Problem A. Magician

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

This is an interactive problem.

A magician is going to play a game with $N = 20$ gnomes. Each gnome i wears a hat of color C_i ($1 \le C_i \le 5 \cdot 10^5$). Each gnome can see the color of all other gnomes' hats except his own. The game consists of two phases:

- **Phase 1:** Each gnome has a card. They are allowed to write either 0 or 1 on the card. They cannot communicate with each other while writing their numbers, i.e., all of them write simultaneously and independently.
- **Phase 2:** The magician collects all the cards from the gnomes and creates an array A of size N, where A_i is the number written by the i-th gnome (the order is preserved). Each gnome must now deduce the color of his own hat. All of them announce their guesses simultaneously.

To prevent cheating, the interaction will be split into two phases:

- In the first phase, you will specify for each gnome what number they write on their card, presented in a shuffled order.
- Similarly, the second phase will proceed in random order. See the interaction section for more details.

Interaction Protocol

First, you must read a line containing a single integer T ($1 \le T \le 4 \cdot 10^4$), the number of times a gnome will be asked to write a number **across all tests**.

For each gnome, you must first read a line containing 21 space-separated integers: i — the index of the gnome being asked to write a number and $C_1, C_2, \ldots, C_{20}$ ($1 \le i \le 20$) — the colors of the other gnomes. Note that $C_i = 0$ always (as they can't see their own hat).

Then, you should print one line containing one integer, which is the value that the i-th gnome writes on the card.

Now, phase two begins. First, you must read a line containing a single integer K ($1 \le K \le 4 \cdot 10^4$) denoting the number of times a gnome shares the color of their hat **across all tests**. Then, a description of the interaction for K gnomes follows:

For each gnome, you must first read a line containing 21 space-separated integers and one string consisting of 20 characters: $i, C_1, C_2, \ldots, C_{20}, A$ ($1 \le i \le 20$). Gnome i is going to tell the color of his hat in some test, identified by the colors of the other gnomes' hats, which are given by the array C (same as in phase 1). Note that $C_i = 0$ always. The string A represents the values written on the cards by each gnome.

Then, you should print one line containing one integer, which is the color of the i-th gnome's hat.

After outputting each line, don't forget to flush the output. For example:

- `fflush(stdout)` in C/C++;
- `System.out.flush()` in Java;
- `sys.stdout.flush()` in Python;
- `flush(output)` in Pascal;

- See the documentation for other languages.

Example

standard input	standard output
20	
1 0 2 3 4 ... 5 1 2 3 4 5	
	0
2 1 0 3 4 ... 5 1 2 3 4 5	
	0
3 1 2 0 4 ... 5 1 2 3 4 5	
	1
...	...
20	
1 0 2 ... 4 5 00100000000000000000	
	1
2 1 0 ... 4 5 00100000000000000000	
	2
7 1 2 ... 4 5 00100000000000000000	
	2
...	...

Note

Sample has only one test, and it has the following color configuration:

$C = [1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 1, 2, 3, 4, 5]$

Problem B. Word Generator

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

A machine has M slots numbered from 1 to M, where cards with letters can be placed. Each slot i can only be filled with cards from a specific set of available cards for that slot. When all M slots are filled, the letters in the cards form a word by concatenating the letters from left to right across the slots.

Each card can only be used once to create words.

A word is considered **beautiful** if and only if all of its characters are distinct. For example, the words `abchd`, `a`, and `ab` are beautiful, while the words `abdsa` and `aa` are not.

The task is to determine the maximum number of distinct beautiful words that can be created using the available cards for each slot.

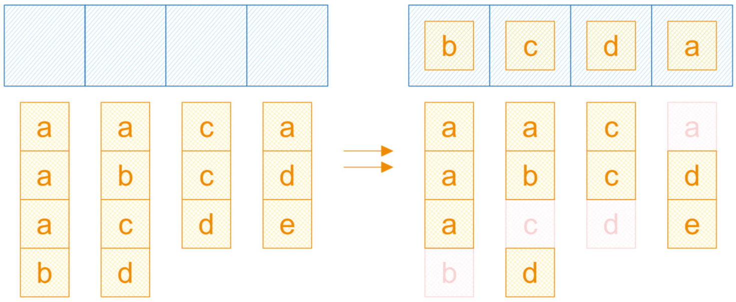

Figure 1. The first test from the sample.

Input

The first line contains one integer T $(1 \le T \le 10^3)$, the number of test cases. Then, for each test case:

- The first line contains one integer M $(1 \le M \le 26)$.
- Then, M lines follow, each containing one string S_i, the set of cards that can be used for slot i. It is guaranteed that for each $c \in S_i$, we have $c \in \{\mathtt{a}, \mathtt{b}, \ldots, \mathtt{z}\}$.

The total sum of $|S_i|$ across all test cases does not exceed 10^6.

Output

For each test:

- First, print one line consisting of one integer C, the maximum number of words that you can create.
- Then, print C lines, each consisting of one string W_i, words which you created.

Example

standard input	standard output
2 4 aaab abcd ccd ade 3 aaabc abbbc abccc	3 bcda adce abcd 5 abc abc abc cab bca

Problem C. Glitch

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

This is an interactive problem!

There's a hidden array $a_1, a_2, \ldots, a_N$ of N distinct positive integers (where N is odd); an incorrect copy of this array is stored in another hidden array b. Each element of array b is defined as follows:

- $b_1 = a_2$
- $b_N = a_{N-1}$
- $b_i = a_{i-1}$ or $b_i = a_{i+1}$ for each i where $1 < i < N$ (one of the adjacent elements)

You need to determine whether there are two indices i and j ($i \neq j$, $1 \leq i, j \leq N$) such that $b_i = b_j$. If such indices exist, return them; otherwise, indicate that no such indices exist.

You can query the elements of arrays a and b using the following operations:

- `1 i`: returns the value of a_i ($1 \leq a_i \leq 10^9$)
- `2 i`: returns the value of b_i ($1 \leq b_i \leq 10^9$)

You can ask up to a maximum of 20 queries.

Interaction Protocol

First, you must read a line containing a single integer T ($1 \leq T \leq 100$) denoting the number of test cases. Then follows a description of the interaction for T test cases.

For each test case, you must first read a line containing one integer N ($3 \leq N \leq 2\,001,\ N$ is odd).

Then interaction starts. You can print one line, containing:

- `1 x` (Ask about a_x): Then, read one line, consisting of one integer, a_x.
- `2 x` (Ask about b_x): Then, read one line, consisting of one integer, b_x.
- `3 i j` (The answer: two indices i, j, where $b_i = b_j$ or `-1 -1` if there are no such two indices): Then, proceed to the next test if there is any; otherwise, terminate.

After outputting each line, don't forget to flush the output. For example:

- `fflush(stdout)` in C/C++;
- `System.out.flush()` in Java;
- `sys.stdout.flush()` in Python;
- `flush(output)` in Pascal;
- See the documentation for other languages.

Example

standard input	standard output
1	
7	
	2 3
2	
	2 1
2	
	3 1 3

Problem D. Electi Lamps

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There are N lamps located along a street at positions X_i, each with a power level P_i. The i-th lamp lights up an area within the range $[X_i - P_i, X_i + P_i]$ when turned on. Initially, some lamps are turned on, and some are turned off. The initial state of the i-th lamp is denoted by T_i: if $T_i = 1$, the lamp is on; if $T_i = 0$, the lamp is off. The positions of the lamps are sorted in increasing order, i.e., $X_i < X_{i+1}$ for all $1 \le i < N$.

For each lamp i ($1 \le i < N$), you need to determine a value $F(i)$, which is the minimum total distance you must walk to achieve the following:

- All lamps before position X_i are considered destroyed and do not exist.
- If the i-th lamp was initially off, it is turned on.
- You start at position X_i and want to walk to position X_N. Along the way, you must turn off every lamp except for the one at position X_N.

If you cannot reach position X_N while ensuring all lamps except lamp N are turned off, then $F(i) = 0$.

The task is to compute the sum of $F(i)$ for all i ($1 \le i < N$), modulo 998244353.

Note: You can only walk in areas illuminated by the lamps.

Input

The first line contains one integer T ($1 \le T \le 10^3$), the number of test cases.

Then, for each test case:

- The first line contains one integer N ($1 \le N \le 10^6$).
- Then, N lines follow. In the i-th of these lines, there are three space-separated integers X_i, P_i, and T_i ($1 \le X_i \le 10^{12}, 1 \le P_i \le 10^{12}, T_i \in \{0, 1\}$).
- It is guaranteed that $X_i < X_{i+1}$ for all $1 \le i < N$.

The total sum of N across all test cases does not exceed 10^6.

Output

For each test, print one line consisting of one integer, the answer.

Example

standard input	standard output
2 3 3 1 0 5 2 0 6 2 1 3 1 8 1 4 7 1 6 1 0	8 0

Note

In the first example:

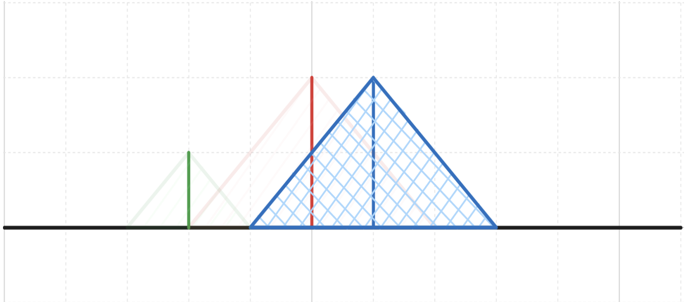

Figure 1. Initial state of Lamps.

Now let's look at what happens when you start at the first lamp.

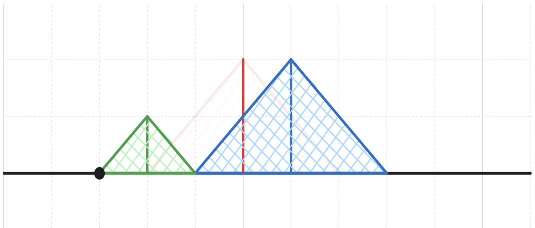

Figure 2. The starting lamp is turned on.

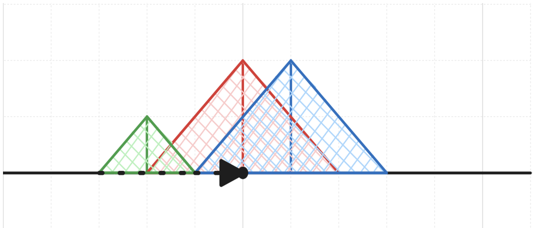

Figure 3. Move to lamp 2 and turn it on.

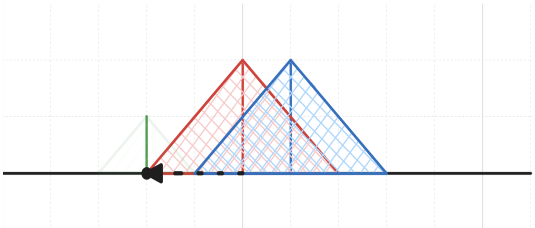

Figure 4. Go back to lamp 1 and turn it off.

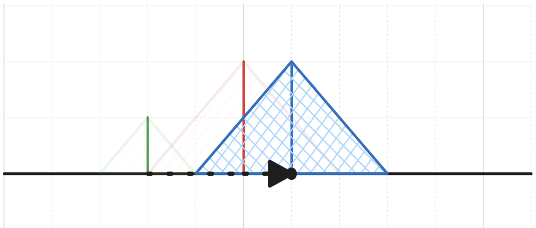

Figure 5. Go to lamp 3 and turn off lamp 2.

Problem E. XOR Again?

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Given an array $A_1, A_2, \ldots A_N$ of N integers. Solve the following problem for every M ($1 \le M \le N$):

- Divide the array into exactly M consecutive blocks.
- The cost of each block is its bitwise XOR. The cost of division is the bitwise OR of its blocks' costs.
- Find the minimum cost of division.

Input

The first line contains one integer N ($1 \le N \le 10^6$). The second line consists of N space-separated integers $A_1, A_2, \ldots, A_N$ ($0 \le A_i \le 10^6$).

Output

Print one line consisting of N integers. The i-th integer is the answer for the problem with $M = i$.

Examples

standard input	standard output
6 0 3 10 2 4 5	10 10 11 11 11 15
4 0 1 0 1	0 0 1 1

This page is intentionally left blank.

Problem F. Binary Permutation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

A permutation is a sequence of length n consisting of integers from 1 to n, in which all the numbers occur exactly once. For example, $[1]$, $[3, 5, 2, 1, 4]$, and $[1, 3, 2]$ are permutations, whereas $[2, 3, 2]$, $[4, 3, 1]$, and $[0]$ are not.

You are given an array $A_1, A_2, \ldots, A_N$ consisting of N integers. Each integer is 0 or 1. Find the number of permutations P of size N such that the following conditions hold:

- $P_1 < P_2 > P_3 < P_4 \ldots$
- $A_{P_i} \equiv i \pmod 2$, for all $(1 \leq i \leq N)$.

Because the number of such permutations can be very large, print the answer modulo 998244353.

Input

The first line contains an integer N $(1 \leq N \leq 10^6)$.

The second line contains N space separated integers A_i - the elements of the array. It's guaranteed that $A_i \in \{0, 1\}$.

Output

Output a single line containing the number of permutations satisfying the conditions modulo 998244353.

Examples

standard input	standard output
2 1 0	1
1 0	0
7 1 1 0 1 0 1 0	8

This page is intentionally left blank.

Problem G. Amoeba Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You're given a tree with N vertices. On this tree, there are M amoebas, each occupying a subset of the vertices. Importantly, the vertices occupied by any given amoeba always form a connected subset.

The goal is to eliminate all the amoebas. This can be achieved by shooting at a vertex, killing any amoeba that has even just a portion of its body within that vertex. More explicitly, if an amoeba is spread across multiple vertices and one of those vertices is targeted, the entire amoeba, regardless of its size or the number of vertices it spans, will be eliminated. You need to determine the minimum amount of ammunition required to eliminate all the amoebas.

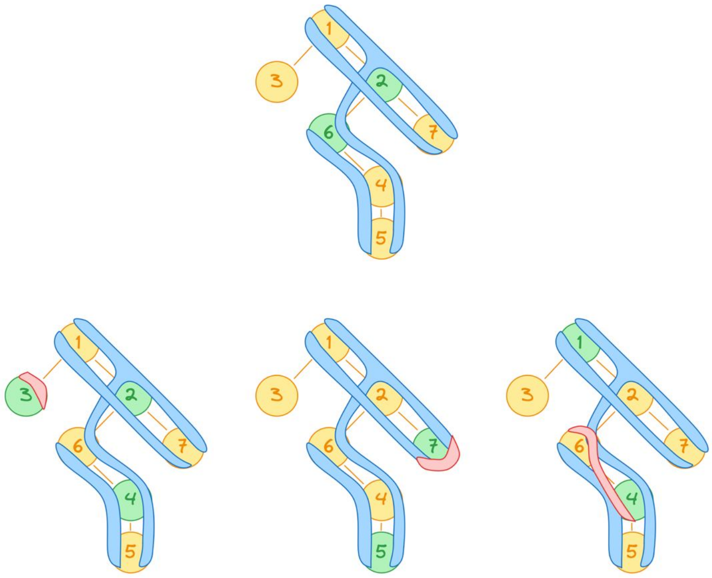

Figure 1. The first test from the sample.

Additionally, Q events will occur. In each event, the scenario is re-evaluated under the assumption that an additional amoeba has appeared. The task is to determine the new minimum amount of ammunition needed.

Note that this new amoeba will be eliminated by the end of the event (i.e., each event is independent).

Input

The first line contains one integer T ($1 \le T \le 10^3$), the number of test cases.

Then, for each test case:

- The first line contains three space-separated integers N, M, and Q ($1 \le N \le 2 \cdot 10^5$, $1 \le M \le 2 \cdot 10^5$, $0 \le Q \le 2 \cdot 10^5$).
- Then, $N - 1$ lines follow, describing the tree. Each line contains two space-separated integers x and y ($1 \le x, y \le N$), indicating an edge between vertex x and vertex y.
- Next, M lines follow, describing the amoebas. Each line starts with an integer k_i followed by k_i space-separated integers $v_{i,1}, v_{i,2}, \ldots, v_{i,k_i}$ ($1 \le k_i, v_{i,j} \le N$), where the vertices $v_{i,1}, v_{i,2}, \ldots, v_{i,k_i}$ form a connected subset from the tree and are all distinct.

- Finally, Q lines follow, describing additional amoebas for each event. Each line starts with an integer l_i followed by l_i space-separated integers $u_{i,1}, u_{i,2}, \ldots, u_{i,l_i}$ $(1 \leq l_i, u_{i,j} \leq N)$, where the vertices $u_{i,1}, u_{i,2}, \ldots, u_{i,l_i}$ form a connected subset from the tree and are all distinct.

It is guaranteed that:

- The total sum of N across all tests does not exceed $2 \cdot 10^5$.
- The total sum of M across all tests does not exceed $2 \cdot 10^5$.
- The total sum of Q across all tests does not exceed $2 \cdot 10^5$.
- The total sum of k_i across all tests does not exceed $2 \cdot 10^5$.
- The total sum of l_i across all tests does not exceed $2 \cdot 10^5$.

Output

For each test, print $Q + 1$ lines. The answer before events, and after each event. *Note that the first line should contain the answer before any events take place!*

Example

standard input	standard output
2 7 3 3 1 2 1 3 2 6 2 7 6 4 4 5 6 1 2 6 7 4 5 3 1 2 7 3 4 5 6 1 3 1 7 2 4 6 1 2 0 1 1 1 1	2 3 2 2 1

Problem H. Magical Puzzles

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 megabytes

Given a tree with N nodes and $N-1$ edges, where each edge i has a complexity level l_i. The distance $d(u,v)$ between two nodes u and v is the sum of the complexity levels along the unique path between them in the tree. Note that $d(u,v) = d(v,u)$ and $d(v,v) = 0$.

There are M puzzles, and each puzzle i requires k_i pieces of information, which are located along specific edges in the tree. It's possible for multiple pieces of information to be on the same edge!

For any two nodes S and T, a journey between S and T is successful if, along the path from S to T, all necessary pieces of information for any puzzle encountered are collected. Specifically, a puzzle is considered solvable during a journey if all required pieces of information for that puzzle are encountered along the path.

The task is to compute the sum of distances for all successful journeys between any two nodes S and T (where $1 \le S \le T \le N$), modulo $10^9 + 7$.

Formally, the task is to calculate:

$$\sum_{\substack{1 \le S \le T \le N, \\ \text{journey } (S,T) \text{ is successful}}} d(S,T)$$

Starting from a node S, you need to travel to another node T along the unique path in the tree. Along the way, any piece of information located on the edges of this path will automatically be collected. For the journey from S to T to be considered successful, it must meet the following condition:

- Let O be the set of puzzles for which at least one piece of information has been collected during the journey. The journey is successful if and only if all the pieces of information required to solve every puzzle in the set O have been collected. In other words, you must have gathered every piece of information needed to completely solve each puzzle in O.

Input

The first line contains two space-separated integers N and M ($2 \le N \le 10^6, 1 \le M \le 10^6$).

Then, $N-1$ lines follow, each containing three space-separated integers a_i, b_i, l_i ($1 \le a_i, b_i \le N, 1 \le l_i \le 10^9$), describing an edge i which connects nodes a_i and b_i with complexity l_i.

Then, the definitions of M puzzles follow. For each puzzle i, the input starts with one line containing k_i followed by k_i space-separated integers $u_{i_1}, u_{i_2}, \ldots, u_{i_{k_i}}$ ($0 \le k_i \le K, 1 \le u_{i_j} \le N-1$), where u_i are the indices of the edges on which the pieces of information required to solve the puzzle are located.

Here, K is the sum of all k_i values across all puzzles, and it is guaranteed that $1 \le K \le 10^6$.

Output

Output one line, containing one integer, the answer for the problem modulo $10^9 + 7$.

Example

standard input	standard output
5 4 1 2 3 1 4 1 1 5 6 4 3 2 2 4 4 1 1 0 2 3 4	17

Note

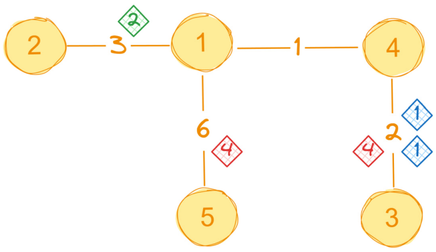

Figure 1. The structure of the tree. The diamonds are the pieces of information, and the values on the edges are the complexities.

Problem I. Bobs Rating

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

Given a graph with N vertices and M unordered edges. Each edge i has a passing rating of W_i and connects vertices U_i and V_i.

Bob is allowed to move from one vertex to another using edges. Bob can use the i-th edge if and only if his rating is greater than or equal to W_i. He can use the same edge and visit the same vertex as many times as he wants.

Whenever Bob is located at vertex x, he has two options for gaining a new rating:

- He can gain A_x rating for free. He can use this option only once during his journey.
- He can gain 1 rating for C_x coins. He can use this option as many times as he wants throughout his journey.

Bob now has Q queries. For each query i, answer the following question:

- Bob has an initial rating of R_i and is located at vertex S_i. He is planning to reach vertex T_i. What is the minimum number of coins required for this?

Print -1 if he cannot reach vertex T_i even with an infinite number of coins.

Input

- The first line contains one integer T $(1 \leq T \leq 100)$, the number of test cases.
- Then, for each test case:
  - The first line contains three space-separated integers N, M, Q $(1 \leq N, M, Q \leq 5 \cdot 10^5)$.
  - The second line contains N space-separated integers $A_1, A_2, \ldots, A_N$ $(1 \leq A_i \leq 10^6)$.
  - The third line contains N space-separated integers $C_1, C_2, \ldots, C_N$ $(1 \leq C_i \leq 10^6)$.
  - Then, M lines follow, each containing three space-separated integers U_i, V_i, W_i $(1 \leq U_i, V_i \leq N; 1 \leq W_i \leq 10^{12})$.
  - Then, Q lines follow, each containing three space-separated integers S_i, T_i, R_i $(1 \leq S_i, T_i \leq N; 1 \leq R_i \leq 10^{12})$.

It is guaranteed that the sum of N across all test cases does not exceed $5 \cdot 10^5$.

It is guaranteed that the sum of M across all test cases does not exceed $5 \cdot 10^5$.

It is guaranteed that the sum of Q across all test cases does not exceed $5 \cdot 10^5$.

Output

For each test case, print Q lines, each consisting of one integer, the answer for that query.

Example

standard input	standard output
1 5 5 4 5 2 4 3 10 1 2 3 4 5 3 3 10 1 2 5 3 4 7 1 3 13 2 5 21 4 5 3 2 2 4 1 4 2 5 2 10	11 0 4 5

Problem J. AND Components

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

There is an array $A_1, A_2, \ldots, A_N$ consisting of N integers, and 2 binary strings L, R. There will be Q updates, where in each update, the value of some element is changed. Before any updates and after each update, print the answer to the following problem:

Let's create an unoriented graph G consisting of N vertices. For each i, we add up to two edges:

- *With the maximum j such that $j < i$, $A_j \ \& \ A_i > 0$, and $L_i = 1$ holds. If no such j exists or $L_i = 0$, then we do not add this edge.*
- *With the minimum j such that $j > i$, $A_j \ \& \ A_i > 0$, and $R_i = 1$ holds. If no such j exists or $R_i = 0$, then we do not add this edge.*

Find the number of connected components in this graph.

Input

- The first line contains one integer T ($1 \le T \le 10^4$), the number of test cases.
- Then, for each test case:
  - The first line contains two space-separated integers N and Q ($1 \le N, Q \le 10^5$).
  - The second line contains N space-separated integers $A_1, A_2, \ldots, A_N$ ($0 \le A_i \le 10^6$).
  - The third line contains a string consisting of N characters $L_1 L_2 \ldots L_N$ ($L_i \in \{0, 1\}$).
  - The fourth line contains a string consisting of N characters $R_1 R_2 \ldots R_N$ ($R_i \in \{0, 1\}$).
  - Then, Q lines follow, each in one of the following formats:
    - `A` i x ($1 \le i \le N, 0 \le x \le 10^6$): Update the value of A_i to x, i.e., $A_i := x$.
    - `L` i x ($1 \le i \le N, x \in \{0, 1\}$): Update the value of L_i to x, i.e., $L_i := x$.
    - `R` i x ($1 \le i \le N, x \in \{0, 1\}$): Update the value of R_i to x, i.e., $R_i := x$.

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

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

Output

For each test, print $Q + 1$ lines, each consisting of one integer, the number of connected components. *Note that the first line is the answer before any updates take place!*

Example

standard input	standard output
2 5 4 1 2 4 1 1 10101 11011 A 1 4 A 1 1 L 2 0 A 5 3 4 4 1 1 1 0 1111 1111 A 2 2 R 1 0 L 4 0 A 4 1	3 3 3 3 2 2 3 3 3 2

Problem K. Hidden Digits

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There's a hidden string of digits (from 0 to 9) of length N ($1 \le N \le 18$). You want to find out whether there's a substring of any size inside this hidden number that is divisible by 3.

You are allowed to ask **up to 2 queries** of the following format: `"? i"`, which tells you the digit at the requested index i ($1 \le i \le N$).

Input

The first line contains the integer N, the length of the hidden string.

Interaction Protocol

You're allowed a maximum of 2 queries of the following format: `? i`

After making a query, you get a single line containing a digit from 0 to 9 indicating the digit at the index i in the hidden string.

To make an answer, output a single line containing `! 1` if there's a substring that's divisible by 3. Otherwise, `! 0`.

Example

standard input	standard output
2	
	? 1
6	
	? 2
8	
	! 1

This page is intentionally left blank.

Problem L. Robots

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 256 megabytes

There are 4 types of robots:

- `A`: Can move only down and right. That is, a robot at point (x, y) can move to point $(x+1, y)$ or $(x, y+1)$.
- `B`: Can move only down and left. That is, a robot at point (x, y) can move to point $(x+1, y)$ or $(x, y-1)$.
- `C`: Can move only up and right. That is, a robot at point (x, y) can move to point $(x-1, y)$ or $(x, y+1)$.
- `D`: Can move only up and left. That is, a robot at point (x, y) can move to point $(x-1, y)$ or $(x, y-1)$.

There are 2 types of cells in the grid:

- `.` (Empty): A robot can be freely placed and move through empty cells.
- `#` (Blocked): A robot cannot move to blocked cells.

Given a value N, construct a grid of size $(N+1) \times (N+1)$ such that each cell is either empty or blocked. Then place N robots of any type you want. Robots should be placed in empty cells, and no two robots should be placed in the same cell.

You are also given a tree, where each non-leaf vertex has at least 4 neighbours. Let E be the set of edges (x_i, y_i). Let S_i be the set of cells to which robot i can move. The following conditions should be satisfied:

- For all pairs $(i, j) \in E$, $S_i \cap S_j \neq \emptyset$ must hold.
- For all pairs $(i, j) \notin E$, $S_i \cap S_j = \emptyset$ must hold.

Input

- The first line contains one integer T $(1 \leq T \leq 10^3)$, the number of tests.
- Then, for each test:
  - The first line contains one integer N $(1 \leq N \leq 50)$.
  - Then, $N-1$ lines follow, each containing two space-separated integers x_i, y_i $(1 \leq x_i, y_i \leq N)$.

Output

For each test, print the answer in the following format:

- First, print $N+1$ lines, each containing $N+1$ characters consisting of `.` or `#`. The j-th character in the i-th line $(1 \leq j \leq N+1, 1 \leq i \leq N+1)$ should represent the type of cell at position (i, j).
- Then, print N lines, each containing two space-separated integers and one character A_i, B_i, C_i $(1 \leq A_i, B_i \leq N+1; C_i \in \{\texttt{A}, \texttt{B}, \texttt{C}, \texttt{D}\})$, where (A_i, B_i) is the cell where the i-th robot is placed, and C_i is the type of robot. *Note that all robots must be placed in different cells.*

Example

standard input	standard output
1 5 1 2 1 3 1 4 1 5	....##... ..#... ...... 6 1 C 1 6 C 1 5 D 2 6 A 2 5 B

Note

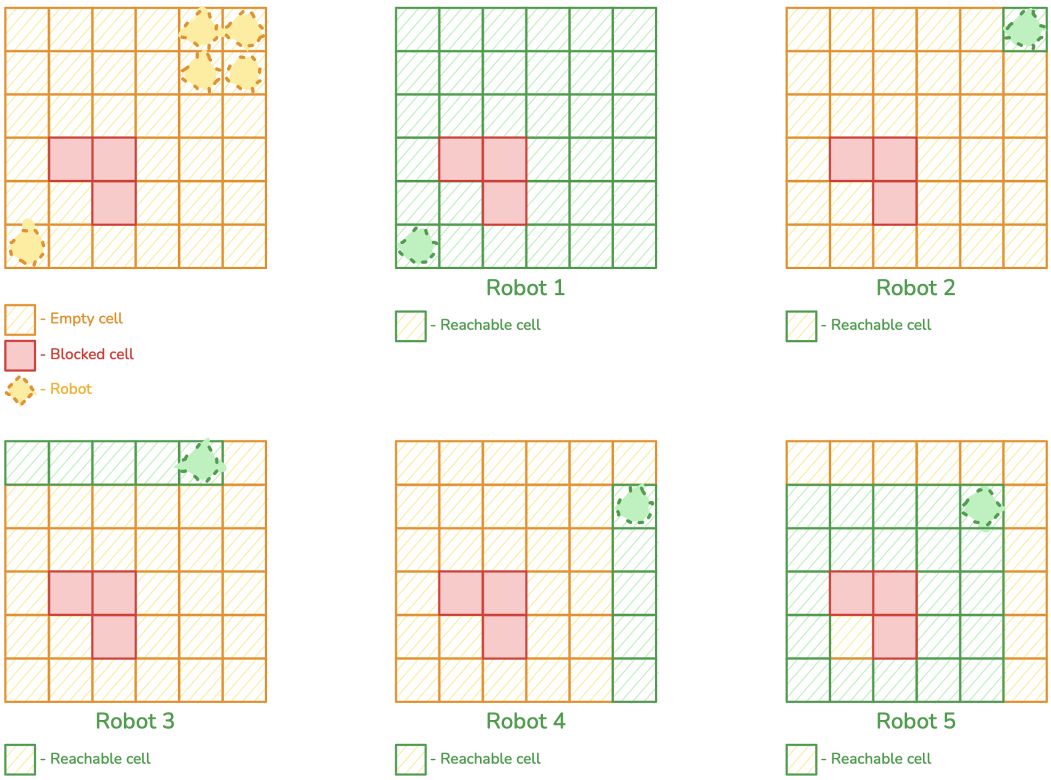

Figure 1. Specification for sample.