Problem A. (A + B) mod P

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

AGI (artificial general intelligence) is around the corner, but it still cannot add two integers modulo prime. In this task, you can speed up the AGI revolution by training a one-layer perceptron model with n neurons for a fixed **prime** p. This model takes two integers a and b as inputs and calculates $(a+b)$ mod p. The architecture of the model is:

$$ReLU(a_{one-hot}W_{input} + b_{one-hot}W_{input})W^T_{output}$$

where

$$ReLU(x) = max(x, 0)$$

and $x_{one-hot}$ is a matrix of size $1 \times p$, which consists of all zeros and a single one in column x.

For example, if $p = 4$, $0_{one-hot} = \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix}$, $2_{one-hot} = \begin{pmatrix}0 & 0 & 1 & 0\end{pmatrix}$.

In other words, you must choose the number of neurons n and generate two real matrices W_{input} and W_{output} of size $p \times n$.

To add integers a and b three steps are executed:

1. The pointwise sum of 0-indexed rows a and b of the W_{input} is calculated.
2. All negative numbers in this sum are replaced with zeros. Let's call the generated row M.
3. The score of integer i is defined as the scalar product of M and i-th row of the W_{output}. The integer with the highest score should equal $(a + b)$ mod p.

See the example explanation for better understanding.

You need to generate two real matrices W_{input} and W_{output} that for all pairs $(a, b), 0 \le a, b < p$, the output produced by the model is correct.

Input

The only line of the input consists of the integer p ($3 \le p \le 100$). It is guaranteed that p is prime.

Output

The first line of the output should contain the integer n ($1 \le n \le 25$) — the number of neurons in the model.

In the following p lines, output matrix W_{input}. Each line should consist of n real numbers not greater than 1000 by the absolute value.

After that, output matrix W_{output} in the same format.

Example

standard input	standard output
3	4 2.5 3.0 -0.5 -3.0 -3.0 0.0 0.5 2.0 -0.5 0.5 -3.0 1.5 1.5 1.0 -2.0 2.5 -3.0 3.0 -1.0 2.0 -0.5 2.5 0.5 2.0

Note

Let's calculate $(1 + 2) \bmod 3$. First, we add rows $[-3.0, 0.0, 0.5, 2.0]$ and $[-0.5, 0.5, -3.0, 1.5]$ and get $[-3.5, 0.5, -2.5, 3.5]$. Then, we remove all negative numbers: $[0.0, 0.5, 0.0, 3.5]$.

Then, we calculate the score for each possible result:

- 0: $[0.0, 0.5, 0.0, 3.5] \times [1.5, 1.0, -2.0, 2.5] = 0.0 \cdot 1.5 + 0.5 \cdot 1.0 + 0.0 \cdot -2.0 + 3.5 \cdot 2.5 = 9.25$.
- 1: $[0.0, 0.5, 0.0, 3.5] \times [-3.0, 3.0, -1.0, 2.0] = 8.5$.
- 2: $[0.0, 0.5, 0.0, 3.5] \times [-0.5, 2.5, 0.5, 2.0] = 8.25$.

Row 0 has the highest score 9.25, and $(1 + 2) \bmod 3$ indeed equal 0.

Problem B. The Best Wife

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

Sofia is the best wife, and she constantly comes up with new ideas on how to spend time with her husband. Idea i could be described by a segment $[l_i..r_i]$. It means Sofia wants to visit some event that starts at day l_i and ends at day r_i. If she decides to visit an event, she fully commits to it and is there every day from l_i to r_i inclusive.

Sofia can't visit more than one event each day. Also, if some event ends on day d and another event starts on day d, she can only visit one of them.

After each new idea comes to her mind, she wonders what the biggest number of events she could visit.

Input

The first line contains one integer n ($1 \le n \le 3 \cdot 10^5$) — the number of ideas.

Each of next n lines contain two integers l_i and r_i ($1 \le l_i \le r_i \le 6 \cdot 10^5$) — description of the i-th idea.

Output

For each i in a range $1..n$, print one integer — the maximum number of events out of $1, 2, ..., i$ that Sofia can visit.

Example

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

Problem C. Cardinality

Input file:	`standard input`
Output file:	`standard output`
Time limit:	10 seconds
Memory limit:	512 megabytes

This is an interactive problem.

There are n sets numbered 1 to n, the i-th of them containing only the integer i. You need to process q queries. The i-th query (1-indexed) is described by a pair of integers (X_i, Y_i). It creates a set number $n + i$, which consists of a union of integers from sets X_i and Y_i.

Each time you create a new set, you must report the number of different integers in this set. **You don't need to report the exact result.** If the correct size of the set is A and you print integer B, it will be accepted if $0.5 \cdot A \leq B \leq 2.0 \cdot A$.

Flushing the output could be costly on some Judging Systems, so all queries are split into batches of size 50. Your solution should read the first 50 queries, print the answers for all of them, flush the output, read the second batch of 50 queries, print the answers for them, flush the output, and so on. The last batch could consist of less than 50 queries.

The interactor is not adaptive, which means for each test, the sequence of queries is always the same for all runs.

Input

The first line contains two integers n and q ($1 \leq n \leq 50\,000, 1 \leq q \leq 5 \cdot 10^5$) — the initial number of sets and the number of queries.

The next q lines contain queries. The i-th of them contains two integers X_i and Y_i ($1 \leq X_i, Y_i < n + i$).

Output

For each query, print one integer in a separate line — your estimate of the size of the created set.

Don't forget to flush output after every 50th query!

Example

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

Problem D. 3D

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

n points $a_1, a_2, ..., a_n$ are **randomly** generated inside a cube of size 1.

You are given a matrix d with $d_{i,j} = d_{j,i} = dist(a_i, a_j) + rand(-0.1..0.1)$ and $d_{i,i} = 0$. Here $dist(p, q)$ is the distance between points $\sqrt{(p_x - q_x)^2 + (p_y - q_y)^2 + (p_z - q_z)^2}$ and $rand(-0.1..0.1)$ is a random shift chosen uniformly from interval $[-0.1..0.1]$. Shifts for different pairs of points are chosen independently.

You need to construct a list of points $b_1, b_2, ..., b_n$ such that $\forall_{i,j} |dist(b_i, b_j) - d_{i,j}| \le 0.1$.

Input

The first line contains one integer n ($1 \le n \le 10$) — the number of points.

The next n lines contain the description of matrix d. The i-th line contains n real values $d_{i,j}$ ($-0.1 \le d_{i,j} \le \sqrt{3} + 0.1$). Each value is given with 6 digits after the decimal point.

It is guaranteed that $d_{i,i} = 0$ and $d_{i,j} = d_{j,i}$.

Output

Print n lines describing the points. i-th line should contain three real numbers x_i y_i z_i ($-10.0 \le x_i, y_i, z_i \le 10.0$).

Example

standard input	standard output
4 0.000000 0.758400 0.557479 0.379026 0.758400 0.000000 0.516608 0.446312 0.557479 0.516608 0.000000 0.554364 0.379026 0.446312 0.554364 0.000000	0.210269 0.581333 0.000000 0.090086 0.000000 0.458722 0.000000 0.498388 0.501723 0.204618 0.204262 0.075724

Note

This problem has 30 test cases.

Problem E. Equal Strings

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

This is an interactive problem.

$n-1$ **random** binary strings s_i of length 50 are generated. Then, one of the strings is duplicated, and all n strings are shuffled. You need to find indexes of equal strings.

You can ask at most 25 000 queries. Each query is a pair (i, j). The answer to the query is the Hamming distance between s_i and s_j — the number of positions where they differ.

It is guaranteed that all strings are picked randomly from the uniform distribution. Also, the list of strings is fixed for each test — interactor is not adaptive.

Interaction Protocol

First, the interactor prints one integer n ($2 \le n \le 1000$) — the total number of strings.

Then, you can ask at most 25 000 queries in the format "i j" ($1 \le i, j \le n, i \ne j$). Don't forget to flush the output after each query!

For each query, you will receive one integer d ($0 \le d \le 50$) — the Hamming distance between s_i and s_j. After receiving $d = 0$, your program should terminate.

Example

standard input	standard output
4 21 23 21 0	 1 2 2 3 1 4 2 4

Note

Strings in the sample:

- 10110000010010010101011100011000000011001011000101
- 10101101000111001111100000010010000100101011100101
- 10011110110010011111100111011000010100011101111011
- 10101101000111001111100000010010000100101011100101

Problem F. Fast Tree Queries

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

You are given a tree consisting of n nodes. Initially, node i has the integer i written on it. You need to process q queries of two types:

- **+ a v x** — add x to all integers written on a simple path from node a to node v.
- **? a v** — calculate the xor of all integers written on a simple path from node a to node v.

Input

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

Each of the next $n-1$ lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$) — edges of the tree.

Each of the next q lines contains queries in the format "+ a v x" ($1 \le a, v \le n$, $1 \le x \le 10^4$) or "? a v" ($1 \le a, v \le n$).

Output

For each query of the second type, print the answer in a separate line.

Example

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

Problem G. Geo Sharding

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

For a given integer n, you must color each cell of an $n \times n$ grid into one of $C = 10 + \lfloor \frac{n^2}{100} \rfloor$ colors. Each specific color could be used at most 150 times.

For each cell (r, c), the set of colors used in cells (r_i, c_i) where $(r - r_i)^2 + (c - c_i)^2 \leq 100$, should contain **at most 8 different colors**.

Input

The only line contains a single integer n ($1 \leq n \leq 1000$) — the grid size.

Output

Print n lines with n integers $1 \leq a_{r,c} \leq C$ in each.

Example

standard input	standard output
3	1 2 3 4 5 6 7 8 8

Problem H. Have You Seen This Subarray?

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

There is an array a with $a[i] = i$ initially. m operations are performed with this array. During each operation, two **random** indexes i and j are chosen, and elements $a[i]$ and $a[j]$ are swapped.

You need to answer q queries. Each query is array b_1, b_2, ..., b_k. You need to find the first time when b was a continuous subarray of a.

Indexes in swap operations are guaranteed to be chosen independently from a uniform distribution.

Input

The first line contains three integers n, m, and q ($1 \le n, m, q \le 10^5$).

Each of the following m lines contains two integers i and j ($1 \le i < j \le n$) — descriptions of swap operations.

The next q lines contain descriptions of the queries. Each description starts with integer k ($1 \le k \le n$) and then k integers b_1, b_2, ..., b_k ($1 \le b_i \le n$). It is guaranteed that the sum of all k doesn't exceed 10^5.

Output

For each query described by the array b, you need to print the number of operations performed before b became a continuous subarray of a. If the initial array a contained b, print 0. It is guaranteed that b was a subarray of a at some point.

Example

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

Problem I. Interactive Casino

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

This is an interactive problem.

You are given 1000$. You play a game consisting of $T = 1000$ rounds.

In each round, the jury **randomly** picks an integer b from the range $[1..m]$, where m is the amount of money you currently have. Each integer in this range has an equal probability of being chosen. You can either play this round or skip it. If you play in the round, one of the outcomes is selected with equal probability:

- You need to pay b dollars.
- Jury gives you $2b$ dollars.

If, at some point, you have more than 10 000 dollars, you win. You lose if you have 0$ or the game ended (you played or skipped all T rounds).

Interaction Protocol

The first line contains one integer T ($T = 1000$ or $T = 5$) — the number of rounds. It is guaranteed that $T = 5$ is used only in the sample. Your solution is considered correct on the sample if you have a positive amount of money after T rounds.

Each of the T rounds starts with a command "ROUND m b", where m is the amount of money you currently have and b is a bet chosen by the jury for this round. You should respond with either "PLAY" or "SKIP". Don't forget to flush the output after each response!

If you have more than 10 000 dollars after some round, you will get the word "WIN" instead of the next round. If you lose, you will get the word "LOSE". Your program should terminate immediately after receiving "WIN" or "LOSE".

Example

standard input	standard output
5 ROUND 1000 43 ROUND 957 433 ROUND 957 525 ROUND 957 125 ROUND 832 685 WIN	 PLAY SKIP SKIP PLAY SKIP

Note

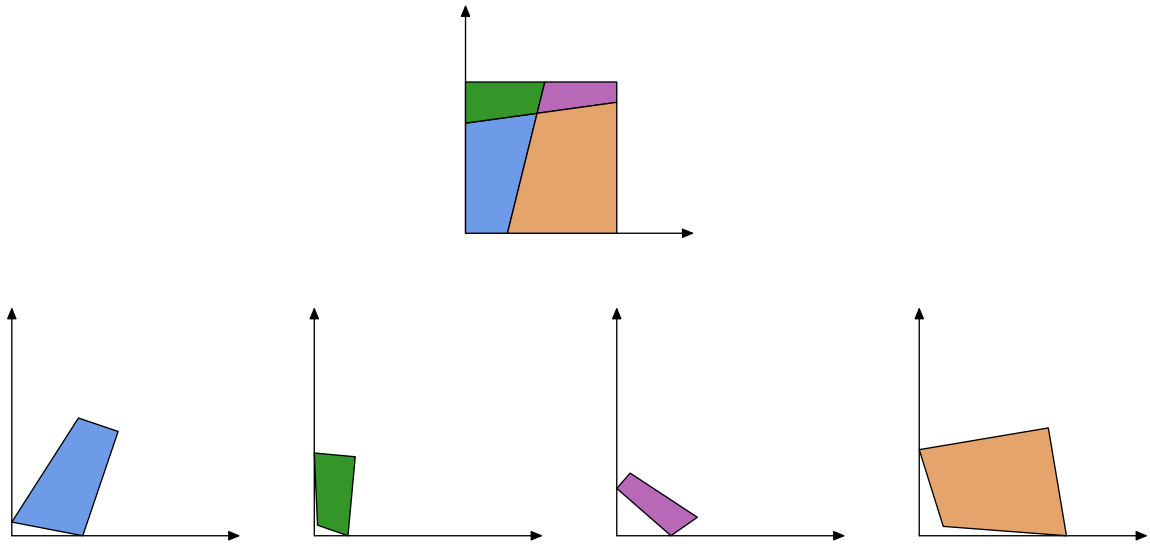
This problem has one sample test and 50 real test cases. Each real test case has some fixed random seed. Note that your decision to participate or not participate in the round affects how the random number generator is used, so all rounds after that will be different.

Problem J. Jigsaw Puzzle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

The following operation is performed at most 20 times with a square piece of paper of the size 1×1. Two points inside the square are picked **uniformly at random**, and a line is drawn through them.

Later, the paper is cut along all of the lines into n pieces. Then, pieces are randomly rotated (but not flipped), shifted, and given to you. You need to figure out the initial position of each piece.



Input

The first line contains one integer n ($2 \le n$) — the number of pieces. The descriptions of the pieces follow.

Each description starts with an integer m ($3 \le m$) — the number of vertices. Each of the following m lines contains two real numbers x_i and y_i ($0.0 \le x_i, y_i \le 2.0$) given with 12 digits after the decimal point. The vertices are given in a counter-clockwise order.

Output

You need to print pieces in the order given in the input.

For each piece, print the coordinates of the vertices in the order given in the input. Each coordinate must fulfill the condition $0.0 \le x_i, y_i \le 1.0$.

If there are several possible ways to put pieces inside a 1×1 square such that they do not intersect, you can print any of them.

The answer will be considered incorrect if the intersection area of any two polygons is bigger than 10^{-6}.

Example

standard input	standard output
4 4 0.440405375916 0.778474079786 0.000000000000 0.090337001520 0.469097990019 0.000000000000 0.702887505082 0.689470121906 4 0.222810526978 0.000000000000 0.270828246634 0.522212063829 0.000000000000 0.547114887265 0.021480010612 0.069880870008 4 0.000000000000 0.312825941471 0.358219176380 0.000000000000 0.532830100286 0.122181578260 0.088431750275 0.414089758021 4 0.158867722074 0.061734605990 0.973532298476 0.000000000000 0.853551564066 0.712811281737 0.000000000000 0.569141075980	0.277161636 -0.000000000 0.473262431 0.793116645 0.000000000 0.728029248 0.000000000 0.000000000 0.524415047 1.000000000 0.000000000 1.000000000 0.000000000 0.728029248 0.473262431 0.793116645 1.000000000 1.000000000 0.524415047 1.000000000 0.473262431 0.793116645 1.000000000 0.865558433 0.473262431 0.793116645 0.277161636 -0.000000000 1.000000000 -0.000000000 1.000000000 0.865558433

Problem K. Knapsack

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

$n = 10^4$ integers $a_1, a_2, ..., a_n$ are **randomly** and independently generated from the range $[1..10^{12}]$.

For each integer, you need to either discard it or put it into one of the sets A, B, or C. The sum of the integers in the set A should be equal to the sum in the set B and equal to the sum in the set C.

Each set should contain at least one integer. If there are multiple solutions, you could print any of them. It is guaranteed that the answer always exists for all tests in the system.

Input

The first line contains one integer n ($n = 10^4$ or $n = 6$) — the number of elements in the array.

The second line contains integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^{12}$).

All tests except the sample are guaranteed to have $n = 10^4$, and integers a_i are randomly generated from a uniform distribution.

Output

Print one line consisting of n characters. For each integer, print "." if you want to discard it. Otherwise, print "A", "B" or "C" to indicate the set to put this integer.

Example

standard input	standard output
6 4 3 8 1 5 4	ABC.BA

Note

This problem has one sample test and 50 real test cases.

Problem L. London Underground

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

You are given a graph of the London Underground with 426 stations and 505 two-way connections between them. You are also given some subset of stations. You need to count the number of ways to add more (possibly zero) stations to this set so that no two stations are directly connected.

Two ways of adding stations are considered different if some station is present in one set and not present in another.

The number of ways could be big, so you need to output it modulo 998 244 353.

Input

The first line contains one integer m ($m = 505$) — the number of connections.

The following m lines contain two stations each. Each station is a string of alphabetical characters, underscores, or digits.

It is guaranteed that the graph is exactly the same in all tests. It doesn't contain self-loops and duplicate edges.

The following line contains one integer k ($0 \le k \le 426$) — the number of stations in the initial set. The following k lines contain station names in the same format.

Output

Print the number of ways to extend the set modulo 998 244 353.

Example

standard input	standard output
505 Baker_Street Regents_Park Charing_Cross Embankment Edgware_Road__Bakerloo_ Marylebone Embankment Waterloo Harlesden Willesden_Junction Harrow_and_Wealdstone Kenton Kensal_Green Queens_Park Kenton South_Kenton ... 2 Baker_Street Liverpool_Street	159589981

Note

You can download the complete sample input at `https://pastebin.com/yuMX9tRL`.

You can check the official map at `https://content.tfl.gov.uk/standard-tube-map.pdf`. This link is provided for reference only. It may have some differences compared to the graph in the sample.

Problem M. Meta

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

Teamwork is critical in programming competitions! Effective communication and solution discussion among team members is key before implementation. Sometimes, team members might choose incorrect solutions and spend excessive time coding them. Other times, someone may suggest a more straightforward and simpler approach.

A team of three members is taking part in the competition. For each team member, you are given the time it takes for them to implement each of the problems. However, they share just one computer, so they could only work on one problem at a time. The contest has a duration of 300 minutes. Your task is to compute the maximum number of different problems the team can solve. You can neglect the time required for switching between problems — if the total time required for solving problems doesn't exceed 300 minutes, they could do it in time.

Let's look at the sample:

- The first team member likes geometry and is most effective at implementing problems such as "3D" or "JigsawPuzzle".
- The second team member likes constructive problems and would be the best for problems like "EqualStrings", "GeoSharding" or "InteractiveCasino". Although the problem "AplusB" may not be time-intensive to implement, the solution idea is not straightforward, so other team members couldn't solve it at all. This team member codes in Java and cannot solve "FastTreeQueries" due to the strict Time Limit.
- The last team member has a good library with already implemented data structures, so they could implement problems like "TheBestWife" faster than others.

Input

The first line contains one integer n ($1 \le n \le 14$) — the number of problems in the contest.

Each of the next n lines contains the description of the problem *name* t_1 t_2 t_3 ($-1 \le t_i \le 300$) — the problem's name and the number of minutes required for each participant to implement it. The problem name consists of at most 30 alphanumeric characters. If t_i equals -1, this team member could not implement this problem at all.

Output

Print single integer — the maximum number of problems the team could solve during the contest.

Example

standard input	standard output
13 AplusB -1 20 -1 TheBestWife 80 90 60 Cardinality 40 50 30 3D 40 -1 70 EqualStrings 25 15 20 FastTreeQueries 120 -1 40 GeoSharding 25 20 30 HaveYouSeenThisSubarray 80 90 60 InteractiveCasino 50 20 30 JigsawPuzzle 40 50 80 Knapsack -1 40 200 LondonUnderground -1 200 40 Meta 5 7 10	10