Problem A. Fugaku NK-kei

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

There are N regions around Mount Fuji, numbered $1, 2, \ldots, N$ and arranged in a circle in this order. For $1 \le i < N$, region i is adjacent to region $i + 1$, and region N is adjacent to region 1.

jastaway, a famous painter, has created a series of paintings of these regions and Mount Fuji, titled "NK Views of Mount Fuji."

The series contains one painting for each pair of integers (s, d) with $1 \le s \le N$ and $1 \le d \le K$, for a total of NK paintings. The painting corresponding to (s, d) depicts d consecutive regions, starting from region s and going in increasing order of region number, wrapping around from region N to region 1. For example, when $N = 6$, the painting corresponding to $(5, 4)$ depicts regions $5, 6, 1, 2$.

For each painting, one replica is kept in each of the regions it depicts.

tatesoto loves Mount Fuji and has decided to collect one replica of every painting in the series. To do so, he will visit each of the N regions exactly once, in an order of his choice.

Initially, *tatesoto* has no replicas. When he visits region i, he receives one replica of each painting that has a replica kept in region i and that he has not collected yet. Let C_i be the number of replicas he receives in region i.

Find the number of distinct integer sequences $(C_1, C_2, \ldots, C_N)$ that can be obtained over all possible visiting orders. Two sequences $(C_1, C_2, \ldots, C_N)$ and $(C'_1, C'_2, \ldots, C'_N)$ are considered distinct if $C_i \ne C'_i$ for some i with $1 \le i \le N$.

Since the answer can be very large, output it modulo 998244353.

You are given T test cases; solve each of them.

Input

The first line contains an integer T ($1 \le T \le 10^5$), the number of test cases.

Each of the next T lines describes one test case and contains two integers N and K ($2 \le N \le 2 \times 10^5$, $1 \le K \le N - 1$).

The sum of N over all test cases does not exceed 2×10^5.

Output

For each test case, output the answer modulo 998244353 on a separate line.

Example

standard input	standard output
3 4 3 2 1 100000 36000	20 1 216261474

Note

In the first test case, the series "12 Views of Mount Fuji" consists of the following 12 paintings:

- A painting depicting region 1.
- A painting depicting region 2.

- A painting depicting region 3.
- A painting depicting region 4.
- A painting depicting regions $1, 2$.
- A painting depicting regions $2, 3$.
- A painting depicting regions $3, 4$.
- A painting depicting regions $4, 1$.
- A painting depicting regions $1, 2, 3$.
- A painting depicting regions $2, 3, 4$.
- A painting depicting regions $3, 4, 1$.
- A painting depicting regions $4, 1, 2$.

For example, suppose *tatesoto* visits the regions in the order $1, 2, 3, 4$.
In region 1, he receives replicas of the following six paintings:

- The painting depicting region 1.
- The painting depicting regions $1, 2$.
- The painting depicting regions $4, 1$.
- The painting depicting regions $1, 2, 3$.
- The painting depicting regions $3, 4, 1$.
- The painting depicting regions $4, 1, 2$.

Next, in region 2, he receives replicas of the following three paintings, which he has not collected yet:

- The painting depicting region 2.
- The painting depicting regions $2, 3$.
- The painting depicting regions $2, 3, 4$.

Then, in region 3, he receives replicas of the following two paintings:

- The painting depicting region 3.
- The painting depicting regions $3, 4$.

Finally, in region 4, he receives a replica of the painting depicting region 4.
Thus, this visiting order yields $(C_1, C_2, C_3, C_4) = (6, 3, 2, 1)$.
Over all visiting orders, 20 distinct sequences can be obtained.
In the second test case, the series "2 Views of Mount Fuji" consists of the following two paintings:

- A painting depicting region 1.
- A painting depicting region 2.

Whatever the visiting order, *tatesoto* receives exactly one replica in each region, so the only obtainable sequence is $(1, 1)$.

Problem B. PerMutation

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

For positive integers N, K, and x with $1 \leq x \leq N$, and a permutation $P = (P_1, P_2, \ldots, P_N)$ of $(1, 2, \ldots, N)$, define the integer $f(P, K, x)$ as follows.

Initialize a sequence $A = (A_0, A_1, \ldots, A_{N+1})$ of length $N + 2$ with all elements equal to 1. Then, for $i = 0, 1, \ldots, KN - 1$ in this order, perform the following operation:

- Let $j = P_{(i \bmod N)+1}$, and replace A_j with $\left\lfloor \frac{A_{j-1}A_{j+1}}{A_j} \right\rfloor + 1$.

After all operations, let $f(P, K, x)$ be the value of A_x.

You are given a positive integer N and a permutation $P = (P_1, P_2, \ldots, P_N)$ of $(1, 2, \ldots, N)$. Process Q queries in the order they are given. Each query is of one of the following two types.

- Type 1: Given i and j, swap the values of P_i and P_j.
- Type 2: Given K and x, find $f(P, K, x)$ modulo 998244353, where P is the current permutation.

Input

The first line contains an integer N ($2 \leq N \leq 2 \times 10^5$).

The second line contains N integers $P_1, P_2, \ldots, P_N$ ($1 \leq P_i \leq N$). These integers are pairwise distinct.

The third line contains an integer Q ($1 \leq Q \leq 2 \times 10^5$).

Each of the next Q lines describes a query. The k-th of these lines describes the k-th query.

For a type 1 query, the line contains three integers: 1, i, and j, in this order, where $1 \leq i < j \leq N$.

For a type 2 query, the line contains three integers: 2, K, and x, in this order, where $1 \leq K \leq 10^{18}$ and $1 \leq x \leq N$.

There is at least one type 2 query. All input values are integers.

Output

Let q be the number of type 2 queries. Print q lines.

The k-th line should contain the answer to the k-th type 2 query.

Examples

standard input	standard output
4 3 1 2 4 9 2 1 1 2 1 2 2 1 3 2 1 4 1 2 3 2 2 1 2 2 2 2 2 3 2 2 4	2 5 2 3 2 7 5 2
5 3 4 5 2 1 15 1 4 5 1 3 5 2 8 1 2 10 1 2 10 4 2 3 3 2 8 5 1 2 3 2 6 4 2 8 1 1 3 4 2 4 3 1 2 3 2 6 2 2 4 5	1 2 7 10 1 7 1 1 11 2
10 2 3 8 9 10 5 4 7 6 1 17 2 100000 8 2 1000000 4 2 10000000 9 2 100000000 7 2 1000000000 6 2 10000000000 1 2 100000000000 1 2 1000000000000 7 1 5 9 1 7 8 1 6 10 2 10000000000000 5 1 2 7 2 100000000000000 6 2 1000000000000000 6 2 10000000000000000 2 2 100000000000000000 5	18 7 11 43 2 3 4 3 40 39 3 3 22

Note

In sample 1, initially, $P = (3, 1, 2, 4)$.

Queries 1 through 4 have the same P and K, so the sequence A changes in the same way for each of these queries. For each query, start with $A = (1, 1, 1, 1, 1, 1)$ and perform the operations as follows.

- Replace A_3 with $\left\lfloor \frac{A_2 A_4}{A_3} \right\rfloor + 1 = 2$. The sequence becomes $A = (1, 1, 1, 2, 1, 1)$.
- Replace A_1 with $\left\lfloor \frac{A_0 A_2}{A_1} \right\rfloor + 1 = 2$. The sequence becomes $A = (1, 2, 1, 2, 1, 1)$.
- Replace A_2 with $\left\lfloor \frac{A_1 A_3}{A_2} \right\rfloor + 1 = 5$. The sequence becomes $A = (1, 2, 5, 2, 1, 1)$.
- Replace A_4 with $\left\lfloor \frac{A_3 A_5}{A_4} \right\rfloor + 1 = 3$. The sequence becomes $A = (1, 2, 5, 2, 3, 1)$.

Therefore, $f(P, 1, 1) = 2$, $f(P, 1, 2) = 5$, $f(P, 1, 3) = 2$, and $f(P, 1, 4) = 3$.

Query 5 swaps P_2 and P_3, resulting in $P = (3, 2, 1, 4)$.

Queries 6 through 9 have the same P and K, so the sequence A changes in the same way for each of these queries. For each query, start with $A = (1, 1, 1, 1, 1, 1)$ and perform the operations as follows.

- Replace A_3 with $\left\lfloor \frac{A_2 A_4}{A_3} \right\rfloor + 1 = 2$. The sequence becomes $A = (1, 1, 1, 2, 1, 1)$.
- Replace A_2 with $\left\lfloor \frac{A_1 A_3}{A_2} \right\rfloor + 1 = 3$. The sequence becomes $A = (1, 1, 3, 2, 1, 1)$.
- Replace A_1 with $\left\lfloor \frac{A_0 A_2}{A_1} \right\rfloor + 1 = 4$. The sequence becomes $A = (1, 4, 3, 2, 1, 1)$.
- Replace A_4 with $\left\lfloor \frac{A_3 A_5}{A_4} \right\rfloor + 1 = 3$. The sequence becomes $A = (1, 4, 3, 2, 3, 1)$.
- Replace A_3 with $\left\lfloor \frac{A_2 A_4}{A_3} \right\rfloor + 1 = 5$. The sequence becomes $A = (1, 4, 3, 5, 3, 1)$.
- Replace A_2 with $\left\lfloor \frac{A_1 A_3}{A_2} \right\rfloor + 1 = 7$. The sequence becomes $A = (1, 4, 7, 5, 3, 1)$.
- Replace A_1 with $\left\lfloor \frac{A_0 A_2}{A_1} \right\rfloor + 1 = 2$. The sequence becomes $A = (1, 2, 7, 5, 3, 1)$.
- Replace A_4 with $\left\lfloor \frac{A_3 A_5}{A_4} \right\rfloor + 1 = 2$. The sequence becomes $A = (1, 2, 7, 5, 2, 1)$.

Therefore, $f(P, 2, 1) = 2$, $f(P, 2, 2) = 7$, $f(P, 2, 3) = 5$, and $f(P, 2, 4) = 2$.

Problem C. Moving Silver

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

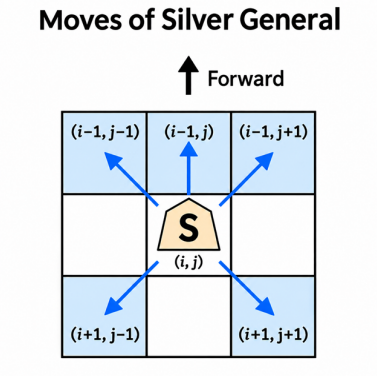
There is a board with H rows and W columns. Let (i, j) denote the cell in the i-th row from the top and the j-th column from the left. Cell (i, j) contains an integer $A_{i,j}$.

You are given Q independent queries. In each query, you are given integers X, Y, N; solve the following problem.

Initially, a silver general (a shogi piece) is placed in cell (X, Y).

The silver general always faces toward the top of the board. When it is in cell (i, j), it can move to any of the following cells that lie within the board:

$$(i-1, j-1), \quad (i-1, j), \quad (i-1, j+1), \quad (i+1, j-1), \quad (i+1, j+1).$$



Its orientation never changes, either when it moves or when the turn passes to the other player.

Alice and Bob take turns moving the silver general, with Alice going first, until exactly N moves have been made in total. On each turn, the current player moves the piece to one of the cells it can move to.

If the piece is in cell (i, j) after the N moves, the score is $A_{i,j}$. Alice wants to maximize the score, and Bob wants to minimize it.

Find the score when both players play optimally.

Input

The first line contains two integers H and W ($2 \le H, W \le 600$, $HW \le 10^5$), the number of rows and columns of the board.

Each of the next H lines contains W integers. The i-th of these lines contains $A_{i,1}, A_{i,2}, \ldots, A_{i,W}$ ($-10^9 \le A_{i,j} \le 10^9$), the integers in the i-th row.

The next line contains an integer Q ($1 \le Q \le 2 \times 10^5$), the number of queries.

Each of the next Q lines describes a query. The i-th of these lines contains three integers X_i, Y_i, N_i ($1 \le X_i \le H$, $1 \le Y_i \le W$, $1 \le N_i \le 10^{18}$), the values of X, Y, N for the i-th query.

Output

Output Q lines. The i-th line should contain the answer to the i-th query.

Examples

standard input	standard output
3 3 2 9 4 7 5 3 6 1 8 4 2 2 1 2 2 2 2 2 3 1 1 5	9 5 9 5
2 2 1000000000 1000000000 -1000000000 -1000000000 1 1 1 1000000000000000000	1000000000

Note

In the first query of the first sample, the silver general starts in cell $(2, 2)$.

Alice can move it to any of the cells $(1, 1)$, $(1, 2)$, $(1, 3)$, $(3, 1)$, and $(3, 3)$, which contain $2, 9, 4, 6, 8$, respectively. By moving the piece to cell $(1, 2)$, she obtains a score of 9. Thus, the answer to this query is 9.

In the second query of the first sample, the silver general starts in cell $(2, 2)$, and Alice and Bob each make one move.

For example, if Alice first moves the piece to cell $(1, 1)$, Bob's only legal move is to cell $(2, 2)$, which gives a score of 5.

Thus, Alice can guarantee a score of at least 5. On the other hand, whatever Alice does, Bob can make sure that the score is at most 5. Hence, the answer to this query is 5.

Problem D. Go Robots

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

You are given a directed acyclic graph with N vertices and $2(N-1)$ edges. For each i ($1 \le i \le N-1$), there are two edges leaving vertex i: $i \to L_i$ and $i \to R_i$. The graph has no self-loops, but may have multiple edges.

Each vertex i ($1 \le i \le N-1$) has a switch S_i. Initially, all switches are set to `L`.

You will release a total of K robots at vertex 1, one after another. When a robot reaches vertex i ($1 \le i \le N-1$), it flips switch S_i and takes one second to move to the vertex indicated by the new switch setting.

More precisely, when a robot reaches vertex i at time t, the switch is updated and the robot moves according to the following rules:

- If $S_i = $ `L`, set $S_i = $ `R`. The robot reaches vertex R_i at time $t+1$.
- If $S_i = $ `R`, set $S_i = $ `L`. The robot reaches vertex L_i at time $t+1$.

The first robot is released at vertex 1 at time 0. For each i ($1 \le i \le K-1$), the $(i+1)$-st robot is released at vertex 1 at the exact moment when the i-th robot reaches vertex N.

Find the time when the K-th robot reaches vertex N.

Solve T test cases.

Input

The first line contains an integer T ($1 \le T \le 10^5$), the number of test cases.

Each test case begins with a line containing two integers N and K ($2 \le N \le 2 \times 10^5$, $1 \le K \le 10^{12}$), the number of vertices and the number of robots.

The next $N-1$ lines describe the outgoing edges. The i-th of these lines contains two integers L_i and R_i ($i+1 \le L_i, R_i \le N$), indicating the edges $i \to L_i$ and $i \to R_i$.

The sum of N over all test cases does not exceed 2×10^5.

All input values are integers.

Output

Print T lines. On the t-th line, print the time when the K-th robot reaches vertex N in the t-th test case.

Example

standard input	standard output
4 3 2 2 3 3 3 2 10 2 2 4 5 2 3 3 4 4 4 10 1000000000000 2 4 3 4 5 6 6 5 8 7 7 8 10 9 9 10 10 10	3 10 11 5000000000000

Note

Consider the first test case of Sample 1.

Initially, the switch settings are `LL`.

At time 0, the first robot is released at vertex 1. Switch S_1 is updated, changing the switch settings to `RL`, and the robot reaches vertex 3 at time 1.

At time 1, the second robot is released at vertex 1. Switch S_1 is updated, changing the switch settings to `LL`, and the robot reaches vertex 2 at time 2.

Then, switch S_2 is updated, changing the switch settings to `LR`. The robot reaches vertex 3 at time 3, completing the process.

Problem E. K-Independent Matrix

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

You are given a positive integer N.

Construct an $N \times N$ matrix A whose entries are integers between 1 and N, inclusive, satisfying the following condition:

- For every $K = 1, 2, \ldots, N$, every $K \times K$ subsquare of A contains exactly one entry that is a multiple of K.

Here, a $K \times K$ subsquare is a submatrix formed by K **consecutive** rows and K **consecutive** columns of A.

It can be proved that such a matrix A always exists under the constraints of this problem.

You are given T test cases; solve each of them.

Input

The first line contains an integer T ($1 \le T \le 10^5$), the number of test cases.

Each of the next T lines describes one test case and contains an integer N ($1 \le N \le 1000$).

The sum of N^2 over all test cases does not exceed 10^6.

Output

For each test case, output a matrix A satisfying the condition in N lines. The i-th line should contain the N space-separated integers $A_{i,1}, A_{i,2}, \ldots, A_{i,N}$.

If there are multiple valid matrices, you may output any of them.

Example

standard input	standard output
3 3 4 1	1 1 2 2 1 1 1 3 2 2 3 1 1 1 1 2 1 2 1 1 1 3 1 4 3 1

Note

In the first test case, the matrix

$$A = \begin{pmatrix} 1 & 1 & 2 \\ 2 & 1 & 1 \\ 1 & 3 & 2 \end{pmatrix}$$

satisfies the condition:

- For $K = 1$, every 1×1 subsquare of A contains exactly one entry that is a multiple of 1.

- For $K = 2$, every 2×2 subsquare of A contains exactly one entry that is a multiple of 2.
- For $K = 3$, the only 3×3 subsquare of A, which is A itself, contains exactly one entry that is a multiple of 3.

Other valid matrices are also accepted.

Problem F. Path XOR on MST

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	1024 megabytes

You are given a connected undirected graph G with N vertices and M edges. The i-th edge connects vertices U_i and V_i, has weight W_i, and has color C_i. The graph has no self-loops, but may have multiple edges.

You are also given K pairs of vertices $(A_1, B_1), (A_2, B_2), \ldots, (A_K, B_K)$.

There are Q queries, which are processed in order. The j-th query adds an edge connecting vertices X_j and Y_j, with weight Z_j and color D_j. This edge is assigned index $M + j$.

For each $q = 0, 1, \ldots, Q$, solve the following problem for the graph after processing the first q queries. Here, $q = 0$ refers to the initial graph.

- Let T be a minimum spanning tree of G. If there are multiple minimum spanning trees, give priority to edges with smaller indices among edges of equal weight. Equivalently, choose the minimum spanning tree whose sequence of edge indices, sorted in increasing order, is lexicographically smallest.

  For each $i = 1, 2, \ldots, K$, let P_i be the bitwise XOR of the colors of all edges on the path from A_i to B_i in T. Find the maximum, over all nonempty subsets S of $\{1, 2, \ldots, K\}$, of the bitwise XOR of P_i over all $i \in S$.

The weights Z_j and colors D_j of the edges added by the queries are encrypted. To decrypt the j-th query, you need the answer for $q = j - 1$. See the Input section for details.

Input

The first line contains three integers N, M, and K ($2 \le N \le 2 \times 10^5$, $N - 1 \le M \le 2 \times 10^5$, $1 \le K \le 2 \times 10^5$).

The next M lines describe the initial edges. The i-th of these lines contains four integers U_i, V_i, W_i, and C_i, in this order ($1 \le U_i < V_i \le N$, $0 \le W_i, C_i < 2^{30}$). They specify an edge connecting vertices U_i and V_i, with weight W_i and color C_i. The initial graph is guaranteed to be connected.

The next K lines describe the vertex pairs. The i-th of these lines contains two integers A_i and B_i ($1 \le A_i < B_i \le N$).

The next line contains an integer Q ($1 \le Q \le 10^5$), the number of queries.

The next Q lines describe the queries in order. The j-th of these lines contains four integers X_j, Y_j, Z'_j, and D'_j, in this order ($1 \le X_j < Y_j \le N$, $0 \le Z'_j, D'_j < 2^{30}$).

The values Z'_j and D'_j are the encrypted weight and color of the edge added by the j-th query. Let ans_{j-1} be the answer for $q = j - 1$. Recover the actual weight Z_j and color D_j as follows:

$$a = (\mathrm{ans}_{j-1} \times 987654321 + 123456789) \bmod 2^{30},$$
$$Z_j = Z'_j \,\mathrm{XOR}\, a,$$
$$D_j = D'_j \,\mathrm{XOR}\, a.$$

The decoded values satisfy $0 \le Z_j, D_j < 2^{30}$.

All input values are integers.

Output

Print $Q + 1$ lines. On the j-th line ($1 \le j \le Q + 1$), print the answer for $q = j - 1$.

Examples

standard input	standard output
4 3 2 1 2 5 3 2 3 2 6 3 4 3 5 1 2 3 4 2 1 3 680673594 680673596 1 2 766761096 766761102	6 5 5
7 8 2 3 4 61 32 1 5 27 30 5 6 5 45 3 4 38 18 4 7 62 52 2 3 20 54 3 5 42 8 3 6 50 58 1 2 5 6 5 6 7 544486451 544486435 2 7 544486448 544486444 1 6 326915773 326915738 2 6 326915750 326915753 4 6 326915738 326915732	45 45 60 60 60 60
10 9 6 5 8 902401407 293322456 4 7 930849631 5730471 6 7 872567750 367631210 2 3 897829007 138510651 8 9 114856688 92032694 4 9 709861086 656791295 3 10 87259312 773491721 2 4 749455618 526006038 1 10 887106268 858731897 3 9 2 5 1 7 4 9 4 8 2 10 8 2 7 140356964 161504920 4 5 781804833 195879806 1 3 446276419 751424261 2 4 472867995 551228575 4 5 378561056 485433981 1 2 184349035 537730583 1 3 823227708 184612008 6 7 816143859 47682480	1063334609 1073686796 1073686796 1037058542 1028634601 1037377830 1070698744 1061831886 1061831886

Note

For nonnegative integers A and B, their bitwise XOR, denoted by $A \operatorname{XOR} B$, is defined as follows. For each $k \geq 0$, the bit corresponding to 2^k in the binary representation of $A \operatorname{XOR} B$ is 1 if exactly one of A and B has a 1 in that position, and is 0 otherwise.

For example, $3 \operatorname{XOR} 5 = 6$, since $011 \operatorname{XOR} 101 = 110$ in binary. More generally, the bitwise XOR of k nonnegative integers $p_1, p_2, \ldots, p_k$ is obtained by successively applying this operation:

$$(\cdots((p_1 \operatorname{XOR} p_2) \operatorname{XOR} p_3) \operatorname{XOR} \cdots \operatorname{XOR} p_k).$$

It can be shown that the result does not depend on the order of $p_1, p_2, \ldots, p_k$.

In Sample 1, consider $q = 0$.

The graph has the following three edges, which together form the minimum spanning tree T.

- Edge 1 connects vertices 1 and 2, with weight 5 and color 3.
- Edge 2 connects vertices 2 and 3, with weight 2 and color 6.
- Edge 3 connects vertices 3 and 4, with weight 3 and color 5.

The path between vertices 1 and 2 consists only of edge 1, so $P_1 = 3$. The path between vertices 3 and 4 consists only of edge 3, so $P_2 = 5$.

- For $S = \{1\}$, the XOR is $P_1 = 3$.
- For $S = \{2\}$, the XOR is $P_2 = 5$.
- For $S = \{1, 2\}$, the XOR is $P_1 \operatorname{XOR} P_2 = 6$.

The maximum is 6, so the answer is 6.

Next, consider $q = 1$.

Decrypting the first query using the answer for $q = 0$ gives $Z_1 = 1$ and $D_1 = 7$. Thus, edge 4, connecting vertices 1 and 3 with weight 1 and color 7, is added.

The minimum spanning tree T now consists of edges 2, 3, and 4. The path between vertices 1 and 2 contains edges 2 and 4, so $P_1 = 6 \operatorname{XOR} 7 = 1$. The path between vertices 3 and 4 consists only of edge 3, so $P_2 = 5$.

- For $S = \{1\}$, the XOR is $P_1 = 1$.
- For $S = \{2\}$, the XOR is $P_2 = 5$.
- For $S = \{1, 2\}$, the XOR is $P_1 \operatorname{XOR} P_2 = 4$.

The maximum is 5, so the answer is 5.

Finally, consider $q = 2$.

Decrypting the second query using the answer for $q = 1$ gives $Z_2 = 2$ and $D_2 = 4$. Thus, edge 5, connecting vertices 1 and 2 with weight 2 and color 4, is added.

There are now two minimum spanning trees: one consisting of edges 2, 3, and 4, and the other consisting of edges 3, 4, and 5. The rule giving priority to smaller edge indices selects the former as T.

Since T is unchanged from $q = 1$, the values of P_1 and P_2 are also unchanged. The maximum remains 5, so the answer is 5.

Problem G. Source to Sink

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

You are given a simple directed graph G with N vertices and M edges. The vertices are numbered from 1 to N, and the i-th edge is directed from vertex u_i to vertex v_i. The graph has no self-loops or multiple edges in the same direction, but it may contain edges in both directions between two vertices.

You may perform the following operation any number of times:

- Choose a vertex v of G with indegree 0, and reverse the direction of every edge incident to v.

Find the maximum number of operations you can perform. If you can continue performing operations indefinitely, report this instead.

You are given T test cases. Solve each test case.

Input

The first line contains an integer T ($1 \le T \le 5 \times 10^4$), the number of test cases.

The first line of each test case contains two integers N and M ($2 \le N \le 10^5$, $1 \le M \le \min(N(N-1), 2 \times 10^5)$), the numbers of vertices and edges.

The next M lines describe the edges. The i-th of these lines contains two integers u_i and v_i ($1 \le u_i, v_i \le N$, $u_i \ne v_i$), representing a directed edge from vertex u_i to vertex v_i. For any distinct indices i and j, $(u_i, v_i) \ne (u_j, v_j)$.

The sum of N over all test cases does not exceed 10^5, and the sum of M over all test cases does not exceed 2×10^5.

All input values are integers.

Output

Print T lines. On the t-th line, print the answer to the t-th test case.

For each test case, print the maximum number of operations you can perform, or -1 if you can continue performing operations indefinitely.

Example

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

Note

In the first test case, you can perform two operations as follows.

- First, operate on vertex 2. This reverses the directions of the second and third edges.
- Then, operate on vertex 5. This reverses the directions of the third and sixth edges.

You cannot perform three or more operations, so the answer is 2.

In the second test case, you can continue performing operations indefinitely by alternating between vertices 1 and 2.

In the third test case, you can operate on vertex 3 any number of times.

In the fourth test case, you cannot perform any operations.

Problem H. Please Insert Coin

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

There is a grid with H rows and W columns. Let (i, j) denote the cell in the i-th row from the top and the j-th column from the left. Each cell (i, j) contains an integer $A_{i,j}$ indicating the time at which a block is placed in that cell.

Time advances from 1 to $K = \max_{1 \le i \le H,\ 1 \le j \le W} A_{i,j}$. The array A satisfies the following properties.

- For every integer k with $1 \le k \le K$, at least one cell has value k.
- The cells with value k form a connected set under horizontal and vertical adjacency.
- Blocks are stacked from bottom to top in each column: $A_{i+1,j} \le A_{i,j}$ for all $1 \le i \le H - 1$ and $1 \le j \le W$.

Let h_j be the number of blocks currently stacked in column j. Initially, $h_j = 0$ for every column.

The player *soryuusi* is initially in column s. At each time $k = 1, 2, \ldots, K$, the following two steps take place in order.

1. **Movement**: *soryuusi* may move to a horizontally adjacent column within the grid, one column at a time, any number of times. He may also stay in his current column. Each move from a column x to an adjacent column y consumes $|h_x - h_y| + 1$ stamina, using the block heights at the time of the move.
2. **Block placement**: A block is placed in every cell (i, j) with $A_{i,j} = k$. For each column j, h_j increases by the number of blocks placed in that column. If at least one new block is placed in the column *soryuusi* currently occupies, he is crushed, and the game ends.

Determine whether *soryuusi* can survive through the completion of block placement at time K. If he can, find the minimum total stamina he must consume. Otherwise, output -1.

You are given T test cases. Solve each test case.

Input

The first line contains an integer T ($1 \le T \le 10^5$), the number of test cases.

For each test case, the first line contains three integers H, W, and s, in this order ($1 \le H, W$, $H \times W \le 5 \times 10^5$, $1 \le s \le W$).

The next H lines describe the array A. The i-th of these lines contains W integers $A_{i,1}, A_{i,2}, \ldots, A_{i,W}$, in this order ($1 \le A_{i,j} \le H \times W$).

Let K be the maximum value in A. For every integer k with $1 \le k \le K$, at least one cell has value k, and the cells with value k form a connected set under horizontal and vertical adjacency. Also, $A_{i+1,j} \le A_{i,j}$ holds for all $1 \le i \le H - 1$ and $1 \le j \le W$.

The sum of $H \times W$ over all test cases is at most 5×10^5.

All input values are integers.

Output

Output T lines. On the t-th line, output the minimum total stamina required to survive in the t-th test case, or -1 if survival is impossible.

Example

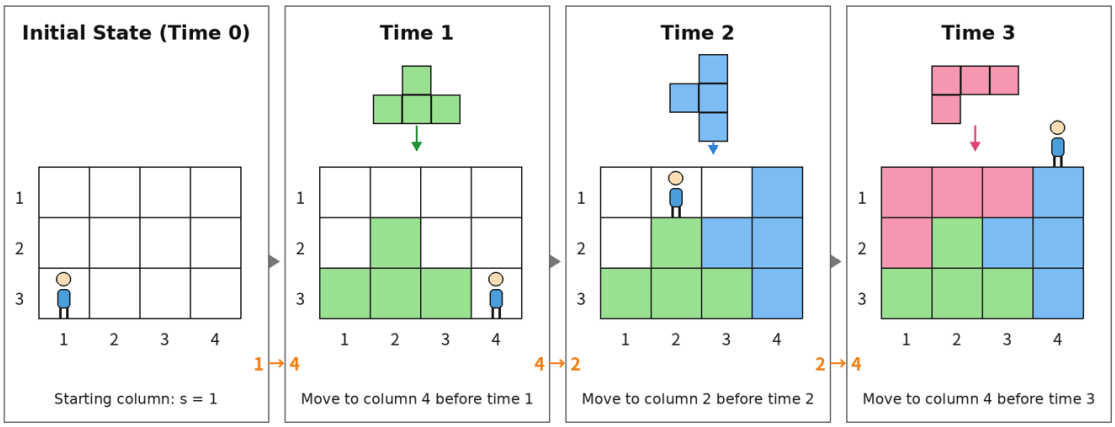
standard input	standard output
3 3 4 1 3 3 3 2 3 1 2 2 1 1 1 2 2 3 2 2 2 3 1 2 2 4 5 5 16 17 18 19 20 11 12 13 14 15 6 7 8 9 10 1 2 3 4 5	10 -1 8

Note

In the first test case, *soryuusi* can consume a total of 10 stamina by acting as follows.

- He moves to column 4, consuming $(|0-0|+1)+(|0-0|+1)+(|0-0|+1)=3$ stamina.
- The blocks for time 1 are placed.
- He moves to column 2, consuming $(|0-1|+1)+(|1-2|+1)=4$ stamina.
- The blocks for time 2 are placed.
- He moves to column 4, consuming $(|2-2|+1)+(|2-3|+1)=3$ stamina.
- The blocks for time 3 are placed.

It is impossible to survive while consuming at most 9 stamina, so the minimum total stamina required is 10.



In the second test case, regardless of how *soryuusi* moves, he cannot avoid having blocks placed in his current column at time 2. Therefore, the output for this test case is -1.

Problem I. tatesoto decomposition

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

You are given a string X consisting of lowercase English letters. Find the number of ordered tuples (A, E, O, S, T) of nonempty strings satisfying

$$X = T + A + T + E + S + O + T + O$$

modulo 998244353.

Here, + denotes string concatenation.

You are given Q test cases. Solve each of them.

Input

The first line contains an integer Q ($1 \le Q \le 5 \times 10^4$), the number of test cases.

Each of the next Q lines contains a string X for one test case.

Each string X consists of lowercase English letters and satisfies $8 \le |X| \le 4 \times 10^5$. The sum of $|X|$ over all test cases does not exceed 4×10^5.

Output

Print Q lines. On the i-th line, print the number of valid ordered tuples for the i-th test case, modulo 998244353.

Example

standard input	standard output
5 tatesoto jastaway ttaatteessoottoo wwwwwwwwww abcabcabcabcabcabcabc	1 0 3 7 72

Note

In the first test case, the tuple $(A, E, O, S, T) = (\texttt{a}, \texttt{e}, \texttt{o}, \texttt{s}, \texttt{t})$ satisfies the condition.

In the second test case, no ordered tuple (A, E, O, S, T) of nonempty strings satisfies the condition.

In the third test case, exactly the following three tuples satisfy the condition:

- $(A, E, O, S, T) = (\texttt{aa}, \texttt{e}, \texttt{oo}, \texttt{ess}, \texttt{tt})$;
- $(A, E, O, S, T) = (\texttt{aa}, \texttt{ee}, \texttt{oo}, \texttt{ss}, \texttt{tt})$;
- $(A, E, O, S, T) = (\texttt{aa}, \texttt{ees}, \texttt{oo}, \texttt{s}, \texttt{tt})$.

Remember to output the count modulo 998244353.

Problem J. Hugest Number

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

You are given two distinct positive integers D_1 and D_2.

Consider all positive integers whose decimal representations have exactly D_1 digits or exactly D_2 digits. Arrange all of these integers in a row in some order, using each integer exactly once. Let X be the integer obtained by concatenating their standard decimal representations in this order, without any separators.

Find the maximum possible value of X. Since the answer must be very large, find it modulo 998244353.

You are given T test cases; solve each of them.

Input

The first line contains an integer T ($1 \le T \le 1000$), the number of test cases.

Each of the next T lines describes one test case and contains two integers D_1 and D_2 ($1 \le D_1 < D_2 \le 10^{18}$).

Output

For each test case, output the maximum possible value of X modulo 998244353 on a separate line.

Example

standard input	standard output
4 1 2 1 3 2 3 333333333259845632 999999999779536896	682115319 588022100 745765835 0

Note

In the first test case, we arrange all one-digit positive integers $1, 2, \ldots, 9$ and all two-digit positive integers $10, 11, \ldots, 99$.

One optimal arrangement is obtained by first arranging the two-digit integers in the order $99, 98, \ldots, 10$ and then, for each $d = 1, 2, \ldots, 9$, inserting d immediately before $11 \times d$. The resulting integer is $X = 9999897\ldots1211110$, and its remainder modulo 998244353 is 682115319.

In the fourth test case, the maximum possible value of X is divisible by 998244353, so the answer is 0.

Problem K. Dyads of Hanoi

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

The Tower of Hanoi is a puzzle consisting of three pegs and N disks of distinct sizes. The pegs are numbered 1, 2, and 3. The disks are numbered $1, 2, \ldots, N$ in increasing order of size.

At all times, every disk must be on one of the pegs, and the disks on each peg must be stacked in decreasing order of size from bottom to top. In one move, you may take the top disk from a peg and move it to another peg that is either empty or has a larger disk on top.

A pair (X, Y) of Tower of Hanoi states is called a *good pair* if there are exactly two sequences of moves that transform state X into state Y using the minimum possible number of moves. Two sequences, each consisting of L moves, are considered different if, for some i with $1 \le i \le L$, their i-th moves differ in at least one of the following: the disk moved, the source peg, or the destination peg. In particular, when $X = Y$, the only sequence using the minimum number of moves is the empty sequence, so (X, Y) is not a good pair.

A Tower of Hanoi state is represented by a string of length N consisting of the characters `1`, `2`, and `3`. If the i-th character is `k`, disk i is on peg k.

You are given two strings A and B of length N consisting of the characters `1`, `2`, `3`, and `?`. Each occurrence of `?` in either string must independently be replaced with `1`, `2`, or `3`. If there are K occurrences of `?` in total, there are 3^K ways to make these replacements.

Let X and Y be the states represented by A and B, respectively, after the replacements. Find the number of ways to make the replacements such that (X, Y) is a good pair, modulo 998244353.

You are given T test cases. Solve each test case.

Input

The first line contains an integer T ($1 \le T \le 10^5$), the number of test cases.

Each test case consists of three lines. The first line contains an integer N ($1 \le N \le 2 \times 10^5$), the number of disks. The second line contains the string A, and the third line contains the string B. Both strings have length N and consist only of the characters `1`, `2`, `3`, and `?`.

The sum of N over all test cases is at most 2×10^5. All numerical input values are integers.

Output

Print T lines. On the t-th line, print the number of valid replacements for the t-th test case, modulo 998244353.

Example

standard input	standard output
5 2 12 2? 3 3?3 ?3? 3 123 231 20 2?3?1?33?21?1???1?2? ?11?2?3?2?3??22?12?? 30 ?????????????????????????????? ??????????????????????????????	1 4 0 1812 915053767

Note

Consider the first test case.

If the states after replacement are $(X, Y) = (12, 21)$, the minimum number of moves required to transform X into Y is 3. There are exactly two such sequences.

In the first sequence, move disk 1 from peg 1 to peg 3, then move disk 2 from peg 2 to peg 1, and finally move disk 1 from peg 3 to peg 2.

In the second sequence, move disk 2 from peg 2 to peg 3, then move disk 1 from peg 1 to peg 2, and finally move disk 2 from peg 3 to peg 1.

Thus, (X, Y) is a good pair in this case.

If the states after replacement are $(X, Y) = (12, 22)$, the minimum number of moves required to transform X into Y is 1. There is exactly one such sequence: move disk 1 from peg 1 to peg 2.

Thus, (X, Y) is not a good pair in this case.

If the states after replacement are $(X, Y) = (12, 23)$, the minimum number of moves required to transform X into Y is 2. There is exactly one such sequence: move disk 2 from peg 2 to peg 3, then move disk 1 from peg 1 to peg 2.

Thus, (X, Y) is not a good pair in this case.

Problem L. Consecutive Pairs

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

This is an interactive problem.

tatesoto has a hidden permutation $P = (P_1, P_2, \ldots, P_N)$ of $(1, 2, \ldots, N)$. He has told you that 1 appears before N in P.

You may ask *tatesoto* queries of the following type:

- choose integers l and r with $1 \le l \le r \le N$, and ask for the number of integers x such that $1 \le x < N$ and both x and $x + 1$ appear in $(P_l, P_{l+1}, \ldots, P_r)$.

Determine the hidden permutation P using at most $12N$ queries.

tatesoto always answers truthfully. The judge is not adaptive: the permutation P is fixed before the interaction begins and does not change during the interaction.

Interaction Protocol

First, read an integer N ($2 \le N \le 2000$) from standard input in the following format:

N

The hidden permutation satisfies $1 \le P_i \le N$ for every $1 \le i \le N$, and $P_i \ne P_j$ whenever $i \ne j$. If $P_i = 1$ and $P_j = N$, then $i < j$. The permutation itself is not given as input.

After reading N, you may ask queries repeatedly to determine P.

To ask a query, output a single line in the following format, where l and r are integers satisfying $1 \le l \le r \le N$:

`?` l r

After each query, read the response from standard input in the following format:

k

Here, k is the number of integers x such that $1 \le x < N$ and both x and $x+1$ appear in $(P_l, P_{l+1}, \ldots, P_r)$.

If the queried indices do not satisfy $1 \le l \le r \le N$, or if the number of queries exceeds $12N$, the judge returns $k = -1$. In this case, your submission has already been judged incorrect, and your program must terminate immediately.

Once you have determined P, output a single line in the following format:

`!` P_1 P_2 $\ldots$ P_N

Then terminate your program immediately.

Values on each output line must be separated by spaces. All numerical values given as input are integers.

After every output, print a newline and flush standard output. Otherwise, your submission may receive a Time Limit Exceeded or Wrong Answer verdict.

Terminate your program immediately after outputting the answer or receiving -1. Otherwise, the behavior of the judge is undefined.

Extra newlines are treated as incorrectly formatted output.

Example

standard input	standard output
4 1 0 0 2 1	 ? 1 2 ? 2 3 ? 3 4 ? 1 3 ? 2 4 ! 2 3 1 4

Note

In the example interaction, you first receive $N = 4$. The hidden permutation is $P = (2, 3, 1, 4)$.

The first query is $l = 1$, $r = 2$. Since $(P_1, P_2) = (2, 3)$, only $x = 2$ satisfies the condition, so the judge returns 1.

The second query is $l = 2$, $r = 3$. Since $(P_2, P_3) = (3, 1)$, no x satisfies the condition, so the judge returns 0.

The third query is $l = 3$, $r = 4$. Since $(P_3, P_4) = (1, 4)$, no x satisfies the condition, so the judge returns 0.

The fourth query is $l = 1$, $r = 3$. Since $(P_1, P_2, P_3) = (2, 3, 1)$, both $x = 1$ and $x = 2$ satisfy the condition, so the judge returns 2.

The fifth query is $l = 2$, $r = 4$. Since $(P_2, P_3, P_4) = (3, 1, 4)$, only $x = 3$ satisfies the condition, so the judge returns 1.

You conclude that $P = (2, 3, 1, 4)$, output the answer, and terminate your program. The permutation $(3, 2, 4, 1)$ is also consistent with all the responses, but it is ruled out because $4 = N$ appears before 1.

Problem M. I hate prime numbers

Input file:	`standard input`
Output file:	`standard output`
Time limit:	10 seconds
Memory limit:	1024 megabytes

soryuusi dislikes prime numbers, and he came up with the following problem.

You are given positive integers N and M. A pair (A, B) of sequences of positive integers, $A = (A_1, A_2, \ldots, A_N)$ and $B = (B_1, B_2, \ldots, B_N)$, is called a *good pair of sequences* if all of the following conditions hold:

- For every integer i with $1 \le i \le N$, we have $1 \le A_i \le N$.
- For every integer i with $1 \le i \le N$, we have $B_i \le B_{A_i}$.
- For every pair of integers (a, b), the number of indices i with $1 \le i \le N$ such that $(A_i, B_i) = (a, b)$ is not a prime number.

For each $m = 1, 2, \ldots, M$, find the number of good pairs of sequences (A, B) for which the maximum element of B is m, modulo 998244353.

Input

The only line contains two integers N and M, in this order, satisfying $1 \le N \le 5 \times 10^4$ and $1 \le M \le 10^5$.

Output

Print M space-separated integers on a single line. The m-th integer must be the answer for m, modulo 998244353, for each $m = 1, 2, \ldots, M$.

Examples

standard input	standard output
2 3	2 6 10
4 7	28 364 1564 4204 8860 16108 26524
45678 4	248811653 85639511 671022534 843777830

Note

In the first sample, there are exactly two good pairs of sequences (A, B) for which the maximum element of B is 1:

- $A = (1, 2),\ B = (1, 1)$.
- $A = (2, 1),\ B = (1, 1)$.

There are exactly six good pairs of sequences (A, B) for which the maximum element of B is 2:

- $A = (1, 1),\ B = (2, 1)$.
- $A = (1, 2),\ B = (1, 2)$.
- $A = (1, 2),\ B = (2, 1)$.
- $A = (1, 2),\ B = (2, 2)$.
- $A = (2, 1),\ B = (2, 2)$.
- $A = (2, 2),\ B = (1, 2)$.

Problem N. Delete Sad Faces (Hard)

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

tatesoto came up with the following problem.

> Consider strings consisting of the characters `:`, `(`, and `)`. Two adjacent characters form an emoticon if they match one of the following patterns:
>
> — `:)` and `(:` are happy faces.
> — `:(` and `):` are sad faces.
>
> The number of happy faces (sad faces) in a string T is the number of integers i satisfying $1 \le i < |T|$ such that T_iT_{i+1} is a happy face (sad face).
>
> Emoticons are counted separately even if they share a character. For example, the string `(:):` contains the emoticons `(:`, `:)`, and `):`, so it contains two happy faces and one sad face.
>
> You are given a string T. Repeat the following operation until T contains no sad faces:
>
> — choose one sad face in the current string T and delete its two characters, joining the remaining characters without changing their order.
>
> Find the maximum possible number of happy faces in the final string T.

soryuusi found this problem too easy, so he came up with the following problem based on it.

> You are given a string S consisting of the characters `:`, `(`, and `)`, and Q queries. The i-th query gives two integers L_i and R_i satisfying $1 \le L_i \le R_i \le |S|$. Find the answer to *tatesoto*'s problem for the substring $T = S_{L_i}S_{L_i+1} \cdots S_{R_i}$.

Solve *soryuusi*'s problem.

Input

The first line contains a string S consisting of the characters `:`, `(`, and `)`, where $1 \le |S| \le 2 \times 10^5$.

The second line contains an integer Q ($1 \le Q \le 2 \times 10^5$), the number of queries.

Each of the next Q lines contains two integers L_i and R_i ($1 \le L_i \le R_i \le |S|$), describing the i-th query.

Output

Print Q lines. On the i-th line, print the answer to the i-th query.

Examples

standard input	standard output
(:):(4 2 5 1 3 3 5 1 5	1 2 0 2
:::::::: 1 1 8	0
)()):)()))::::(:::):(((::)((): 10 13 22 20 22 16 26 2 21 1 28 7 28 14 18 23 24 3 7 10 16	1 0 3 2 4 4 0 1 0 0

Note

In the first sample, the queries are processed as follows.

For the first query, $T =$ `:):(`. Deleting the sad face `:(` at positions 3 and 4 leaves $T =$ `:)`. This string contains one happy face, and it can be proved that this is the maximum possible number.

For the second query, $T =$ `(:)`. It already contains no sad faces and contains two happy faces.

For the third query, $T =$ `):(`. Deleting the sad face `):` at positions 1 and 2 leaves $T =$ `(`. This string contains zero happy faces, and it can be proved that this is the maximum possible number.

For the fourth query, $T =$ `(:):(`. Deleting the sad face `:(` at positions 4 and 5 leaves $T =$ `(:)`. This string contains two happy faces, and it can be proved that this is the maximum possible number.