The 3rd Universal Cup

Stage 8: Cangqian

August 31 - September 1, 2024

This problem set should contain 13 problems (A to M) on 23 numbered pages.

Based on

International Collegiate Programming Contest (ICPC)

Hosted by

Problem A. Bingo

Time limit: 2 seconds
Memory limit: 1024 megabytes

Do you remember the hand-clapping game in elementary school? Here is a harder version.

You're given two positive integers n and m. Find the minimal integer x where $x > n$ and x is a **good** number. A good number x satisfies $x \equiv 0 \pmod{m}$ or x contains m as a substring in decimal representation.

For example, when $m = 3$ and $n = 7$, $x = 9$ is the answer since $9 \equiv 0 \pmod{3}$. When $m = 3$ and $n = 12$, $x = 13$ is the answer since 13 contains 3 as a substring.

Input

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

For each test case, one line contains two integers n and m ($1 \le n < 10^{10^6}, 1 \le m \le 10^9$). n does not contain leading zeros.

It is guaranteed that $\sum \lceil \log_{10}(n) \rceil \le 3 \times 10^6$

Output

For each test case, output one single line containing one integer x, representing the answer.

Example

standard input	standard output
6 7 3 12 3 9 10 249 51 1369 37 2 1	9 13 10 251 1370 3

Problem B. Simulated Universe

Time limit: 1 second
Memory limit: 1024 megabytes

"Star Rail" is a mixture of two traditional logic puzzle genres: "Star Battle" and "Rail Pool".

— Star Rail, CCBC 14

Sugar is playing his favorite game, Honkai: Star Rail. He is playing a game mode called Simulated Universe.

The gameplay revolves around collecting upgrades and artifacts, known as Blessings and Curios, to enhance the team's power and completing stages containing progressively more difficult enemies.

At the start of each playthrough, the player may choose a Path to follow, which will grant Path-specific buffs and an increased chance to obtain Blessings of the corresponding Path. During the playthrough, the player will complete different types of Domains (Combat, Occurrence, Transaction, Encounter, Elite, Respite) with the end goal of arriving at the final Boss Domain and defeating the boss.

Blessings will be accumulated through various means, which provide a variety of passive effects and buffs to the team. Blessings can be upgraded through Occurrences or within Respite Domains to enhance their effects. A currency known as Cosmic Fragments will also be obtained which can be used in certain Random Events or to upgrade Blessings.

Sugar is playing in low difficulty mode, so the scene has been greatly simplified. **We strongly recommend that you carefully read the following content, especially for players who are very familiar with the game. The game mode scene described in the prompt is quite different from the original game, so please discern carefully.** There are n domains in this playthrough, and the type of each domain is occurrence. Each domain provides **exactly one** of the following two rewards:

- Blessing: Sugar gains a Blessing, and his attack value increases by 1 immediately. Sugar can choose to upgrade this Blessing with 1 Cosmic Fragment if he has at least 1 Cosmic Fragment, and the upgraded Blessing can increase his attack value by 2 instead. A Blessing can only be upgraded no more than once.
- Curio: Sugar gains a Curio. There are two types of Curio, where **exactly one** of them can be chosen. The first type of Curio obtained in the i-th domain can upgrade no more than a_i un-upgraded Blessings obtained before the i-th domain, and the second type of Curio obtained in the i-th domain can provide b_i Cosmic Fragments immediately.

Note that the Cosmic Fragments **cannot** be used to upgrade the Blessings that Sugar has already obtained. It can only be used to upgrade the Blessings at the domain providing Blessing Rewards, and can only be used to upgrade the Blessing he has obtained in that domain. And the first type of Curio **can only** upgrade Blessings Sugar has already obtained.

Sugar starts this playthrough with 0 Blessings, 0 Cosmic Fragments and 0 attack value. Please help him calculate his maximum attack value after he passes through the n domains.

Input

The input contains multiple test cases, the first line contains an integer t ($1 \le t \le 10^3$), denoting the number of test cases.

For each test case, the first line contains an integer n ($1 \le n \le 8 \cdot 10^3$), denoting the number of domains.

The i-th of the following n lines starts with a single character t_i ($t_i \in \{\texttt{B}, \texttt{C}\}$), denoting the reward type

of the domain, which is a Blessing or a Curio. If $t_i = \texttt{C}$, then two integers a_i, b_i ($1 \le a_i, b_i \le n$) follow, denoting the Curio in the format described above.

It is guaranteed that the sum of n over all test cases does not exceed $8 \cdot 10^3$.

Output

For each test case, output one integer in one line, denoting Sugar's maximum attack value after he passes through the n domains.

Example

standard input	standard output
2 2 B C 1 1 6 B B C 2 1 C 1 2 B B	2 8

Problem C. Challenge NPC

Time limit: 1 second
Memory limit: 256 megabytes

> *Sugar is a SAT-based Constraint Solver. Constraint Satisfaction Problem (CSP) is encoded to a Boolean CNF formula, and it is solved by an external SAT solver.*
>
> — Sugar: a SAT-based Constraint Solver

It is well known that finding the chromatic number of a graph is an NPC problem. However, little tarjen claims that he can solve this problem with a simple greedy algorithm:

Color the vertices from 1 to n. For each vertex u, assign the minimum excluded (MEX) positive integer of $\{col_v | v < u, (v, u) \in E\}$ to col_u, where E is the edge set of the graph. For example, $\text{MEX}(\{1, 1, 2, 4\}) = 3$, $\text{MEX}(\varnothing) = 1$.

You want to show that this greedy algorithm is completely wrong. Construct a graph such that you can color this graph in c colors, but the greedy algorithm will color this graph with at least $c + k$ colors.

Input

The only line contains one integer k ($1 \le k \le 500$).

Output

The first line contains three integers n, m, and c ($1 \le n \le 1024, 0 \le m \le \frac{n(n-1)}{2}, 1 \le c \le n$) , representing the number of vertices, the number of edges, and the number of colors you can use to color this graph.

The following line contains n integers $col_1, col_2 \ldots col_n$ ($1 \le col_i \le c$), representing your coloring.

The following m lines each contain two integers u, v ($1 \le u, v \le n$, $u \ne v$, $col_u \ne col_v$), representing an edge in your graph.

If there are multiple solutions, output any.

Example

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

Problem D. Puzzle: Easy as Scrabble

Time limit: 1 second
Memory limit: 1024 megabytes

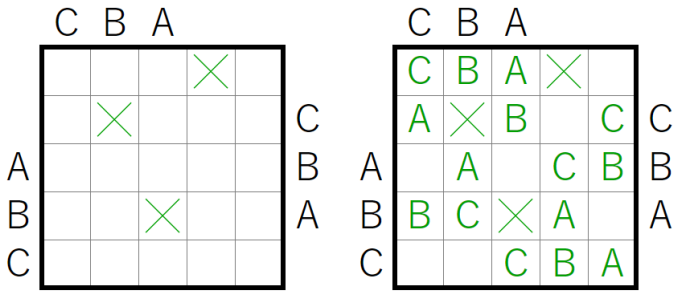
A lot of puzzle variants tend to be harder than the corresponding genres. But paradoxically, just as there is a word that becomes shorter when you add two letters to it, there is a variant which makes any genre easier.

— Freddie Hand

Grammy is a puzzle master. Today, she is playing a variant of "Scrabble" puzzle — "Easy as Scrabble".

The puzzle consists of an $n \times m$ grid. Each cell should be filled in with a capital letter, or left empty.

There are some clues outside the puzzle. The outside clue denotes the letter in the first non-empty cell from that direction. Additionally, some of the cells may contain "`x`" marks, denoting that the cell should be empty.



For example, the picture on the left illustrates an unsolved puzzle, and the picture on the right shows a solution to the puzzle.

Grammy wants to find a solution to the puzzle. Please help Grammy to find a solution, or report that no solution exists.

Input

The first line contains 2 integers n, m ($1 \le n, m \le 1000$), denoting the number of rows and the number of columns of the grid.

The next line contains a dot ("."), followed by m characters U_i, denoting the clues above the grid, followed by a dot (".").

Each of the next n lines contains a clue L_i, followed by m characters c_{ij}, followed by a clue R_i, denoting the clue to the left of the grid, the grid cells, and the clue to the right of the grid.

The next line contains a dot ("."), followed by m characters D_i, denoting the clues below the grid, followed by a dot (".").

Each of the clues U_i, L_i, R_i, D_i is either an upper case English character or a dot ("."). Each of the central cells c_{ij} is either an "`x`" or a dot (".").

All dots in the input means that the corresponding position is empty.

Output

If the solution does not exist, output "NO" on a single line.

Otherwise, output "YES" on the first line, then output n lines, each containing m characters, denoting a solution to the puzzle. Each empty cell should be represented by a dot. If there are multiple solutions, output any.

Please note that you should replace the given "x" characters in the central cells by a dot.

Please also note that you should not output the clue cells.

Examples

standard input	standard output
5 5 .CBA... ....x.. ..x...C A.....B B..x..A C.....	YES CBA.. A.B.C .A.CB BC.A. ..CBA
1 2 .... Nx.. ..O.	NO

Problem E. Team Arrangement

Time limit: 1 second
Memory limit: 512 megabytes

Misconception of the majority of teachers: w_i is a strictly increasing sequence.

— Master Chicken, colleague of Master Bo

There are n students attending the algorithm class held by Master Bo. Bo asks the students to do teamwork, and then helps them divide into teams.

Each student must belong to exactly one of the teams. Master Bo knows his students well and knows that the i-th will be satisfied with the division only if the number of students in his team is not less than l_i and not greater than r_i (including himself). Note that a team can consist of exactly one student.

You will be given n integers $w_1, w_2, \ldots, w_n$. Assume there will be m teams finally, the i-th of which consists of c_i students, the **weight** of such an arrangement will be $w_{c_1} + w_{c_2} + \cdots + w_{c_m}$.

Master Bo is now wondering how to divide students into teams such that every student will be satisfied, and the **weight** of the arrangement is maximized. Please write a program to help Master Bo.

Input

The first line of the input contains a single integer n ($1 \le n \le 60$), denoting the number of students.

In the next n lines, the i-th line contains two integers l_i and r_i ($1 \le l_i \le r_i \le n$), describing the i-th student.

The next line contains n integers $w_1, w_2, \ldots, w_n$ ($|w_i| \le 10^7$).

Output

Print a single line containing an integer: the maximum possible value of the weight. If it is impossible to find such an arrangement, please print "`impossible`" instead.

Examples

standard input	standard output
3 2 3 1 2 2 2 4 5 100	9
3 1 3 3 3 2 3 1 1 100	100
2 1 1 2 2 1 1	impossible
3 2 3 1 2 2 2 -100 -200 100000	-300

Note

The last example illustrates why ICPC requires teams of three.

Problem F. Stage: Agausscrab

Time limit: 1 second
Memory limit: 256 megabytes

Solve this problem in ??:?? or less to obtain A Gaussian Crab!

— Animal Crew

The infamous problem setting group "Animal Crew" has just prepared a problemset for the "Unicorn Cupola", which will be the newest stage of this competition. The UniCup Committee just got the list of the names of the problem setters and the number of problems that each of them set. They decided to name this stage using the following rule:

- Assume there are n problem setters. The name of the i-th problem setter is a string s_i consisting of several lowercase Latin letters, and the number of problems set by this person is a_i.
- The Committee first calculates the rank of each problem setter. The rank of the i-th problem setter r_i is defined as one plus the number of people who set **strictly more** problems than this person.
- From problem setter 1 to problem setter n, remove the last r_i characters of the i-th problem setter's name and concatenate them all together to form a string t. If the i-th problem setter's name has no more than r_i characters, then all the characters will be removed.
- Finally, capitalize the first character of t, and this will be the name of the stage.

You can see the Notes section for further explanation.

Input

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

The i-th line contains a string s_i ($2 \le |s_i| \le 20$) and an integer a_i ($1 \le a_i \le 10$), denoting the name of the i-th problem setter and the number of problems he/she set.

Output

Output a string "`Stage:`" as the beginning, then output a space, and finally output the name of the competition. The strings should be output in one line.

Examples

standard input	standard output
4 arcos 2 gausr 5 scrail 3 bei 3	Stage: Agausscrab
4 zhe 1 jiang 3 sheng 5 sai 2	Stage: Jiashen

Note

In the first example, there are 4 problem setters, and they set $a_1 = 2, a_2 = 5, a_3 = 3, a_4 = 3$ problems, then the ranks of them are $r_1 = 4, r_2 = 1, r_3 = 2, r_4 = 2$.

After removing the last $r_1 = 4$ characters from s_1, the resulting string is "a".

After removing the last $r_2 = 1$ characters from s_2, the resulting string is "gaus".

After removing the last $r_3 = 2$ characters from s_3, the resulting string is "scra".

After removing the last $r_4 = 2$ characters from s_4, the resulting string is "b".

The string t is the concatenation of the resulting strings, "agausscrab". After the final step, we can obtain the name of the stage — "Agausscrab".

You should output "`Stage: Agausscrab`" as the answer.

In the second example, after removal, the first and the last string become empty strings.

Problem G. Crawling on a Tree

Time limit: 6 seconds
Memory limit: 1024 megabytes

Yet, confronted with what seems like an endless expanse of future possibilities, I find myself inclined towards Calvino's notion in "The Baron in the Trees" - a life lived amidst the branches appears preferable to prematurely taking flight.

— *Living in a Tree*

There is a tree with n vertices, labeled by $1, 2, \ldots, n$. At the 1-st vertex, there are m turtles. Each turtle can crawl along the bidirectional edges of the tree to reach other vertices. Turtles are very heavy, so for the i-th edge, after the k_i-th time that a turtle passes it, the edge will be broken such that turtles can't pass again. Note that multiple turtles can pass the same edge at the same moment. Assuming there are cnt turtles passing the same edge at the same moment, you should count cnt times for the edge. Of course, $cnt > k_i$ is not allowed.

Your task is to command the movement of these m turtles, such that for the i-th ($2 \le i \le n$) vertex, it will be visited by at least c_i turtles. Note that if a turtle visits a vertex for multiple times, it will be counted only once. Please find a movement to minimize the total distance of all turtles, or determine it is impossible.

Input

The first line of the input contains two integers n and M ($2 \le n \le 10^4, 1 \le M \le 10^4$), denoting the number of vertices and the upper bound of m.

Each of the next $(n-1)$ lines contains four integers u_i, v_i, l_i and k_i ($1 \le u_i, v_i \le n, u_i \ne v_i, 1 \le l_i, k_i \le 10^9$), denoting a bidirectional edge between the u_i-th vertex and the v_i-th vertex, whose length is l_i, and it will be broken after the k_i-th time that a turtle passes it. It is guaranteed that the roads form a tree.

The next line contains $(n-1)$ integers $c_2, c_3, \ldots, c_n$ ($1 \le c_i \le M$), denoting the minimum number of turtles that each vertex will be visited by.

Output

Print M lines, the i-th ($1 \le i \le M$) line containing an integer, denoting the minimum total distance when $m = i$. If it is impossible to find a feasible movement, please print "`-1`" instead.

Examples

standard input	standard output
4 2 1 2 3 2 2 3 2 1 2 4 5 1 1 1 1	-1 13
4 2 1 2 3 2 2 3 2 1 2 4 5 1 2 2 2	-1 -1

Note

In the first example, when $m = 1$, it is impossible to let one turtle reach both vertex 3 and vertex 4. When $m = 2$, one of the possible solutions that minimizes the total distance is to let both turtles move from vertex 1 to vertex 2, then let the first turtle move to vertex 3, and let the second turtle move to vertex 4. The total distance traveled by the two turtles is $(3 + 2) + (3 + 5) = 13$.

Problem H. Permutation

Time limit: 2 seconds
Memory limit: 1024 megabytes

The original version of this problem asks you to find the position of the second largest number in an interval for a lot of times, but none of the problem setters knows how to solve that problem, so...

— Dummy Animal Crew

This is an interactive problem.

There's an unknown permutation of length n. You want to determine the position of number n in this permutation.

To do that, you can ask the following question:

- Choose an interval $[l, r]$ $(l < r)$ and ask for the **position** of the **second largest** number in interval $[l, r]$.

You want to determine the position of number n in no more than $\lceil 1.5 \log_2 n \rceil$ queries. Also, since we are not clever enough, our interactor can only find the second largest number in $\Theta(r - l)$, so the sum of $r - l + 1$ over your queries should not exceed $3n$.

In this problem, the interactor is non-adaptive. That is, the permutation is fixed before all queries.

Input

The first line contains an integer T $(1 \le T \le 10000)$, representing the number of test cases.

For each test case, the first line contains an integer n $(2 \le n \le 10^6)$, representing the length of the permutation. It is guaranteed that the sum of n over all test cases does not exceed 10^6.

Interaction Protocol

To ask a question, print a line of the form "`? l r`" $(1 \le l < r \le n)$. Then you should read the response from standard input.

To report the answer, print a line of the form "`! x`", representing that the position of number n is x.

After printing the answer, your program should process the next test case, or terminate if there are no more test cases.

After printing each line, do not forget to output end of line and flush the output. To do the latter, you can use `fflush(stdout)` or `cout.flush()` in C++, `System.out.flush()` in Java, or `stdout.flush()` in Python.

Example

standard input	standard output
2 5 3 3 3 2 2 	 ? 1 5 ? 1 3 ? 2 3 ! 2 ? 1 2 ! 1

Note

Please note that the examples are only for showing the correct format of the interaction process, but you are not guaranteed to obtain a definitive result after these interactions.

Problem I. Piggy Sort

Time limit: 1 second
Memory limit: 1024 megabytes

Piggy sort? I thought it was a pork order.

— Cyno, The General Mahamatra

Putata and Budada proposed a new algorithm, Piggy Sort. This algorithm can easily sort n real numbers through the following process:

- Assume that the sequence needs to be sorted is $v_1, v_2, \ldots, v_n$, which are n non-negative real numbers.
- Putata and Budada carefully select n piggies from Pigetown, where the speeds of the n piggies are exactly $v_1, v_2, \ldots, v_n$. Pigetown can be described using a coordinate axis. The i-th piggy was initially located at coordinate x_i. The initial coordinates of the piggies are **pairwise distinct**.
- All the piggies begin to run at the same time. After t seconds, the i-th piggy is at coordinate $x_i + v_i \cdot t$. Notice that the speed could be zero, which means the piggy does not move at all.
- After a considerable amount of time, the order of the piggies on the coordinate axis is the (sorted) order of the sequence $v_1, v_2, \ldots, v_n$.

Putata and Budada conducted an experiment to verify the correctness of the algorithm. However, time is money. A very long waiting time is impractical. As an alternative, they took m pictures of the piggies. To ensure an adequate amount of experimental data, they ensured that the number of photos is more than the number of elements in the array, which means m is **greater than** n.

Unfortunately, much of the information in the photos was damaged. They could get the following information from the pictures:

- The first picture was taken at time 0, while the times where the other pictures were taken are unable to be distinguished. The pictures were taken at **distinct** times.
- The coordinate of the piggies in the i-th picture are $x_{i,1}, x_{i,2}, \ldots, x_{i,n}$, while the piggies are unable to be distinguished.

Please, help Putata and Budada figure out the experiment result. You should find a sequence $r_1, r_2, \ldots, r_n$, which is the rank of the speed of the piggy with the i-th smallest coordinate in the first picture. You should guarantee $r_i < r_j$ if and only if $(v_i < v_j) \vee ((v_i = v_j) \wedge (x_{1,i} < x_{1,j}))$, and $r_1, r_2, \ldots, r_n$ is a permutation of $1, 2, \ldots, n$.

Input

The input contains multiple test cases. The first line contains an integer t ($1 \le t \le 250$), denoting the number of test cases.

For each test case, the first line contains two integers n, m ($1 \le n < m \le 500$), denoting the length of the array and the number of pictures.

The i-th of the following m lines contains n integers, the j-th of which denotes $x_{i,j}$ ($-10^7 \le x_{i,j} \le 10^7, x_{i,j} \le x_{i,j+1}$), which is the position of a piggy in the i-th picture. It is guaranteed that $x_{1,u} \ne x_{1,v}$ if $u \ne v$.

It is guaranteed that the sum of m does not exceed 500.

Output

For each test case, output n integers in one line, denoting $r_1, r_2, \ldots, r_n$.

You should guarantee that $r_1, r_2, \ldots, r_n$ is a permutation of $1, 2, \ldots, n$. If there are multiple answers, output any.

Example

standard input	standard output
3 2 4 1 2 3 4 5 6 7 8 1 2 1 1 3 4 1 2 3 6 9 9 10 15 17 12 18 21	1 2 1 3 1 2

Problem J. Even or Odd Spanning Tree

Time limit: 2 seconds
Memory limit: 512 megabytes

Oddly, even I find it odd that even though this seems like a natural rule to consider, I haven't even been able to find a single example of it existing previously. The odd problem here and there even resembles it a little bit, but even so, none of them are quite the same. How odd. Or even.

— Sam Cappleman-Lynes

You are given an undirected graph with n vertices and m edges. Assume T is a spanning tree of this graph, and let's denote $\text{Cost}(T)$ as the total weights of all the edges in T. Please find T_1 and T_2 such that:

- $\text{Cost}(T_1)$ is even, and $\text{Cost}(T_1)$ is minimized.
- $\text{Cost}(T_2)$ is odd, and $\text{Cost}(T_2)$ is minimized.

Input

The first line contains a single integer T ($1 \le T \le 10^4$), the number of test cases. For each test case:

The first line contains two integers n and m ($2 \le n \le 2 \cdot 10^5$, $1 \le m \le 5 \cdot 10^5$), denoting the number of vertices and the number of edges.

Each of the following m lines contains three integers u_i, v_i and w_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $1 \le w_i \le 10^9$), describing an undirected edge.

It is guaranteed that the sum of all n is at most $2 \cdot 10^5$, and the sum of all m is at most $5 \cdot 10^5$.

Output

For each test case, output a single line containing two integers: $\text{Cost}(T_1)$ and $\text{Cost}(T_2)$. Note that if you can't find such a spanning tree, please print "`-1`" as the cost instead.

Example

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

Problem K. Sugar Sweet 3

Time limit: 3 seconds
Memory limit: 1024 megabytes

Which problem was the Moore vote problem again?

— Anonymous Apiad, problem verifier

Pibi, Prof. Peng and Emperor Bie are playing a game. The three players hold three different types of cards. Pibi holds A cards of type 1, Prof. Peng holds B cards of type 2, and Emperor Bie holds C cards of type 3. The three players play this game with a card pile which is initially empty.

The Game lasts for $A + B + C$ Rounds, in each round:

- One player with at least 1 card left plays his card.
- If the card pile is empty, or all the cards in the card pile is of the same type as his card, the player put this card into the card pile.
- Otherwise, he takes away one card from the card pile, and throw it away, along with the card he has played.

For example, assume Pibi holds 1 card of type 1, Prof. Peng holds 3 cards of type 2, and Emperor Bie holds 2 cards of type 3. The order in which the players play their cards is [Pibi, Prof. Peng, Prof. Peng, Emperor Bie, Emperor Bie, Prof. Peng]. The cards in the card pile after each round is :

After the first round: [1] (Pibi put his card into the card pile)

After the second round :[] (Prof. Peng plays his card, and throws away one card from the card pile, along with the card he has played).

After the third round : [2] (Prof. Peng put his card into the card pile)

After the fourth round: [] (Emperor Bie plays his card, and throws away one card from the card pile, along with the card he has played)

After the fifth round: [3] (Emperor Bie put his card into the card pile)

After the sixth round: [] (Prof. Peng plays his card, and throws away one card from the card pile, along with the card he has played)

Now, assume that there're m moments that the card pile is empty, and the pile is empty at the last moment. Formally, there exist a list $t_1, t_2 \dots t_m$ ($1 \le t_1 < t_2 \cdots < t_m = A + B + C$) containing all the integers t_i such that after the t_i-th ($1 \le i \le m$) round, the card pile is empty. The three players will gain m^x bags of sugar. Here, x is a constant decided before the game. If the pile is not empty after the game, the three players will not gain any sugar.

Now the three players want to know, what's the sum of bags of sugar they can get in all possible games. Two games are considered different if and only if there exists i, such that the player of the i-th round is different. Output it modulo $10^9 + 7$.

Input

The only line contains four integers A, B, C, x ($1 \le A, B, C \le 1000, 1 \le A + B + C \le 1000, 1 \le x \le 10^9$), denoting the number of cards in Pibi's hand, the number of cards in Prof. Peng's hand, and the number of cards in Emperor Bie's hand.

Output

Output a single integer, representing the sum of bags of sugar in all possible games, modulo $10^9 + 7$.

Examples

standard input	standard output
1 2 3 1	110
4 5 7 12	881078346

Note

For the first example, there are 6 possible valid games with $m = 1$, 16 possible games with $m = 2$, and 24 possible games with $m = 3$, so the answer is $6 \times 1^1 + 16 \times 2^1 + 24 \times 3^1 = 110$.

Problem L. Challenge Matrix Multiplication

Time limit:	9 seconds
Memory limit:	256 megabytes

Take an $(\omega - 2)$, cut away 0.001 of it every day, and at the end of ten thousand generations, there will still be some left.

— Little Cyan Fish

It is well known that counting the reachability of the directed acyclic graph is hard to solve efficiently. But little tarjen claims that he has an algorithm to solve it in almost linear time. Formally, he solves the following problem and offers many test cases to show that his algorithm is correct:

- Given a directed acyclic graph (DAG) with n vertices and m edges, for each node u, find the number r_u, which is the number of vertices that can be reached starting from vertex u (including u itself).

However, clever as you are, you have found out that all the graphs little tarjen generates are special. To be more specific, let in_i be the in-degree of vertex i, and out_i be the out-degree of vertex i, then all the graphs satisfy $\sum_{i=1}^{n} |in_i - out_i| \leq 120$.

You want to show that under this constraint, solving the problem is very easy. Please write a program to solve the problem.

Input

The first line contains two integers n and m ($2 \leq n \leq 10^6$, $1 \leq m \leq 10^6$).

The following m lines, each line contains two integers u, v ($1 \leq u < v \leq n$), representing a directed edge in the graph.

It is guaranteed that $\sum_{i=1}^{n} |in_i - out_i| \leq 120$.

Output

One line contains n integers, representing $r_1, r_2, \ldots, r_n$.

Examples

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

Problem M. Triangles

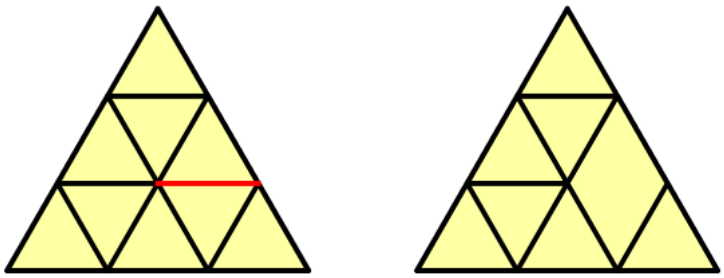
Time limit: 1 second
Memory limit: 256 megabytes

Did you know that Egypt is not the country with the most pyramids? That title actually goes to Sudan, whose Nubian pyramids number over 200, compared to roughly 118 in Egypt.

— Freddie Hand

Grammy has a triangular grid paper with a large triangle on it. The large triangle, having a side length of n, is divided into n^2 smaller triangles with side length 1. The original shape of the triangular grid paper is illustrated in the first picture.

Now, Grammy wants to set a problem with this paper, so she chooses a horizontal edge and deletes it, and asks you to find the number of remaining triangles on the paper. One example of the resulting grid is shown in the second picture.



Input

The only line contains 3 integers n, a, b ($1 \le b \le a \le n \le 10^6$), denoting the side length of the grid, the chosen row, and the chosen index of the deleted edge.

Output

Output a single integer, denoting the number of triangles in the resulting grid.

Examples

standard input	standard output
3 2 2	10
849586 233333 123456	153307446989958297

Note

In the first example, the initial triangle grid has a side length of 3. After deleting the second edge in the second horizontal row, the resulting grid has 10 triangles left.