Problem A. Outer LIS

Input file: `standard input`
Output file: `standard output`

You are given two sequences $a_1, a_2, \ldots, a_n$ and $w_1, w_2, \ldots, w_n$. You are also given q queries.

In each query, you are given two integers l and r ($1 \le l \le r \le n$, $r - l + 1 \ne n$). You are required to select a sequence of indices $p_1, p_2, \ldots, p_k$ such that:

- The length of sequence k can be chosen arbitrarily.
- $1 \le p_1 < p_2 < \cdots < p_k \le n$, and $a_{p_1} \le a_{p_2} \le \cdots \le a_{p_k}$.
- Every element in p should be chosen outside the given range $[l, r]$. In other words, $\forall i \in [1, k]$, $(p_i < l) \vee (p_i > r)$.
- $\sum_{i=1}^{k} w_{p_i}$ is maximized. You only need to report this value as the answer.

Input

The first line contains two integers n and q ($2 \le n \le 10^5$, $1 \le q \le 10^5$), denoting the length of the sequence and the number of queries.

In the next n lines, the i-th line contains two integers a_i and w_i ($1 \le a_i \le n$, $1 \le w_i \le 10^4$).

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

Output

Output q lines, the i-th ($1 \le i \le q$) of which contains an integer, denoting the answer to the i-th query.

Example

standard input	standard output
5 4 1 2 1 3 2 1 1 4 3 5 1 2 2 3 3 4 4 5	9 11 10 6

Problem B. Turn on the Light 3

Input file: `standard input`
Output file: `standard output`

Prof. Chen is currently teaching at Pigetown University. Pigetown University has a total of n lights labeled $1, 2, \ldots, n$, as well as n switches labeled $1, 2, \ldots, n$. The lighting control system at Pigetown University is quite peculiar: pressing the switch labeled d will turn on all lights whose labels are multiples of d. Since the system lacks a turn-off function, if pressing a switch does not change any light from off to on, the control system will issue a warning.

Initially, all lights are off. Prof. Chen presses the switches m times, with the i-th operation being the switch labeled a_i. Your task is to simulate the lighting control system, outputting the number of newly turned-on lights after each operation or issuing an alarm if no new lights are turned on.

Input

The input contains multiple test cases. The first line contains an integer T ($1 \le T \le 2 \cdot 10^5$), denoting the number of test cases.

For each test case, the first line contains two integers n, m ($1 \le n, m \le 2 \cdot 10^5$), denoting the number of lights and the number of times Prof. Chen presses the switches.

The following line contains m integers, the i-th integer a_i ($1 \le a_i \le n$) denotes the switch Prof. Chen presses with the i-th operation.

It is guaranteed that the sum of n does not exceed $2 \cdot 10^5$, and the sum of m does not exceed $2 \cdot 10^5$.

Output

For each test case, print m lines. For the i-th line, if the newly turned on lights in the i-th operation are not zero, output the number of lights that are newly turned on. Otherwise, output "`the lights are already on!`".

Example

standard input	standard output
2 5 3 2 4 1 100 2 100 100	2 the lights are already on! 3 1 the lights are already on!

Problem C. RDDCCD

Input file: `standard input`
Output file: `standard output`

Prof. Chen has invented a new data structure that supports "Reverse Diagonal Digits, Calculate Common Divisor" (RDDCCD) operation. This operation reverses the digits of every single number on some diagonal of a matrix and calculates the greatest common divisor of all numbers in the matrix.

Reversing the digits of a number means to write the number down in decimal form and read the number from right to left. For example, 12345 should be changed into 54321, and 2748 should be changed into 8472.

However, this data structure is currently a trade secret. To break through the technological blockade, you decided to reinvent the data structure. Please write a program that can process the query efficiently.

Input

The first line contains two integers n, q ($1 \le n \le 1000, 1 \le q \le 10^6$), denoting the size of the matrix and the number of operations.

Each of the following n lines contains n integers a_{ij} ($1 \le a_{ij} < 10^9, 1 \le i, j \le n$), denoting the numbers in the matrix. It is guaranteed that none of the numbers ends with 0.

Each of the following q lines contains a single character op_k and a single integer t_k ($op_k \in \{+, -\}, -n < t_k < 2n$), denoting an operation. If op_k is $+$, then this operation flips the digits of all a_{ij} such that $i + j = t_k$. Otherwise, this operation flips the digits of all a_{ij} such that $i - j = t_k$. It is guaranteed that at least one number is flipped in each operation.

Output

Output q lines, each containing an integer denoting the answer to each query.

Example

standard input	standard output
3 5 202 4 6 8 12 21 32 44 82 + 2 + 5 - 0 + 4 - 2	1 2 1 1 2

Problem D. Too Clever by Half

Input file: `standard input`
Output file: `standard output`

Prof. Chen has developed a new type of robot named: **S**peedy **U**niversal **G**uided **A**ssistance **R**obot. To test whether this new robot can provide stable services to the residents of Pigeland, Prof. Chen has prepared a test.

The test site consists of a horizontal runway extending from left to right, with $n+1$ locations labeled $0, 1, 2, \ldots, n$. Prof. Chen can issue the following commands to the robot:

- Move Left: Suppose the robot is currently at location x ($x > 0$). After executing this command, the robot will move to $x - 1$. The robot can not execute this command at location 0.
- Move Right: Suppose the robot is currently at location x ($x < n$). After executing this command, the robot will move to $x + 1$. The robot can not execute this command at location n.

At the beginning of the test, the robot is at location 0, and at the end of the test, the robot must also return to location 0. Prof. Chen requires that the robot must arrive at location i exactly c_i times after performing a move (excluding the initial position). Now, Prof. Chen asks you to design an operation sequence of length $\sum_{i=0}^{n} c_i$, consisting of characters L and R, where L represents Move Left and R represents Move Right, such that the robot sequentially executing the operations in this test sequence can meet the requirements. If there are multiple valid sequences, Prof. Chen wants you to find the *lexicographically smallest*† one. If no such sequence exists, you should report that.

†: A string s of length m is said to be *lexicographically smaller* than string t of length m if and only if there exists $1 \le i \le m$ that $s_j = t_j$ for $j < i$, and $s_i < t_i$. In lexicographical comparisons, we define "L" to be lexicographically smaller than "R".

Input

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

For each test case, the first line contains an integer n ($1 \le n \le 5 \cdot 10^5$).

The second line contains $n+1$ integers, the i-th integer is c_{i-1} ($1 \le c_{i-1} \le 10^6$).

It is guaranteed that the sum of n does not exceed $5 \cdot 10^5$, and the sum of $\sum_{i=0}^{n} c_i$ does not exceed $2 \cdot 10^6$.

Output

For each test case, if there exists a test sequence, output a single string consisting of $\sum_{i=0}^{n} c_i$ characters in a single line, and the string should only contain characters `L` and `R`. Otherwise, output "`Impossible`" in a single line, denoting that there is no such sequence.

Example

standard input	standard output
3 2 2 3 1 5 1 2 2 2 2 1 4 1 1 1 1 1	RLRRLL RRRRRLLLLL Impossible

Problem E. Gold Miner

Input file: `standard input`
Output file: `standard output`

Putata has recently been obsessed with playing Gold Miner. The game is set on a two-dimensional plane where the map contains n gold nuggets, with the i-th nugget located at (x_i, y_i), where $y_i < 0$.

The player is positioned at a point $(p, 0)$ on the ground. Each level has a target k, representing the number of gold nuggets that must be collected to pass the level. Due to special geological properties, when the Euclidean distance between the player and a gold nugget is s, the force required to pull the nugget is $2 \cdot s$. The energy needed to collect a nugget is equal to the *work*† required to pull it to the player's position. Putata will use the optimal strategy to pass the level while minimizing the total energy consumption.

Now, Budada has designed q randomized levels. For the i-th level, the player's position p is a uniformly and randomly generated **real number** from the interval $[a_i, b_i]$, and the required number of gold nuggets is k_i. Your task is to help Putata compute, for each randomized level, the expected minimum total energy required to pass the level, modulo $10^9 + 7$.

It can be shown that the answer can be expressed as an irreducible fraction $\frac{x}{y}$, where x and y are integers and $y \not\equiv 0 \pmod{10^9 + 7}$. Output the integer equal to $x \cdot y^{-1} \pmod{10^9 + 7}$. In other words, output such an integer a that $0 \le a < 10^9 + 7$ and $a \cdot y \equiv x \pmod{10^9 + 7}$.

†: In science, *work* is the energy transferred to or from an object via the application of force along a displacement. When the force is variable, the *work* is given by the line integral: $W = \int \mathbf{F} \cdot ds$. The *work* of pulling a gold nugget at distance s equals $\int_0^s 2x \, dx = s^2$.

Input

The first line contains two integers n, q $(1 \le n \le 2000, 1 \le q \le 5 \cdot 10^5)$, denoting the number of gold nuggets and the number of levels.

The i-th of the following n lines contains two integers x_i, y_i $(0 \le x_i \le 10^9, -10^9 \le y_i < 0)$, denoting the coordinates of the i-th gold nugget.

The i-th of the following q lines contains three integers a_i, b_i, k_i $(0 \le a_i \le b_i \le 10^9, 1 \le k_i \le n)$, denoting a level.

Output

Output q lines, the i-th line contains the answer to the i-th level.

Examples

standard input	standard output
4 4 1 -2 4 -1 4 -3 5 -2 2 3 1 0 6 4 3 4 2 4 7 2	333333339 40 666666679 9
6 10 7 -5 2 -7 2 -7 5 -3 9 -4 5 -3 2 4 1 2 10 2 5 8 3 3 9 1 5 8 5 1 2 4 4 5 3 7 10 6 3 8 3 2 9 2	333333349 846354201 625000051 406250015 143 333333477 50 273 575000054 443452410

Note

For the first sample, the answers to the four queries are $\frac{10}{3}, 40, \frac{23}{3}, 9$.

Problem F. Challenge NPC III

Input file: `standard input`
Output file: `standard output`

Prof. Chen is an expert in solving graph coloring problems. Now, he presents you with a directed graph with colors and claims that each *simple path*† of length $\leq k$ in the graph contains distinct colors.

Please check whether Prof. Chen's proclamation is correct.

†: A *simple path* is a sequence of distinct vertices $u_1, u_2, \ldots, u_L$ such that there exists an edge from u_i to u_{i+1} for each $i \in [1, L-1]$. The length of this simple path is L.

Input

There are multiple test cases. The first line contains an integer T ($1 \leq T \leq 10^5$), denoting the number of test cases. For each test case:

The first line contains two integers n, m, k ($1 \leq n, k \leq 10^5, 0 \leq m \leq 10^5$), denoting the number of vertices, the number of edges in the graph, and Prof. Chen's parameter k.

The second line contains n integers c_i ($1 \leq c_i \leq 50$), denoting the color of each vertex.

Each of the following m lines contains two integers u_j, v_j ($1 \leq u_j, v_j \leq n$), denoting a directional edge in the graph.

It is guaranteed that neither the sum of n nor the sum of m exceeds 10^5.

Output

For each test case, if Prof. Chen's proclamation is correct, output "`YES`" in a line; otherwise, output "`NO`" in a line.

Example

standard input	standard output
3 3 2 2 1 2 1 1 2 2 3 3 3 2 1 2 1 1 2 2 3 1 3 3 3 50 1 1 2 1 2 2 3 1 3	YES NO NO

Problem G. Tariff-ied

Input file: `standard input`
Output file: `standard output`

Prof. Chen is running for president of Pigeland and has proposed a tariff strategy that could bring enormous profits to the country.

For a trade route consisting of n countries, all tariffs initially start at 0%. Each time the i-th country increases its tariff, it rises by l_i%. If the i-th country raises its tariff t_i times, the tariff becomes $\frac{l_i \cdot t_i}{100}$, meaning a commodity priced at x passing through this country will become $\left(1 + \frac{l_i \cdot t_i}{100}\right) \cdot x$.

Prof. Chen plans to raise tariffs a total of k times, which can be distributed arbitrarily among any of the countries. Your task is to help Prof. Chen determine the maximum possible price multiplier for a commodity passing through all n countries in sequence after optimally allocating these k tariff increases.

Input

The input contains multiple test cases. The first line contains an integer T ($1 \le T \le 5\,000$), denoting the number of test cases.

For each test case, the first line contains two integers n, k ($1 \le n \le 40, 1 \le k \le 10^6$), denoting the number of countries and the total number of tariff increases.

The following line contains n integers. The i-th integer l_i ($1 \le l_i \le 100$) denoting the increase rate of the tariff of the i-th country is l_i%.

Note that there is **no additional constraints** on the sum of n and the sum of k.

Output

Output a single real number in a single line, denoting the answer.

Your answer will be considered correct if its absolute or relative error does not exceed 10^{-4}. Formally, let your answer be a, and the jury's answer be b. Your answer will be considered correct if $\frac{|a-b|}{\max(1,|b|)} \le 10^{-4}$.

Example

standard input	standard output
2 3 5 50 100 50 10 1000000 1 2 3 4 5 6 7 8 9 10	9 3.63944265968e+36

Note

For the first test case, the optimal allocation of tariff increases is $2, 2, 1$. The tariffs for the three countries become $100\%, 200\%, 50\%$ respectively, so the final answer is $2 \cdot 3 \cdot 1.5 = 9$.

Problem H. Sweet Sugar IV

Input file: `standard input`
Output file: `standard output`

To celebrate the coming Children's Day, a sugar collection game will be hosted. The game is played on a grid map of $n \times m$ cells. The rows and columns of the grid are numbered from 1 to n and 1 to m, respectively.

There are d pieces of sugar on the map. The i-th piece of sugar is located at the cell (x_i, y_i), the sweetness of which is s_i. Every cell contains at most one piece of sugar.

There are k pawns on the map. Initially, all the pawns are located at the cell (S_x, S_y). In each operation, the player can move a pawn from (x, y) to $(x+1, y)$ or $(x, y+1)$. Note that a cell can contain multiple pawns. Whenever a pawn is located at a cell containing a piece of sugar, the player will take that sugar, and of course, each piece of sugar can not be taken more than once. The player can do legal operations for an arbitrary number of times. The final score of the player is equal to the sum of sweetness among all the pieces of taken sugar.

There are also p rectangular obstacles on the map. Assume the i-th rectangle is $[x_1, x_2] \times [y_1, y_2]$, then a cell (x, y) is considered to be dangerous if and only if $x_1 \le x \le x_2$ and $y_1 \le y \le y_2$. The player should never move the pawns into any dangerous cell. Fortunately:

- The edge of the map will never be dangerous (i.e. $2 \le x_1 \le x_2 \le n-1$, $2 \le y_1 \le y_2 \le m-1$).
- No two obstacles will overlap or touch each other. In other words, two cells (x, y) and (x', y') are considered to be adjacent if and only if $\max\{|x-x'|, |y-y'|\} \le 1$. If there are two adjacent dangerous cells, they must belong to the same obstacle.

You are now training for this sugar collection game. Given all the information about the game, try to find the maximum possible value of the final score that you can achieve. To become the master of this game, you will replay the game for q times, each time the location of the starting cell (S_x, S_y) and the number of pawns k may be changed.

Input

The first line contains five integers n, m, p, d and q ($1 \le n, m \le 10^5$, $n \times m \le 10^6$, $0 \le p \le 10^5$, $1 \le d \le 2 \cdot 10^5$, $1 \le q \le 3 \cdot 10^5$), denoting the size of the map, the number of obstacles, the number of pieces of sugar, and the number of replay times, respectively.

Each of the following p lines contains four integers x_1, y_1, x_2 and y_2 ($2 \le x_1 \le x_2 \le n-1$, $2 \le y_1 \le y_2 \le m-1$), describing each rectangular obstacle. It is guaranteed that no two obstacles will overlap or touch each other.

Each of the following d lines contains three integers x_i, y_i and s_i ($1 \le x_i \le n$, $1 \le y_i \le m$, $1 \le s_i \le 10^9$), describing each piece of sugar. It is guaranteed that every cell contains at most one piece of sugar, and no piece of sugar will be located at a dangerous cell.

Each of the following q lines contains three integers S_x, S_y and k ($1 \le S_x \le n$, $1 \le S_y \le m$, $1 \le k \le 5$), denoting the location of the starting cell and the number of pawns in a replay. It is guaranteed that (S_x, S_y) is safe.

Output

Output q lines, the i-th ($1 \le i \le q$) of which containing an integer, denoting the maximum value of the final score that you can achieve in the i-th replay.

Examples

standard input	standard output
3 3 1 4 5 2 2 2 2 1 1 1 3 3 1 1 2 3 2 1 5 1 1 1 1 1 2 1 3 2 1 2 2 2 1 1	7 10 1 4 6
5 7 1 3 4 2 3 3 4 3 1 1 2 2 10 5 4 100 1 2 1 1 2 2 1 1 2 2 7 1	110 110 111 0

Problem I. Version Number

Input file: `standard input`
Output file: `standard output`

Grammy is playing a game that has multiple versions. Each version has a unique version number composed of numeric segments separated by dots (e.g., "123.24.5155").

When comparing two versions A and B, we compare their corresponding numeric segments **in order**. If the corresponding segment of A is larger than that of B, then version A is considered newer. If the segments are equal, we proceed to the next pair of segments. It is guaranteed that all version numbers in the game have the same number of segments.

Examples

- 1.2.3 vs 1.2.4 $\rightarrow$ B is newer (third segment: $3 < 4$)
- 3.11 vs 3.8 $\rightarrow$ A is newer (second segment: $11 > 8$)
- 3.0 vs 3.0 $\rightarrow$ Equal (all segments equal)

Since newer versions contain additional content, Grammy wants to experience the latest version. Given two version numbers A and B, determine which one is newer and output the result (A or B or "Equal").

Input

The input contains multiple test cases. The first line contains an integer T ($1 \leq T \leq 100$) denoting the number of test cases.

For each test case, there are two lines respectively describing the version number of A and B. It is guaranteed that there are at most 3 dots in each version number, the length of each numeric segment doesn't exceed 9, and the numbers in each segment don't contain any leading zeros.

Output

For each test case, output one line indicating which version is newer, `A` or `B` or "`Equal`".

Example

standard input	standard output
3 1.0.2 1.1.0 123.0 2.100 0 0	B A Equal

Problem J. Chocolate Sweet

Input file: `standard input`
Output file: `standard output`

Putata and Budada have a chocolate bar, which is divided into n rows and m columns, making a total of $n \cdot m$ small chocolate pieces. The deliciousness of the piece in the i-th row and j-th column is $a_{i,j}$. They decide to take turns eating the chocolate, with Putata going first. When eating, they can choose to eat the last few remaining rows or the last few remaining columns, but they cannot choose to eat nothing. Formally, if the remaining chocolate consists of pieces (i, j) satisfying $1 \le i \le a$ and $1 \le j \le b$, the player can choose $1 \le x \le a$ and eat all chocolate pieces (i, j) satisfying $a - x + 1 \le i \le a$ and $1 \le j \le b$, or choose $1 \le y \le b$ and eat all chocolate pieces (i, j) satisfying $1 \le i \le a$ and $b - y + 1 \le j \le b$.

Their teacher, Prof. Chen, always believes that humility is a virtue. Therefore, Prof. Chen has a tolerance level s. If, after Putata or Budada eats some chocolate, the total deliciousness of the remaining chocolate is less than or equal to s, the last person to eat will be criticized. Putata and Budada are very smart and will always choose the best way to avoid being criticized. According to the survey results, Prof. Chen's tolerance level has q possible values. For each $1 \le i \le q$, you need to determine who will be criticized if Prof. Chen's tolerance level is s_i.

Input

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

For each test case, the first line contains three positive integers n, m, q ($1 \le n \cdot m \le 2 \cdot 10^5, 1 \le q \le 2 \cdot 10^5$), denoting the size of the chocolate.

The i-th of the next n lines contains m integers, the j-th integer is $a_{i,j}$ ($1 \le a_{i,j} \le 10^9$), denoting the deliciousness of the piece of chocolate in the i-th row and j-th column.

The i-th of the next q lines contains an integer s_i ($0 \le s_i < \sum_{i=1}^{n} \sum_{j=1}^{m} a_{i,j}$), denoting a query.

It is guaranteed that the sum of $n \cdot m$ does not exceed $2 \cdot 10^5$, and the sum of q does not exceed $2 \cdot 10^5$.

Output

For each test case, output q lines, the i-th line contains a string "`Putata`" or "`Budada`", denoting the one being criticized.

Example

standard input	standard output
2 3 3 2 1 2 1 1 1 3 1 4 2 1 5 2 2 1 1000000000 1000000000 1000000000 1000000000 0	Putata Budada Putata

Problem K. Wavelike Finding

Input file:	`standard input`
Output file:	`standard output`

In this problem, you are given two fixed parameters k and W.

We call a sequence $a_1, a_2, \ldots, a_{2k}$ **shaky** if and only if $\sum_{i=1}^{k}(k-i+1)(a_i + a_{2k-i+1}) \geq W$. Note that the length of the sequence must be $2k$.

We also call a sequence $b_1, b_2, \ldots, b_m$ **wavelike** if and only if:

- $m \geq 2k$.
- It contains at least one **shaky** subsequence. Note that the **shaky** subsequence is not required to be contiguous.

You are also given a sequence $s_1, s_2, \ldots, s_n$, select disjoint **wavelike** consecutive subsequences from it as many as possible. Note that the selected consecutive subsequences don't need to cover the entire sequence. Here, "consecutive subsequence" means that it must be contiguous (i.e. a substring).

Input

The first line contains three integers n, k and W ($2 \leq n \leq 2 \cdot 10^5$, $1 \leq k \leq \frac{n}{2}$, $|W| \leq 10^{17}$).

The second line contains n integers $s_1, s_2, \ldots, s_n$ ($|s_i| \leq 10^6$).

Output

Output a single line containing an integer, denoting the maximum possible number of disjoint **wavelike** consecutive subsequences that you can select.

Examples

standard input	standard output
9 2 25 6 4 3 5 7 1 2 5 8	2
9 2 30 6 4 3 5 7 1 2 5 8	1
2 1 1 0 0	0
2 1 -6 -1 -5	1

Problem L. Nailoongs Always Lie

Input file: `standard input`
Output file: `standard output`

Grammy arrives on a planet inhabited by mythical creatures, including a species called Nailoong. Through her interactions, she discovers two rules:

- Nailoongs always lie.
- All other creatures may either tell the truth or lie.

There are n creatures on the planet. The i-th creature claims: "The creature numbered a_i is a Nailoong."

Please determine the maximum possible number of Nailoongs among these creatures that could possibly exist under these rules.

Input

The first line of input contains one integer n ($1 \le n \le 10^5$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$), where the i-th creature claims that the a_i-th creature is a Nailoong.

Output

Output one integer, the maximum possible number of Nailoongs among these creatures that could possibly exist under these rules.

Example

standard input	standard output
6 4 1 1 5 6 5	4

Problem M. Master of Both VII

Input file: `standard input`
Output file: `standard output`

This is an interactive problem.

Prof. Chen is proficient in computational geometry and interactive problems. After teaching a lesson on *polygon triangulation*†, Prof. Chen prepared the following homework assignment.

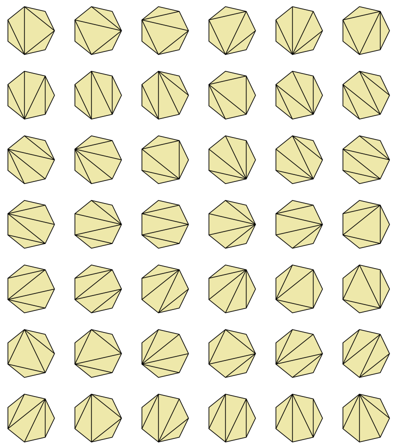
There is a regular n-gon with vertices labeled $1, 2, \dots, n$ in counter-clockwise order. There is a hidden triangulation consisting of $n-3$ edges connecting the vertices. You can make the following query:

- "`?` u v": Query the information about the edge connecting u and v. If the edge u-v exists in the triangulation, it will return 0. Otherwise, it will return the number of edges in the triangulation that intersect with (u, v) at **non-endpoint** points.

Your task is to determine this triangulation using no more than $n-3$ queries.

In this problem, the interactor is **not adaptive**, which means that the triangulation is predetermined and will not change during your interaction with the interactor.

†: In computational geometry, *polygon triangulation* is the partition of a simple polygon P into a set of triangles, i.e., finding a set of triangles with pairwise non-intersecting interiors whose union is P.



Pic. 1: An illustration of all 42 **VII**-gon triangulations (adapted from *Wikipedia*, CC BY-SA 3.0).

Input

The input contains multiple test cases. The first line contains an integer T ($1 \le T \le 2500$), denoting the number of test cases.

For each test case, the first line contains an integer n ($4 \le n \le 100$), denoting the polygon is a regular n-gon.

It is guaranteed that the sum of n does not exceed 10^4.

Interaction Protocol

For each test case, you can make no more than $n-3$ queries. To make a query, output "`?` u v" ($1 \le u \ne v \le n$, u and v are non–adjacent on the convex polygon) on a separate line, then read the response from standard input. In response to the query, the interactor will return an integer x representing the result of the query. If $x = -1$, it means your query is invalid or exceeds the allowed number of queries, and you should terminate your program, resulting in a Wrong Answer verdict.

When you are ready to submit your answer, output "`!` a_1 b_1 a_2 b_2 $\dots$ a_{n-3} b_{n-3}" on a separate line, representing the edges in the triangulation are (a_i, b_i) for $1 \le i \le n-3$. The answer submission does not count toward the $n-3$ query limit. In response to your answer, the interactor will return an integer r. If $r = 1$, it means your answer is correct; if $r = 0$, it means your answer is incorrect, and you should terminate your program, resulting in a Wrong Answer verdict.

After submitting your answer, your program should proceed to the next test case. Once all test cases are processed, your program should terminate.

After printing a query or submitting the answer, do not forget to output the end of the line and flush the output. To do this, use `fflush(stdout)` or `cout.flush()` in C++, `System.out.flush()` in Java, or `stdout.flush()` in Python.

Example

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