 -1 3 3 -3 3 12 -4 -4 4 -4 -2 1 -2 -1 -5 -5 -1 -2 1 -2 5 -5 2 -1 2 1 5 5 1 2 -1 2 -5 5 8 -4 -4 4 -4 -5 2 -2 -1 -5 -5 -1 -2 1 -2 5 -5 2 -1 5 2	1 1 1 0 0 0 0 1 0 0 0 1 1 1 1 0 No 1 1 1 0 0 0 0 1 0 0 0 1 1 1 1 0 Yes 0 1 1 1 0 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 0 1 1 Yes 0 1 1 1 0 0 0 0 0 0 0 0 1 1 1 0 Yes

标准输入	标准输出
2 26 6 20 20 6 23 8 19 6 16 5 13 5 10 6 8 7 7 8 6 10 5 13 5 16 6 19 8 23 5 21 3 19 2 17 1 14 1 11 2 7 3 5 5 3 7 2 11 1 14 1 17 2 19 3 21 5 18 -15 20 15 20 -3 16 -6 15 -15 22 -18 13 -18 7 -21 -2 -21 -8 -18 -14 -6 -20 6 -20 18 -14 21 -8 21 -2 18 7 18 13 15 22 6 15 3 16	0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 No 0 1 1 1 1 0 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 1 1 1 1 0 Yes

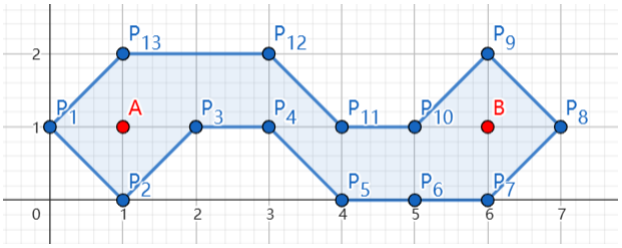
提示

各样例的图示如下：

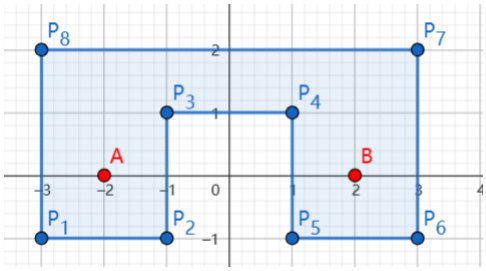
样例 1-1:



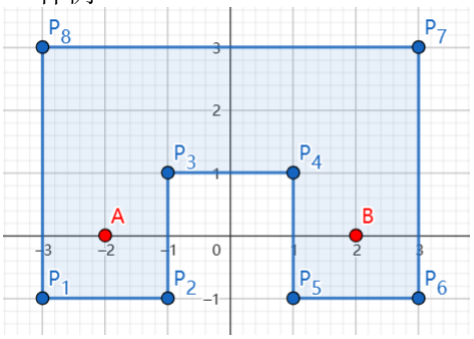
样例 1-2:



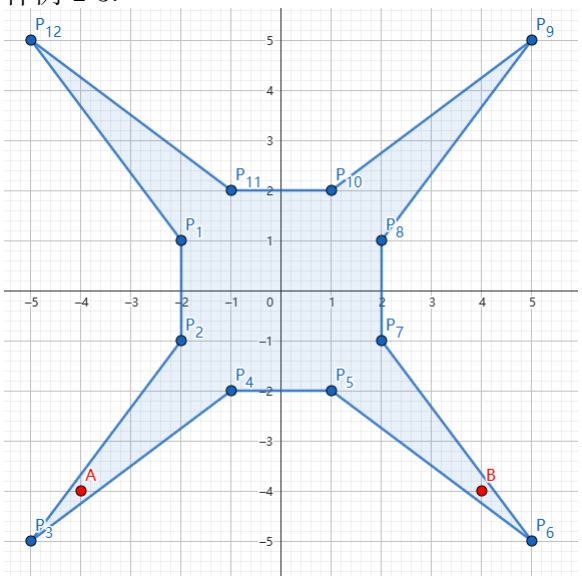
样例 2-1:



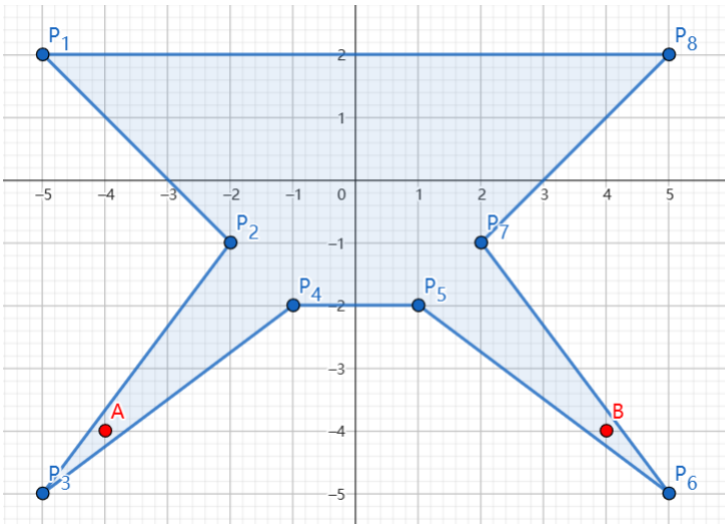
样例 2-2:



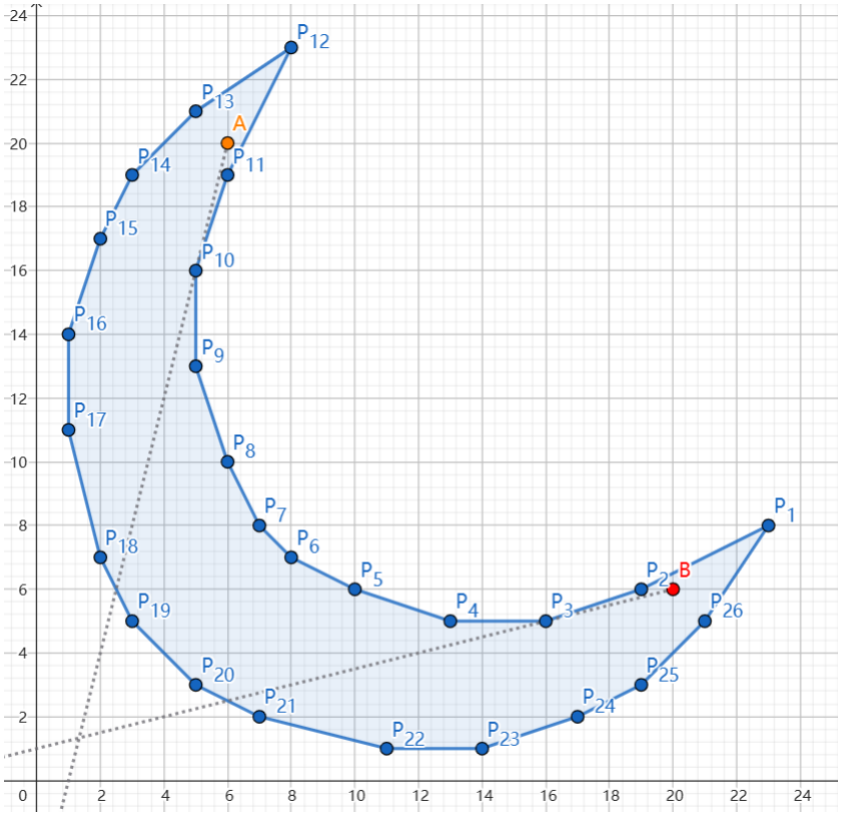
样例 2-3:



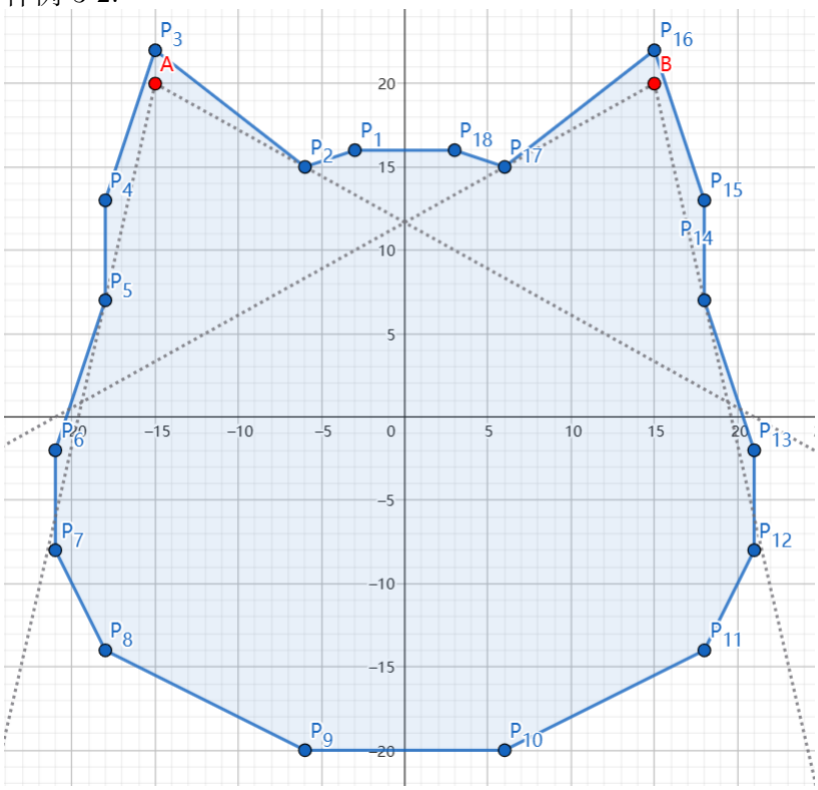
样例 2-4:



样例 3-1:


P12
P13
A
P14
P11
P15
P10
P16
P9
P17
P8
P7
P6
P5
P4
P3
P2
B
P1
P18
P19
P26
P20
P25
P21
P24
P22
P23

样例 3-2:


P3
A
P16
B
P2
P1
P18
P17
P4
P15
P14
P5
P6
P13
P7
P12
P8
P11
P9
P10

Problem H. VI Civilization

输入文件: 标准输入
输出文件: 标准输出
时间限制: 1s
内存限制: 512MB

在六明文游戏中，玩家需要达成科技胜利条件：在 t 回合内累计投入至少 s 点科技点到科技胜利槽。

游戏中共有 n 项科技。游戏初始时，只有第一项科技 $Tech_1$ 处于解锁状态并可被完成，其余科技均处于锁定状态。玩家必须按照 $Tech_1 \to Tech_2 \to \cdots \to Tech_n$ 的固定顺序依次完成科技，不能跳过或改变顺序。具体而言，只有当玩家完成了前 $i-1$ 项科技（即 $Tech_1$ 至 $Tech_{i-1}$）后，第 i 项科技 $Tech_i$ 才会被解锁。

每项科技的完成都需要投入一定数量的科技点。玩家可以投入生产力来触发该科技的”尤里卡”时刻，以减少科技完成所需投入的科技点，**每个科技的“尤里卡”时刻只能触发一次**。完成科技后，玩家每回合获得的科技点将会增加。

每个科技 $Tech_i$ 有四个参数：

- a_i：完成此科技所需的科技点
- k_i：完成后每回合科技点的增量
- b_i：触发尤里卡所需的生产力
- c_i：触发尤里卡后可减少的科技点（$0 \leqslant c_i < a_i$）

六明文是回合制游戏，每回合玩家先获得科技点和生产力，然后进行科技点和生产力的分配。科技点和生产力的分配必须是**完整的（不能拆分到多个任务）**，且当前回合的科技点和生产力**不会保存到下一回合**。

游戏进程如下：

1. 在每个回合开始时，玩家获得：
   - 科技点 m（完成科技 i 后，m 永久增加 k_i）
   - 固定生产力 p（整场游戏保持不变）
2. 接着，玩家进行回合操作：
   - **科技点分配：**
     (a) 将此回合获得的科技点 m 完整投入到已解锁的科技或科技胜利槽。
     (b) 投入科技时，超出部分浪费，且不会用于完成下一个科技。完成第 i 项科技 $Tech_i$ 后，m 永久增加 k_i 。
     (c) 投入科技胜利槽时直接累加。
   - **生产力分配：**
     (a) 将此回合获得的生产力 p **完整（不能拆分到多个科技尤里卡）**投入到**任意科技尤里卡（无论该科技是否已解锁）**。
     (b) 投入科技尤里卡时，超出部分浪费。触发科技尤里卡后，可减少对应科技完成所需的科技点。

求最小的生产力 p（非负整数），使得存在一种操作策略，能在 t 回合内达成科技胜利（科技胜利槽的科技点 $\geqslant s$）。若无法在 t 回合内完成，输出 -1。

输入格式

第一行包含三个整数 m, s, t（$1 \leqslant m \leqslant 100$, $1 \leqslant s \leqslant 10^9$, $1 \leqslant t \leqslant 100$）。

第二行包含一个整数 n（$0 \leqslant n \leqslant 100$）。

接下来 n 行，每行包含四个整数 a_i, k_i, b_i, c_i（$1 \leqslant a_i \leqslant 10^6$, $0 \leqslant k_i \leqslant 1000$, $1 \leqslant b_i \leqslant 10000$, $0 \leqslant c_i < a_i$）。

输出格式

输出最小生产力 p（非负整数）。若无法在 t 回合内完成，输出 -1。

样例

样例

标准输入	标准输出
10 100 9 2 50 10 20 25 60 10 30 20	4
22 970 8 3 85 24 9 27 81 20 85 44 30 80 75 7	-1

提示

样例中，$p = 4$ 的合法策略如下：

回合 1： 获得 10 科技点和 4 生产力。将生产力分配给 $Tech_1$ 的尤里卡，科技点分配给 $Tech_1$。

回合 2： 获得 10 科技点和 4 生产力。将生产力分配给 $Tech_1$ 的尤里卡，科技点分配给 $Tech_1$。

回合 3： 获得 10 科技点和 4 生产力。将生产力分配给 $Tech_1$ 的尤里卡，科技点分配给 $Tech_1$。

回合 4： 获得 10 科技点和 4 生产力。将生产力分配给 $Tech_1$ 的尤里卡，科技点分配给科技胜利槽。

回合 5： 获得 10 科技点和 4 生产力。将生产力分配给 $Tech_1$ 的尤里卡，科技点分配给科技胜利槽。在这个回合，$Tech_1$ 的尤里卡已累计获得 20 生产力，触发了尤里卡。$Tech_1$ 现在完成需要的科技点是 $50 - 25 = 25$。由于已经分配了 30 点科技点，$Tech_1$ 研究完成。每回合获得的科技点增加到 $10 + 10 = 20$。

回合 6： 获得 20 科技点和 4 生产力。将科技点分配给科技胜利槽。

回合 7： 获得 20 科技点和 4 生产力。将科技点分配给科技胜利槽。

回合 8： 获得 20 科技点和 4 生产力。将科技点分配给科技胜利槽。

回合 9： 获得 20 科技点和 4 生产力。将科技点分配给科技胜利槽。科技胜利槽总共积累了 $10(\text{T4}) + 10(\text{T5}) + 20(\text{T6}) + 20(\text{T7}) + 20(\text{T8}) + 20(\text{T9}) = 100$ 点科技点。达成科技胜利！

Problem I. Block Combination Minimal Perimeter

输入文件: 标准输入
输出文件: 标准输出
时间限制: 1s
内存限制: 512MB

给定 n 个矩形方块，其中第 i 个方块的尺寸为 $1 \times i$。你需要将**所有**方块组合成一个实心的矩形（不允许重叠或有空隙）。求出所形成矩形的最小周长。

保证在给定的数据范围下，你总能合成一个实心的矩形。

输入格式

第一行包含一个整数 n ($1 \leqslant n \leqslant 2 \times 10^5$)，表示方块的数量。

输出格式

输出一个整数，表示所形成矩形的最小周长。

样例

标准输入	标准输出
1	4
6	20
10	32

Problem J. Fastest Coverage Problem

输入文件: 标准输入
输出文件: 标准输出
时间限制: 2s
内存限制: 512MB

给定一个 $n \times m$ 的二元矩阵，其中 1 代表黑色单元格，0 代表白色单元格。每一秒，每个黑色单元格会将其上、下、左、右四个相邻的白色单元格变为黑色。

你可以将**最多**一个白色单元格变为黑色（将 0 变为 1），以最小化整个矩阵完全变黑所需的时间。求出这个最短时间。

输入格式

第一行包含两个整数 n 和 m ($1 \leqslant n \times m \leqslant 2 \times 10^5$)，表示矩阵的行数和列数。

接下来的 n 行，每行包含 m 个数字（0 或 1），表示矩阵的初始状态。

输出格式

输出一个整数，表示在增加一个黑色单元格后，使整个矩阵变黑所需的最短时间。

样例

标准输入	标准输出
3 3 0 0 0 0 0 0 0 0 1	2
2 2 1 0 0 0	1
1 5 0 1 0 0 0	1

提示

对于样例 1, 您可以选择位置 (1,1)。

第 0 秒时的矩阵:

1 0 0

0 0 0

0 0 1

第 1 秒时的矩阵:

1 1 0

1 0 1

0 1 1

第 2 秒时的矩阵:

1 1 1

1 1 1

1 1 1

Problem K. Perfect Journey

输入文件: 标准输入
输出文件: 标准输出
时间限制: 2s
内存限制: 512MB

在一个有 n 个城市的国家，有 $n-1$ 条双向道路连接这些城市，形成一棵树。你现在正在这个国家旅游，有 m 条特定的道路是你一定要经过的。旅行社提供了 k 条可选的旅游路线。每条路线从城市 s_i 出发，并沿着最短路径到达城市 t_i。

你的目标是从这 k 条旅游路线中选择尽可能少的路线，确保所有 m 条关键道路都至少被经过一次。

请计算你需要选择的最少旅游路线数量，以及达到这个最少数量有多少种可能的方案，答案对 998244353 取模。一个方案定义为你选择的旅游路线的集合。如果存在一条旅游路线在一个方案中被选择而另一个方案中没有，则认为这两个方案是不同的。

如果无法经过所有特定的道路，输出 -1。

题目保证答案在模 998244353 意义下非零。

输入格式

第一行包含三个整数 $2 \leqslant n \leqslant 2 \times 10^5$, $1 \leqslant m \leqslant 22$, $1 \leqslant k \leqslant 2 \times 10^5$，分别表示城市数量、特定道路数量和旅游路线数量。

接下来的 $n-1$ 行，每行包含两个整数 $1 \leqslant u_i, v_i \leqslant n$，保证给定的图是一棵树。

接下来一行包含 m 个不同的整数 $1 \leqslant x_i \leqslant n-1$，表示特定道路的索引（根据输入顺序）。

接下来的 k 行，每行包含两个整数 $1 \leqslant s_i, t_i \leqslant n$，表示旅游路线从 s_i 到 t_i。

输出格式

输出你需要选择的最少旅游路线数量，以及达到这个最少数量的方案数，答案对 998244353 取模。

如果无法经过所有特定的道路，输出 -1。

样例

标准输入	标准输出
3 2 2 1 2 1 3 1 2 2 3 1 2	1 1
7 3 3 1 2 1 3 2 4 2 5 3 6 6 7 1 3 5 1 4 2 7 2 4	2 2

提示

对于样例 2，我们需要选择至少 2 条路线，有两种可能的方案：

1. 路线 1 和路线 2，路线 1 覆盖道路 1 和道路 3，路线 2 覆盖道路 1 和道路 5。
2. 路线 2 和路线 3，路线 2 覆盖道路 1 和道路 5，路线 3 覆盖道路 3。

Problem L. Float

输入文件: 标准输入
输出文件: 标准输出
时间限制: 1s
内存限制: 512MB

有 n 个关卡，通过第 i 个关卡可从 $i-1$ 号点到 i 号点；你从 0 号点出发，并想要依次通过每个关卡到达 n 号点。每当你尝试一个关卡，你都有 p 的概率通过，和 $(1-p)$ 的概率失败。

你最开始有一些金币，如果你闯关失败且至少有一个金币，你将花一个金币停留在当前点并重试（不会留着金币不用，即使你在 0 号点）；如果你闯关失败且已没有金币剩余，则你将被送回起点（0 号点）。

给定 m，请计算对于从 0 到 m 每一种可能的初始金币数量，你到达终点所需的期望步数，输出时对 998244353 取模。

输入格式

第一行含三个整数 n, m, p $(1 \leqslant n \leqslant 10^9, 1 \leqslant m \leqslant 2 \times 10^5, 0 < p < 998244353)$，依次表示关卡数量、最大金币数量和通过关卡的概率；概率已对 998244353 取模。

输出格式

一行含 $m+1$ 个整数，其中第 i 个数表示初始拥有 $i-1$ 个金币时，到达终点的期望步数对 998244353 取模后的结果。

样例

标准输入	标准输出
2 2 499122177	6 499122182 5
1 3 332748118	3 3 3 3

提示

对于样例 2：

在模 998244353 意义下，数值 332748118 对应概率 $\frac{1}{3}$。

此时结果与 m 无关，因为无论是否有金币，失败时总是返回起点。

Problem M. Mysterious Spacetime

输入文件: 标准输入
输出文件: 标准输出
时间限制: 4s
内存限制: 512MB

在一个神秘的时空中，有 x 种未知的能量，编号从 1 到 x。这些能量在特定的时间和位置出现，遵循以下规则：

1. **时空出现规则**：

- 有 n 个不同的时间点，编号从 1 到 n。
- 在时间点 t，区间 $[l, r]$ 内的所有能量会出现，并在时间点 $t+1$ 消失。

2. **生成规则**：

- 有 m 个生成器，编号从 1 到 m。每个生成器 i 需要区间 $[L_i, R_i]$ 内的所有能量。
- 如果在某个整数时间点 t，区间 $[L_i, R_i]$ 内至少有 k_i 种能量同时出现，那么生成器 i 将在时间点 t 被激活。
- 每个生成器只在最早满足条件的时刻被激活一次。

你需要处理 q 次查询。每次查询给出 `tl tr l r`，询问：

- 在时间范围 $[\mathrm{tl}, \mathrm{tr}]$ 内，最早的时间点 t 是多少，使得存在某个满足 $l \leqslant L_i \leqslant R_i \leqslant r$ 的生成器 i 在时间点 t 被激活。
- 如果不存在这样的 t，输出 -1。

输入格式

第一行包含一个数字 $1 \leqslant T \leqslant 10^5$，表示有 T 组测试用例，每组格式如下：

- 第一行包含四个整数 $1 \leqslant n, m, x, q \leqslant 5 \times 10^5$。
- 接下来的 n 行，每行包含三个整数 $1 \leqslant t \leqslant 10^9$, $1 \leqslant l \leqslant r \leqslant x$，表示在时间点 t，区间 $[l, r]$ 内的东西出现。（题目保证所有时间点都不同）
- 接下来的 m 行，每行包含三个整数 $1 \leqslant L_i \leqslant R_i \leqslant x$, $1 \leqslant k_i \leqslant R_i - L_i + 1$，描述生成器 i 的需求。
- 接下来的 q 行，每行包含四个整数 $1 \leqslant tl \leqslant tr \leqslant 10^9$, $1 \leqslant l \leqslant r \leqslant x$，代表一次查询。

题目保证 $\sum n, \sum m, \sum q, \sum x \leqslant 5 \times 10^5$。

输出格式

对于每次查询，每行输出一个整数，代表答案。

样例

标准输入	标准输出
1 6 6 4 4 1 2 3 5 1 3 3 1 1 6 1 3 2 4 4 4 4 4 1 3 2 1 3 1 1 2 2 1 3 1 1 2 1 4 4 1 1 4 1 2 2 4 1 3 1 4 1 3 3 5 4 4	1 -1 1 -1