

Hong Kong Joint Collegiate Programming Contest 2026

2026 年香港联合大学生程序设计竞赛

30 AUGUST 2026 · 2026 年 8 月 30 日



No.	Problem Name	题目名称
A	Assigning Distinct Integers	互异整数分配
B	Box Projections	长方体投影
C	Constructing Box Projections	构造长方体投影
D	Dynamic Connectivity Across Graphs	多图动态连通性
E	Encoded Shortest Paths	编码最短路
F	First and Second Minimum	最小值与次小值
G	Game of Passing Cards	传牌游戏
H	Human-Readable Number	易读数字
I	Interprocess Sum and Scatter	进程间求和与分发
J	Jumbled Block Hierarchy	乱序区间层级
K	Keep the Fairness	保持公平
L	Let's Go Yum Cha	一起去饮茶

Please refer to the time and memory limits displayed on the online judge.

请参考在线评测系统上显示的时间和内存限制。



The problems in this contest were prepared by the eps.ac problem-setting team.

本场比赛题目由 eps.ac 出题组命制。



Follow eps.ac and scan the QR Code AFTER THE CONTEST to learn more.

欢迎关注 eps.ac，赛后扫描二维码了解更多。

Problem A. Assigning Distinct Integers

Input file: standard input
Output file: standard output

You are given n positive integers $a_1, a_2, \dots, a_n$.

Construct a sequence of n pairwise distinct positive integers $b_1, b_2, \dots, b_n$ such that $b_i \geq a_i$ for every $1 \leq i \leq n$. First minimize $\sum_{i=1}^n b_i$. Among all sequences with the minimum sum, output the lexicographically smallest one.

A sequence $(x_1, x_2, \dots, x_n)$ is lexicographically smaller than $(y_1, y_2, \dots, y_n)$ if there is an index p such that $x_i = y_i$ for every $1 \leq i < p$, and $x_p < y_p$.

Input

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

For each test case:

- The first line contains an integer n ($1 \leq n \leq 3 \times 10^5$).
- The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq 10^9$).

It is guaranteed that the sum of n over all test cases does not exceed 3×10^5 .

Output

For each test case, print one line containing n integers $b_1, b_2, \dots, b_n$, in index order.

Example

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

Note

In the first test case, the minimum possible sum is $1 + 2 + 3 = 6$.

In the third test case, every optimal sequence uses the values 1, 2, 3. The lexicographically smallest valid order is (2, 1, 3).

Problem B. Box Projections

Input file: standard input
Output file: standard output

For a positive integer d , let $[d] = \{1, 2, \dots, d\}$. You are given three positive integers a, b, c . Consider a set $S \subseteq [a] \times [b] \times [c]$ of integer triples. It must satisfy all of the following conditions:

- for every $(x, y) \in [a] \times [b]$, at least one triple (x, y, z) belongs to S ;
- for every $(x, z) \in [a] \times [c]$, at least one triple (x, y, z) belongs to S ;
- for every $(y, z) \in [b] \times [c]$, at least one triple (x, y, z) belongs to S .

In other words, deleting any one coordinate from the triples in S must produce every possible pair of the other two coordinates. Determine the minimum possible value of $|S|$.

Input

The first line contains an integer T ($1 \leq T \leq 100$), the number of test cases.
For each test case, one line contains three integers a, b, c ($1 \leq a \leq 100, 1 \leq b \leq 100, 1 \leq c \leq 100$).

Output

For each test case, print one line containing one integer: the minimum possible value of $|S|$.

Example

standard input	standard output
2	4
2 2 2	12
4 3 2	

Note

For $(a, b, c) = (2, 2, 2)$, one possible choice of S is:

$$S = \{(1, 1, 1), (1, 2, 2), (2, 1, 2), (2, 2, 1)\}$$

For $(a, b, c) = (4, 3, 2)$, one possible choice of S is:

$$S = \{(1, 1, 1), (1, 2, 1), (1, 3, 2), \\ (2, 1, 1), (2, 2, 2), (2, 3, 1), \\ (3, 1, 2), (3, 2, 2), (3, 3, 1), \\ (4, 1, 2), (4, 2, 1), (4, 3, 2)\}$$

Problem C. Constructing Box Projections

Input file: standard input
Output file: standard output

For a positive integer d , let $[d] = \{1, 2, \dots, d\}$. You are given three positive integers a, b, c . Construct a set $S \subseteq [a] \times [b] \times [c]$ of integer triples satisfying all of the following conditions:

- for every $(x, y) \in [a] \times [b]$, at least one triple (x, y, z) belongs to S ;
- for every $(x, z) \in [a] \times [c]$, at least one triple (x, y, z) belongs to S ;
- for every $(y, z) \in [b] \times [c]$, at least one triple (x, y, z) belongs to S .

In other words, deleting any one coordinate from the triples in S must produce every possible pair of the other two coordinates. The size $|S|$ must be as small as possible.

Input

The first line contains an integer T ($1 \leq T \leq 100$), the number of test cases.
For each test case, one line contains three integers a, b, c ($1 \leq a \leq 100, 1 \leq b \leq 100, 1 \leq c \leq 100$).

Output

For each test case, print several lines:

- The first line contains one integer m , the minimum possible value of $|S|$.
- Each of the next m lines contains three integers x, y, z , describing one triple in S .

The m triples must be pairwise distinct. Their order does not matter. If several minimum-size sets are valid, output any one of them.

Example

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

Note

The sample output is one possible answer. Other valid minimum-size sets are also accepted.

Problem D. Dynamic Connectivity Across Graphs

Input file: standard input
Output file: standard output

There are k undirected graphs $G_1, G_2, \dots, G_k$. They share the vertex set $\{1, 2, \dots, n\}$, and initially none of them has an edge.

Process q operations in order. An operation i, u, v adds an undirected edge between distinct vertices u and v in G_i . The same edge may be added more than once, and u and v may already be connected in G_i .

After every operation, determine the number of unordered pairs of distinct vertices $\{x, y\}$ such that x and y belong to the same connected component in every one of the k graphs.

Input

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

For each test case:

- The first line contains three integers n, k, q ($1 \leq n \leq 5 \times 10^5$, $1 \leq k \leq 5 \times 10^5$, $0 \leq q \leq 5 \times 10^5$), the number of vertices, graphs, and operations.
- Each of the next q lines contains three integers i, u, v ($1 \leq i \leq k$, $1 \leq u \leq n$, $1 \leq v \leq n$, $u \neq v$), describing one operation.

It is guaranteed that the sum of q over all test cases does not exceed 5×10^5 .

Output

For each test case, print q lines. The j -th line must contain one integer: the required number of unordered pairs immediately after the j -th operation. If $q = 0$, print no lines for that test case.

Example

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

Note

In the first test case, vertices 1 and 2 become connected in both graphs after the second operation. After the fourth operation, all three pairs of vertices are connected in both graphs.

The second test case has only one graph, so the answer is the number of vertex pairs within its connected components.

Problem E. Encoded Shortest Paths

Input file: standard input
Output file: standard output

You are given an undirected graph with n vertices numbered from 1 to n and m edges. Each edge has an integer weight w satisfying $0 \leq w < k$. Parallel edges are allowed, while self-loops are not allowed.

Consider a walk that starts at vertex 1 and traverses edges with weights $w_1, w_2, \dots, w_L$ in this order. A walk may visit vertices and edges more than once. If $L \geq 1$, the **total weight** of the walk is

$$w_1 \cdot k^{L-1} + w_2 \cdot k^{L-2} + \dots + w_{L-1} \cdot k + w_L$$

The empty walk has total weight 0.

For every vertex v , let $\text{dist}(v)$ be the minimum total weight among all walks from vertex 1 to vertex v . If no such walk exists, $\text{dist}(v)$ is undefined.

For every vertex v , determine $\text{dist}(v)$ modulo 998 244 353.

Input

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

For each test case:

The first line contains three integers n , m , and k ($1 \leq n \leq 2 \times 10^5$, $0 \leq m \leq 5 \times 10^5$, $1 \leq k \leq 10^9$), denoting the number of vertices, the number of edges, and the parameter used in the definition of total weight, respectively.

Each of the next m lines contains three integers u , v , and w ($1 \leq u, v \leq n$, $u \neq v$, $0 \leq w < k$), denoting an undirected edge between vertices u and v with weight w .

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5 , and the sum of m over all test cases does not exceed 5×10^5 .

Output

For each test case, print one line containing n integers. For each $1 \leq i \leq n$, consider the i -th integer. If $\text{dist}(i)$ is defined, it must be $\text{dist}(i)$ modulo 998 244 353; otherwise, it must be -1 .

Example

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

Note

In the sample, the empty walk gives $\text{dist}(1) = 0$. The one-edge walks give $\text{dist}(2) = 1$ and $\text{dist}(3) = 2$.

There are two two-edge walks from vertex 1 to vertex 4. Their total weights are $1 \cdot 3 + 0 = 3$ and $2 \cdot 3 + 0 = 6$, so $\text{dist}(4) = 3$. Vertex 5 is unreachable from vertex 1.

Problem F. First and Second Minimum

Input file: standard input
Output file: standard output

You are given a permutation $a_1, a_2, \dots, a_n$ of $1, 2, \dots, n$.

For $1 \leq x < y \leq n$, the subarray $[x, y]$ is called **special** if the values at its two endpoints are exactly the smallest and the second smallest values in the subarray. In other words, the unordered pair $\{a_x, a_y\}$ must consist of the two smallest values among $a_x, a_{x+1}, \dots, a_y$.

Every subarray of length 2 is special.

For each query interval $[l, r]$, among all special subarrays $[x, y]$ satisfying $l \leq x < y \leq r$, determine the maximum length $y - x + 1$.

Input

The first line contains two integers n and q ($2 \leq n \leq 3 \times 10^5$, $1 \leq q \leq 3 \times 10^5$), denoting the length of the permutation and the number of queries, respectively.

The second line contains n integers $a_1, a_2, \dots, a_n$, which form a permutation of $1, 2, \dots, n$.

Each of the next q lines contains two integers l and r ($1 \leq l < r \leq n$), denoting a query interval.

Output

Print q lines. The i -th line must contain one integer: the maximum length of a special subarray completely contained in the i -th query interval.

Example

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

Note

For the query $[1, 5]$, the whole subarray is special because its endpoint values are 1 and 2, the two smallest values in the subarray. Therefore, the answer is 5.

For the query $[1, 4]$, the whole subarray is also special because its endpoint values are 1 and 3, so the answer is 4. For the query $[1, 3]$, the answer is 3. For each remaining query, no special subarray of length greater than 2 is fully contained in the query interval.

Problem G. Game of Passing Cards

Input file: standard input
Output file: standard output

n people numbered $0, 1, \dots, n-1$ sit in a circle. The left neighbor of person i is person $(i-1) \bmod n$, and the right neighbor is person $(i+1) \bmod n$. When $n=2$, these two neighbors are the same person.

Each person initially holds one card. Every card is marked with one direction, either L or R.

Before any passes are performed, person 0 receives one additional card marked R. This initial reception is not counted as a pass. At this moment, person 0's original card is called the **old card**, and the additional card is called the **new card**.

At every moment, exactly one person holds two cards. For that person, the card held earlier is the old card, and the card received later is the new card.

One pass is performed as follows:

1. The person holding two cards examines the old card.
2. If the old card is marked L, it is passed to the left neighbor. If it is marked R, it is passed to the right neighbor.
3. The new card is kept. This counts as one pass performed by the current person.
4. For the recipient, the card already held becomes the old card, and the received card becomes the new card.

After exactly k passes, determine the final card configuration and the number of passes performed by each person.

Input

The first line contains two integers n and k ($2 \leq n \leq 18$, $1 \leq k \leq 10^{18}$), denoting the number of people and the number of passes, respectively.

The second line contains a string s of length n consisting only of characters L and R. For each $0 \leq i < n$, the character at position $i+1$ is the direction on the card initially held by person i .

Output

Print three lines.

The first line must contain an integer p and a character c , separated by a space. Here, p is the person holding two cards after all k passes, and c is the direction on that person's new card.

The second line must contain a string of length n . For each $0 \leq i < n$, consider the character at position $i+1$. If $i=p$, it is the direction on person i 's old card; otherwise, it is the direction on that person's only card.

The third line must contain n integers. For each $0 \leq i < n$, the integer at position $i+1$ is the number of passes performed by person i .

Examples

standard input	standard output
5 1 RRRRR	1 R RRRRR 1 0 0 0 0
5 10 RRRRR	0 R RRRRR 2 2 2 2 2
5 8 RRLRL	4 R RLRLR 1 2 2 2 1

Note

In the first sample, person 0 passes the original R card to person 1. Person 1 then holds two cards, and the received card is the new card.

In the second sample, every card is marked R. The person holding two cards moves one position to the right after every pass, so each person performs exactly two passes.

Problem H. Human-Readable Number

Input file: standard input

Output file: standard output

You are given the decimal representation of a positive integer n . Insert commas so that its digits are divided into groups of three from right to left. The leftmost group may contain one, two, or three digits, and no comma may appear before the first digit.

Input

The first line contains the decimal representation of a positive integer n . It consists of between 1 and 10^5 digits and has no leading zeros.

Output

Print one line containing the representation of n after inserting the commas.

Examples

standard input	standard output
1	1
999	999
1234	1,234
998244353	998,244,353
1000000007	1,000,000,007

Note

A number with at most three digits is unchanged. For example, 999 remains 999.

For 1234, the rightmost group is 234 and the remaining group is 1, giving 1,234.

Problem I. Interprocess Sum and Scatter

Input file: standard input
Output file: standard output

In distributed deep learning frameworks, **Reduce-Scatter** is a common collective communication primitive.

There are P processes numbered from 0 to $P - 1$. Process i holds an array A_i of length $P \cdot K$, indexed from 0 to $P \cdot K - 1$.

First, add the arrays element by element. For every index x , let S_x be the sum of $A_i[x]$ over all processes i .

Next, divide S from left to right into P consecutive blocks of length K . Process j receives block j , which consists of $S_{jK}, S_{jK+1}, \dots, S_{jK+K-1}$.

Given all arrays, determine the block received by every process.

Input

The first line contains two integers P and K ($1 \leq P, K \leq 50$), the number of processes and the length of each block.

Each of the next P lines contains $P \cdot K$ integers. For every $0 \leq i < P$, line $i + 1$ among these lines contains $A_i[0], A_i[1], \dots, A_i[P \cdot K - 1]$ ($|A_i[x]| \leq 10^9$), the array held by process i .

Output

Print P lines. For every $0 \leq j < P$, line $j + 1$ must contain the K integers received by process j , in their original order.

Examples

standard input	standard output
2 2 1 2 3 4 5 6 7 8	6 8 10 12
1 3 -1 1 2	-1 1 2

Note

In the first example, the element-wise sum is $[6, 8, 10, 12]$. Therefore, process 0 receives $[6, 8]$, and process 1 receives $[10, 12]$.

In the second example, there is only one process, so it receives the entire array.

Problem J. Jumbled Block Hierarchy

Input file: standard input
Output file: standard output

Consider all permutations $P_1, P_2, \dots, P_n$ of the integers from 1 to n .

You are given m inclusive intervals of positions, called blocks. The blocks are laminar: any two blocks are either identical, have no position in common, or one is entirely contained in the other. In particular, two blocks never partially overlap.

A permutation respects the blocks if, for every given block $[l, r]$, the set

$$\{P_l, P_{l+1}, \dots, P_r\}$$

is a set of consecutive integers. The order of these values inside the block is unrestricted. Equivalently, the block must satisfy

$$\max_{l \leq i \leq r} P_i - \min_{l \leq i \leq r} P_i = r - l.$$

Only the blocks given in the input impose restrictions. Repeated copies of the same block do not add any restriction.

An index i with $1 \leq i < n$ is a descent of P if $P_i > P_{i+1}$. For every d from 0 to $n - 1$, let C_d be the number of permutations that respect all the given blocks and have exactly d descents.

No permutation is given in the input. Determine $C_0, C_1, \dots, C_{n-1}$ by counting all permutations that satisfy the restrictions.

Input

The first line contains two integers n and m ($1 \leq n \leq 2 \cdot 10^5$, $0 \leq m \leq 10^6$), the length of the permutation and the number of blocks.

Each of the next m lines contains two integers l_i and r_i ($1 \leq l_i \leq r_i \leq n$), describing the inclusive interval of positions of block i .

The blocks are given in an arbitrary order. It is guaranteed that they form a laminar family. Duplicate blocks may appear.

Output

Print one line containing n integers $C_0, C_1, \dots, C_{n-1}$ in this order, each taken modulo 998 244 353.

Examples

standard input	standard output
5 3 1 5 1 4 1 3	1 6 10 6 1
3 0	1 4 1

Note

In the first example, there are 24 permutations that respect all three blocks. The numbers having exactly 0, 1, 2, 3, 4 descents are respectively

$$1, 6, 10, 6, 1.$$

In the second example, there are no blocks, so every permutation of 1, 2, 3 is allowed. Among the six permutations, one has no descents, four have one descent, and one has two descents.

Problem K. Keep the Fairness

Input file: standard input
Output file: standard output

There are n teams and m distinct souvenirs. The teams are numbered from 1 to n , and the souvenirs are numbered from 1 to m .

Each team ranks all souvenirs from most preferred to least preferred. The ranking of team i is a permutation

$$t_{i,1}, t_{i,2}, \dots, t_{i,m},$$

where $t_{i,1}$ is its most preferred souvenir.

Every souvenir has already been assigned to exactly one team. Let a_k be the team to which souvenir k is assigned. A team may receive any number of souvenirs, including none.

The assignment is called **prefix-fair** if the following condition holds:

- For every team i , every other team j , and every integer L with $1 \leq L \leq m$, among the first L souvenirs in team i 's ranking, the number assigned to team i is at least the number assigned to team j .

Both numbers in this comparison are counted using team i 's ranking. Equality is allowed.

Determine whether the given assignment is prefix-fair.

Input

The first line contains two integers n and m ($1 \leq n, m \leq 2000$), the number of teams and the number of souvenirs.

The next n lines describe the rankings. The i -th of these lines contains m pairwise distinct integers $t_{i,1}, t_{i,2}, \dots, t_{i,m}$ ($1 \leq t_{i,k} \leq m$), from team i 's most preferred souvenir to its least preferred souvenir. Thus, each line is a permutation of $1, 2, \dots, m$.

The last line contains m integers $a_1, a_2, \dots, a_m$ ($1 \leq a_k \leq n$), where a_k is the team to which souvenir k is assigned.

Output

Print one line containing YES if the assignment is prefix-fair. Otherwise, print NO.

The answer is case-sensitive.

Examples

standard input	standard output
3 6 1 2 3 4 5 6 2 1 3 5 4 6 3 1 2 6 4 5 1 2 3 1 2 3	YES
2 4 1 2 4 3 2 1 4 3 1 2 1 2	NO

Note

In the first example, consider team 2. The owners of the souvenirs in its ranking are

$$2, 1, 3, 2, 1, 3.$$

In every prefix, team 2 occurs at least as many times as each of team 1 and team 3. The same property holds for teams 1 and 3, so the assignment is prefix-fair.

In the second example, the first three souvenirs in team 1's ranking are 1, 2, 4. Team 1 receives only souvenir 1, while team 2 receives souvenirs 2 and 4. Therefore, this prefix contains one souvenir assigned to team 1 and two assigned to team 2, so the assignment is not prefix-fair. Notice that both teams still receive two souvenirs in total; equal total numbers alone are not sufficient.

Problem L. Let's Go Yum Cha

Input file: standard input
Output file: standard output

At a Hong Kong tea house, the bill for a table consists of the tea charge, the dim sum charges, and possibly a 10% service charge, commonly called plus one.

Each kind of tea has a price per person. Everyone at the same table chooses the same kind of tea, and the tea charge is the number of people multiplied by the price of that tea per person.

Every dim sum dish belongs to exactly one of five price categories. All dishes in the same category have the same price.

Token	Category
SMALL	Small
MEDIUM	Medium
LARGE	Large
SPECIAL	Special
DELUXE	Deluxe

For each table, calculate the bill in the following order:

1. Charge the per-person price of the selected tea once for each of the n people at the table.
2. For each order record, find the price category of the named dish, then charge that category's price once for each serving in the record. The same dish may appear in more than one order record, and every record must be counted.
3. Add all tea and dim sum charges to obtain the subtotal.
4. If $w = 1$, the service charge is waived and the subtotal is the final bill. Otherwise, add a 10% service charge to the entire subtotal. Round the resulting total to the nearest integer Hong Kong dollar; if it is exactly halfway between two integers, round upward.

Given the menu and the orders of several tables, determine the final bill for every table.

Input

The first line contains an integer T ($1 \leq T \leq 1000$), the number of kinds of tea.

Each of the next T lines contains a string t_i and an integer a_i ($1 \leq a_i \leq 1000$), denoting the name and the price per person of tea i .

The next line contains five integers b_{SMALL} , b_{MEDIUM} , b_{LARGE} , b_{SPECIAL} , and b_{DELUXE} ($1 \leq b_{\text{SMALL}}, b_{\text{MEDIUM}}, b_{\text{LARGE}}, b_{\text{SPECIAL}}, b_{\text{DELUXE}} \leq 1000$), the prices of the five categories in exactly this order. These prices are not necessarily increasing.

The next line contains an integer D ($1 \leq D \leq 1000$), the number of dim sum dishes.

Each of the next D lines contains two strings d_i and c_i , denoting the name and the category of dim sum dish i . c_i is one of SMALL, MEDIUM, LARGE, SPECIAL, and DELUXE.

The next line contains an integer m ($1 \leq m \leq 1000$), the number of tables.

For each table:

- The first line contains an integer n , a string t , an integer k , and an integer w ($1 \leq n \leq 1000$, $0 \leq k \leq 1000$, $w \in \{0, 1\}$). They denote the number of people, the selected tea's name, the number

of order records, and whether the service charge is waived, respectively. If $w = 1$, the service charge is waived; if $w = 0$, it is charged.

- Each of the next k lines contains a string d_j and an integer q_j ($1 \leq q_j \leq 1000$), denoting that order record j orders q_j servings of dish d_j .

It is guaranteed that the sum of k over all tables does not exceed 1000.

Every tea name and dish name consists of 1 to 20 uppercase English letters, lowercase English letters, or underscores. Names are case-sensitive. Tea names are pairwise distinct, and dish names are pairwise distinct. Every referenced tea and dish is present in the menu.

Output

Print m lines. The i -th line must contain one integer: the final bill of table i in Hong Kong dollars.

Example

standard input	standard output
2 Pu_Er 8 Jasmine 10 20 24 28 32 45 6 Har_Gow DELUXE Siu_Mai LARGE Char_Siu_Bao MEDIUM Spring_Roll SMALL Egg_Tart SPECIAL Custard_Bun DELUXE 3 3 Pu_Er 4 0 Har_Gow 2 Siu_Mai 1 Har_Gow 1 Egg_Tart 2 2 Jasmine 2 1 Char_Siu_Bao 2 Spring_Roll 3 1 Jasmine 3 0 Custard_Bun 1 Spring_Roll 1 Spring_Roll 1	276 128 105

Note

For the first table, the tea charge is $3 \cdot 8 = 24$. Its dim sum charges total $3 \cdot 45 + 1 \cdot 28 + 2 \cdot 32 = 227$, so the subtotal is 251. The service charge is not waived, and the final bill is 276 after rounding.

For the second table, the subtotal is $2 \cdot 10 + 2 \cdot 24 + 3 \cdot 20 = 128$. The service charge is waived, so the final bill is 128.

For the third table, the subtotal is $1 \cdot 10 + 1 \cdot 45 + 2 \cdot 20 = 95$. The service charge is not waived. The total is exactly halfway between 104 and 105, so it is rounded upward to 105.