Problem A. Puzzle: X-Sums Sudoku

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

An $n \times m$ sudoku puzzle is a grid consisting of $m \times n$ regions, and each region contains $n \times m$ cells. Hence an $n \times m$ sudoku puzzle contains $nm \times nm$ cells. Every integer from 1 to nm occurs exactly once in each row, each column, and each region of an $n \times m$ sudoku puzzle.

Listing the integers in a row or a column starting from some direction as a sequence of length nm, X is the first integer of the sequence, and X-sum is the sum of the first X integers of the sequence.

	1	6	11	20	22	32	34	36	
1	1	2	3	4	5	6	7	8	36
8	3	4	1	2	7	8	5	6	29
27	5	6	7	8	1	2	3	4	10
34	7	8	5	6	3	4	1	2	3
3	2	1	4	3	6	5	8	7	34
10	4	3	2	1	8	7	6	5	27
29	6	5	8	7	2	1	4	3	8
36	8	7	6	5	4	3	2	1	1
	36	34	32	22	20	11	6	1	

The above figure is a 4×2 sudoku puzzle with X-sums. The 7-th row listed from right to left is $[3, 4, 1, 2, 7, 8, 5, 6]$ and the first integer X is 3, so the X-sum of the 7-th row from the direction right is $8 = 3 + 4 + 1$.

Given two positive integers n and m, a direction d, and an index x, you need to find the X-sum of the x-th row or x-th column from the direction d in **the lexicographically smallest** $2^n \times 2^m$ sudoku.

Denoting $a_{i,j}$ as the i-th row and the j-th column of a sudoku puzzle a, a sudoku puzzle a is lexicographically smaller than a sudoku puzzle b of the same size if there exists i and j satisfying that $a_{i,j} < b_{i,j}$, that $a_{x,y} = b_{x,y}$ for all $x < i$, and that $a_{x,y} = b_{x,y}$ for all $x = i$ and $y < j$. You can find that the above is the lexicographically smallest 4×2 sudoku puzzle.

Input

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

For each test case:

The only line contains two integers n and m $(1 \le n, m \le 30)$, a string d, and an integer x $(1 \le x \le 2^{n+m})$. $2^n \times 2^m$ is the size of the sudoku puzzle. d is the direction of X-sum, and it is one of "left", "right", "top", and "bottom". x is the index of a row or a column.

Output

For each test case:

Output an integer – the X-sum of the x-th row or x-th column from the direction d in **the lexicographically smallest** $2^n \times 2^m$ sudoku.

Note that the answer may exceed $2^{64} - 1$. Consider using `__int128_t` in C++, `BigInteger` in Java or Kotlin, or `int` in Python.

Examples

standard input	standard output
4 2 1 top 1 2 1 bottom 2 2 1 left 3 2 1 right 4	1 34 27 3
4 11 19 top 1053766555 12 26 top 230781535210 14 10 right 8344647 7 30 right 70120568170	565741033271081135 31719572400444316026492 112693473538824 477453505821905419941

Problem B. Puzzle: Patrick's Parabox

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	512 megabytes

Patrick's Parabox is a *Sokoban*-like game. A Sokoban puzzle is a gird, and each cell is a wall or a floor. There are several boxes and a player in some distinct floor cells, and they can not move to the wall cells or coincide. You can control the player to move in one of four directions, left, right, up, and down. When the player touches a box, it can push the box. The target is to move all boxes to some target cells.

Please read the following rules carefully. They may be different from the usual rules.

In this problem, there is **only one** box, and the box is the grid itself. That means if the player moves out of the grid, it may be "teleported" to a cell adjacent to the box; if the player moves to the box, it may be "teleported" to a cell on the boundary of the grid. Besides, there is also a target cell for the player. The player needs to move to the target cell at the end too.

Given a puzzle, you need to find the **minimum times to push** the box, such that the box and the player move to the target cell separately.

The following are the detailed and formal rules.

Consider an $n \times m$ grid. Denote (i, j) as the cell in i-th row and j-the column. The rows are numbered 1, 2, $\cdots$, n from top to bottom, and the columns are numbered 1, 2, $\cdots$, m from left to right.

Denote W, S, A, and D as the control commands, which means to move up, down, left, and right separately.

Define that $v_{\mathrm{W}} = (-1, 0)$, $v_{\mathrm{S}} = (1, 0)$, $v_{\mathrm{A}} = (0, -1)$, $v_{\mathrm{D}} = (0, 1)$.

Define that $w_{\mathrm{W}} = (n, \lceil \frac{m}{2} \rceil)$, $w_{\mathrm{S}} = (1, \lceil \frac{m}{2} \rceil)$, $w_{\mathrm{A}} = (\lceil \frac{n}{2} \rceil, m)$, $w_{\mathrm{D}} = (\lceil \frac{n}{2} \rceil, 1)$.

In each operation, you can choose one of the control commands c, one of W, S, A, and D. Denote p as the cell which contains the player and b as the cell which contains the box before the operation:

- If $p + v_c = b$ and $b + v_c$ is a floor cell, the player moves to $p + v_c$ and the box moves to $b + v_c$. **Only this case will be counted in the answer.**
- If $p + v_c$ is a wall cell, nothing happens.
- If $p + v_c$ is a floor cell and $p + v_c \neq b$, the player moves to $p + v_c$.
- If $p + v_c = b$ and $b + v_c$ is outside the grid, nothing happens.
- If $p + v_c = b$, $b + v_c$ is a wall cell and w_c is a wall cell, nothing happens.
- If $p + v_c = b$, $b + v_c$ is a wall cell and w_c is a floor cell, the player moves to w_c.
- if $p + v_c$ is outside the grid and $b + v_c$ is a wall cell, nothing happens.
- if $p + v_c$ is outside the grid and $b + v_c$ is outside the cell, nothing happens.
- if $p + v_c$ is outside the grid and $b + v_c$ is a floor cell, the player moves to $b + v_c$.

Note that the above are listed for covering all possibilities, but the operations are valid in only four of them.

Input

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

For each test case:

The first line contains two integers n and $m (2 \leq n,\ m \leq 10^5)$, the size of the parabox.

Each of the following n lines contains a string of length m. Each character is one of "#", ".", "p", "b", "=" and "-". "#" denotes the wall cell, "p" denotes the floor cell which contains the player, "b" denotes the floor cell which contains the box, "=" denotes the target floor cell of the player, "-" denotes the target floor cell of the box, and "." denotes the other floor cell.

It is guaranteed that each of "p", "b", "=", "-" occurs exactly once.

It is guaranteed that the sum of nm over all test cases does not exceed 4×10^5.

Output

For each test case:

If it is impossible to move the box and the player to the target cells, output -1. Otherwise, output an integer – the **minimum times to push** the box.

Examples

standard input	standard output
3 9 9 ######### #####..-# #..=##.## #.p.##.## ....##.## #...b..## #...##.## #....#### ####.#### 9 9 ######### #.......# #.#####.# #.#=....# ..#....-# ###.p.#.# #.....#b# #.....#.# ####.#### 9 9 ####.#### #....#### #.####.## #.......# #.......# ###.##### #=.b#..## #-..p..## #########	7 4 19
1 2 2 pb -=	-1

Note

The three puzzles in the first example are real levels in the game.

Problem C. Puzzle: Hearthstone

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

Hearthstone is one of the popular video card games. **Please read the following rules carefully. They are different from the usual rules.**

There are n kinds of secret cards numbered 1, 2, $\cdots$, n. There are two types of events about secrets:

- `add`: Add a secret with an unknown number into the hero zone. No two secrets with the same number can be in the hero zone simultaneously.
- `test x y`: Test whether secret x exists. If secret x exists, then $y = 1$ and secret x is removed from the hero zone; otherwise, $y = 0$. Note that whatever y is, secret x does not exist in the hero zone after `testing` x.

An event sequence $E = [e_1, \cdots, e_m]$ is valid if and only if it is able to assign a number from 1 to n for each `add` event and perform the events e_1, e_2, $\cdots$, e_m in order such that:

- no secrets are in the hero zone at the beginning;
- secret x does not exist before the event which `add`s a secret x;
- secret x exists before the event `test x 1`;
- secret x does not exist before the event `test x 0`.

Given q events e_1, e_2, $\cdots$, e_q, you need to maintain an event sequence E. Initially, E is empty. For each $i = 1, 2, \cdots, q$ in order, try to append e_i to the end of E. If E is invalid, remove e_i and report a bug. Otherwise, find the number of the secrets that must exist in the hero zone and the number of the secrets that must not exist in the hero zone after performing the events of E in order.

Input

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

The first line contains two integers n and q ($1 \le n, q \le 10^5$) – the number of kinds of secrets and the number of events.

The i-th line of the following q lines represents e_i and it contains:

- a string "add";
- or a string "test", two integers x and y ($1 \le x \le n$, $0 \le y \le 1$).

It is guaranteed that the sum of n and the sum of q over all test cases do not exceed 10^5.

Output

For each test case:

For each event, if it can be appended, output two integers - the number of the secrets that must exist in the hero zone and the number of the secrets that must not exist in the hero zone; otherwise, output the string "bug".

Examples

standard input	standard output
2 1 8 test 1 0 test 1 1 add test 1 0 test 1 1 add test 1 1 test 1 0 2 10 add add add test 1 1 test 1 1 add add add test 2 1 test 2 1	0 1 bug 1 0 bug 0 1 1 0 0 1 0 1 0 0 2 0 bug 1 1 bug 2 0 bug bug 1 1 bug
1 4 7 add add test 3 0 test 4 0 add test 1 1 test 3 1	0 0 0 0 0 1 2 2 2 0 1 1 1 3

Problem D. Poker Game: Decision

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	256 megabytes

Alice and Bob invent a new game based on texas hold'em. **Please read the following rules carefully as they are different from the usual rules. The background of this problem is exactly the same as problem E.**

There are 13 ranks, which are A, K, Q, J, T, 9, 8, 7, 6, 5, 4, 3, and 2 from high to low. There are 4 suits, which are S, H, C, and D. Every combination of a rank and a suit occurs exactly once, so there are $52(= 13 \times 4)$ cards.

A hand is a set of five cards. Each hand has a rank. There are 10 types of hands. Each type also has a rank. If two hands are of different types, the hand of the type with a higher rank always ranks higher. A hand can be represented as a sequence $(r_1, r_2, r_3, r_4, r_5)$, where r_i is the rank of the i-th card and the order of the five cards depends on the type of the hand. If two hands are of the same type, the hand represented as the lexicographically larger sequence ranks higher, i.e., find the smallest index i such that r_i of two hands are different, the hand with higher r_i ranks higher. If the types and the sequences r of two hands are equal, two hands have the same rank.

The 10 types are given in the following from low rank to high rank. If a hand matches the patterns of multiple types, it belongs to the one with the highest rank of them.

- **Highcard**: Any five cards. The sequence r satisfies that $r_1 > r_2 > r_3 > r_4 > r_5$.
- **Pair**: Two cards with the same rank. The sequence r satisfies that $r_1 = r_2$, $r_3 > r_4 > r_5$.
- **Two pairs**: Two cards with the same rank and another two cards with the same rank. The sequence r satisfies that $r_1 = r_2 > r_3 = r_4$.
- **Three of a kind**: Three cards with the same rank. The sequence r satisfies that $r_1 = r_2 = r_3$, $r_4 > r_5$.
- **Straight**: Five cards with five consecutive ranks. The sequence r satisfies that $r_1 > r_2 > r_3 > r_4 > r_5$. Especially, `A 2 3 4 5` is a straight, and A is regarded as a rank lower than 2 in the situation. Hence `A 2 3 4 5` is the straight with the lowest ranks.
- **Flush**: Five cards with the same suit. The sequence r satisfies that $r_1 > r_2 > r_3 > r_4 > r_5$.
- **Full house**: Three cards with the same rank and another two cards with the same rank. The sequence r satisfies that $r_1 = r_2 = r_3$, $r_4 = r_5$.
- **Four of a kind**: Four cards with the same rank. The sequence r satisfies that $r_1 = r_2 = r_3 = r_4$.
- **Straight flush**: A straight with the same suit. The sequence r satisfies that $r_1 > r_2 > r_3 > r_4 > r_5$. Especially, `A 2 3 4 5` with the same suit is a straight flush, and A is regarded as a rank lower than 2 in the situation. Hence `A 2 3 4 5` with the same suit is the straight flush with the lowest ranks.
- **Royal flush**: Straight flush with the ranks T, J, Q, K, and A. Four different royal flushes are of the same rank.

Two cards are dealt to each of Alice and Bob. Instead of the regular rule, 6 community cards are dealt. Two players take one card from the community cards in turn until each player has five cards, also a hand. **Alice takes first.** The player who has a hand with higher ranks wins. If two hands are with the same rank, there is a draw. **Note that all ten cards are shown to both, and they always choose the optimal strategy.**

The above are the same as problem E. The task is the following.

Given the cards of Alice, the cards of Bob and the 6 community cards, find the winner.

Input

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

The first line contains two strings a_1, a_2 – Alice's hole cards.

The second line contains two strings b_1, b_2 – Bob's hole cards.

The third line contains 6 strings c_1, c_2, c_3, c_4, c_5, c_6 – the community cards.

Each string is of length two. The first character is one of "A", "K", "Q", "J", "T", "9", "8", "7", "6", "5", "4", "3", "2", which represents the rank of a card. The second character is one of "S", "H", "C", "D", which represents the suit of a card.

It is guaranteed that any two of the 10 cards are different.

Output

For each test case:

Output a string, one of "Alice", "Bob", "Draw" – representing that Alice wins, that Bob wins, and that there is a draw separately.

Example

standard input	standard output
9 JC 4H TS 5D JS JH JD 4S 4C 4D JC 4H TS 5D TH TC TD 5S 5H 5C JC 4H TS 5D 4S JS 5S TH TC TD 7C 3C 7H TH 3S 3H 3D 2C 4H 5S 7C 3C 7H TH 2H 4H 5H 6H 8H 9H 7C 3C 7H TH TS 3S 2S 2H 4C 4D 2D KH 4D JC 2S 2H 2C KS KC KD 2D KH 4D JC 4S 4H 4C JS JH JD 2D KH 4D JC 2S KS 4S JS JH JD	Alice Bob Draw Alice Bob Draw Alice Bob Draw

Problem E. Poker Game: Construction

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

Alice and Bob invent a new game based on texas hold'em. **Please read the following rules carefully as they are different from the usual rules. The background of this problem is exactly the same as problem D.**

There are 13 ranks, which are A, K, Q, J, T, 9, 8, 7, 6, 5, 4, 3, and 2 from high to low. There are 4 suits, which are S, H, C, and D. Every combination of a rank and a suit occurs exactly once, so there are $52(= 13 \times 4)$ cards.

A hand is a set of five cards. Each hand has a rank. There are 10 types of hands. Each type also has a rank. If two hands are of different types, the hand of the type with a higher rank always ranks higher. A hand can be represented as a sequence $(r_1, r_2, r_3, r_4, r_5)$, where r_i is the rank of the i-th card and the order of the five cards depends on the type of the hand. If two hands are of the same type, the hand represented as the lexicographically larger sequence ranks higher, i.e., find the smallest index i such that r_i of two hands are different, the hand with higher r_i ranks higher. If the types and the sequences r of two hands are equal, two hands have the same rank.

The 10 types are given in the following from low rank to high rank. If a hand matches the patterns of multiple types, it belongs to the one with the highest rank of them.

- **Highcard**: Any five cards. The sequence r satisfies that $r_1 > r_2 > r_3 > r_4 > r_5$.
- **Pair**: Two cards with the same rank. The sequence r satisfies that $r_1 = r_2$, $r_3 > r_4 > r_5$.
- **Two pairs**: Two cards with the same rank and another two cards with the same rank. The sequence r satisfies that $r_1 = r_2 > r_3 = r_4$.
- **Three of a kind**: Three cards with the same rank. The sequence r satisfies that $r_1 = r_2 = r_3$, $r_4 > r_5$.
- **Straight**: Five cards with five consecutive ranks. The sequence r satisfies that $r_1 > r_2 > r_3 > r_4 > r_5$. Especially, `A 2 3 4 5` is a straight, and A is regarded as a rank lower than 2 in the situation. Hence `A 2 3 4 5` is the straight with the lowest ranks.
- **Flush**: Five cards with the same suit. The sequence r satisfies that $r_1 > r_2 > r_3 > r_4 > r_5$.
- **Full house**: Three cards with the same rank and another two cards with the same rank. The sequence r satisfies that $r_1 = r_2 = r_3$, $r_4 = r_5$.
- **Four of a kind**: Four cards with the same rank. The sequence r satisfies that $r_1 = r_2 = r_3 = r_4$.
- **Straight flush**: A straight with the same suit. The sequence r satisfies that $r_1 > r_2 > r_3 > r_4 > r_5$. Especially, `A 2 3 4 5` with the same suit is a straight flush, and A is regarded as a rank lower than 2 in the situation. Hence `A 2 3 4 5` with the same suit is the straight flush with the lowest ranks.
- **Royal flush**: Straight flush with the ranks T, J, Q, K, and A. Four different royal flushes are of the same rank.

Two cards are dealt to each of Alice and Bob. Instead of the regular rule, 6 community cards are dealt. Two players take one card from the community cards in turn until each player has five cards, also a hand. **Alice takes first.** The player who has a hand with higher ranks wins. If two hands are with the same rank, there is a draw. **Note that all ten cards are shown to both, and they always choose the optimal strategy.**

The above are the same as problem D. The task is the following.

Given the cards of Alice and the cards Bob, find possible 6 community cards such that Alice wins, that Bob wins, and that there is a draw separately.

Input

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

The first line contains two strings a_1, a_2 – Alice's hole cards.

The second line contains two strings b_1, b_2 – Bob's hole cards.

Each string is of length two. The first character is one of "A", "K", "Q", "J", "T", "9", "8", "7", "6", "5", "4", "3", "2", which represents the rank of a card. The second character is one of "S", "H", "C", "D", which represents the suit of a card.

It is guaranteed that any two of the four cards are different.

Output

For each test case:

For each of that Alice wins, that Bob wins and that there is a draw, if it is possible, output "YES" and 6 strings representing the 6 cards; otherwise, output "NO".

The format of the cards in the output should be the same as the format in the input.

Any two of the ten cards in input and output must be different.

Examples

standard input	standard output
3 JC 4H TS 5D 7C 3C 7H TH 2D KH 4D JC	YES JS JH JD 4S 4C 4D YES TH TC TD 5S 5H 5C YES 4S JS 5S TH TC TD YES 3S 3H 3D 2C 4H 5S YES 2H 4H 5H 6H 8H 9H YES TS 3S 2S 2H 4C 4D YES 2S 2H 2C KS KC KD YES 4S 4H 4C JS JH JD YES 2S KS 4S JS JH JD
2 AS AH AC AD AS AH 2S 2H	YES 2H 3S 4D 5C 6H 7S NO YES 2C 2D 3S 3H 4C 4D YES AC AD 3D 4C 5H 6S YES 2C 2D 3D 4C 5H 6S NO

Problem F. Longest Common Subsequence

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

Given sequence s of length n and sequence t of length m, find the length of the longest common subsequence of s and t.

Input

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

For each test case:

The only line contains 7 integers, n, m, p, x, a, b, c ($1 \le n, m \le 10^6$, $0 \le x, a, b, c < p \le 10^9$). n is the length of s, m is the length of t.

To avoid large input, you should generate the sequences as follows:

For each $i = 1, 2, \cdots, n$ in order, update x to $(ax^2 + bx + c) \bmod p$, and then set s_i to x. And then, for each $i = 1, 2, \cdots, m$ in order, update x to $(ax^2 + bx + c) \bmod p$, and then set t_i to x.

It is guaranteed that the sum of n and the sum of m over all test cases does not exceed 10^6.

Output

For each test case:

Output an integer – the length of the longest common subsequence of s and t, in one line.

Example

standard input	standard output
2 4 3 1024 1 1 1 1 3 4 1024 0 0 0 0	0 3

Note

In the first sample, $s = [3, 13, 183, 905]$ and $t = [731, 565, 303]$.

In the second sample, $s = [0, 0, 0]$ and $t = [0, 0, 0, 0]$.

Problem G. Lexicographic Comparison

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

a and p are permutations of length n. Initially, $a_i = p_i = i$ for all $1 \le i \le n$. A is a sequence of permutations such that $A_1 = a$ and $A_{i,j} = A_{i-1,p_j}$ for all $i \ge 1$ and $1 \le j \le n$.

There are three types of operations, where x and y are positive integers:

1. `swap_a x y`: swap a_x and a_y, where $1 \le x, y \le n$;
2. `swap_p x y`: swap p_x and p_y, where $1 \le x, y \le n$;
3. `cmp x y`: compare A_x with A_y lexicographically.

For each operation 3, output the relationship between A_x and A_y. A permutation s is lexicographically smaller than a permutation t if and only if there exists an index i such that $s_i < t_i$ and $s_j = t_j$ for all $1 \le j < i$.

Input

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

The first line contains an integer n and q ($1 \le n, q \le 10^5$) - the length of the permutations and the number of operations.

Each of the following q lines contains one string f and two integers x and y representing an operation. f is one of "swap_a", "swap_p" and "cmp". If f is one of "swap_a" and "swap_p", $1 \le x, y \le n$. If f is "cmp", $1 \le x, y \le 10^{18}$.

It is guaranteed that the sum of n and the sum of q over all tests do not exceed 10^5.

Output

For each test case:

For each query, output "<" if A_x is lexicographically smaller than A_y; output ">" if A_x is lexicographically greater than A_y (i.e., A_y is lexicographically smaller than A_x); output "=" if $A_x = A_y$.

Example

standard input	standard output
2 5 5 cmp 1 2 swap_p 1 2 cmp 1 2 swap_a 1 2 cmp 1 2 1 1 swap_a 1 1	= < >

Problem H. Expression Evaluation

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

Expression evaluation is a classic problem. In this problem, you only need to evaluate an expression of length less than 10^5. It only contains non-negative integers (possibly with leading zeros), '+'s, '-'s and '*'s, i.e., it satisfies the following grammar:

```
<expression>:=  <term>|<expression>+<term>|<expression>-<term>
      <term>:=  <number>|<term>*<number>
    <number>:=  <digit>|<number><digit>
     <digit>:=  0|1|2|3|4|5|6|7|8|9
```

For example, `013`, `0213-2132*0213` are valid, but `-2132` and `32113+-3213` are invalid.

The result of the evaluation may be too large or negative. Output the result module 2^{32} to avoid overflow since you will use a 32-bit machine.

The 32-bit machine contains 2^{10} units of memory, denoted as $r[0]$, $r[1]$, $\cdots$, $r[2^{10}-1]$. Each unit is a 32-bit unsigned number, also an instruction.

In each cycle, let $pc = r[0] \bmod 2^{10}$, the machine execute the instruction $r[pc]$. Let $r[pc] = a2^{30} + b2^{20} + c2^{10} + d$ at the beginning of cycle, where a, b, c, d are non-negative integers and less than 2^{10}:

- If $a = 0$, the machine outputs the value of $r[b]$ and stops.
- If $a = 1$, and then:
  - If there are no characters in input left, set $r[0]$ to $r[d]$;
  - Otherwise, set $r[b]$ to the ASCII code of the next character of input, and then set $r[0]$ to $r[c]$.
- If $a = 2$, set $r[b]$ to $(r[b] + r[c]) \bmod 2^{32}$, and then set $r[0]$ to $r[d]$.
- If $a = 3$, and then:
  - If $r[b] = 0$, set $r[0]$ to the value of $r[c]$;
  - Otherwise, set $r[0]$ to the value of $r[d]$.

Note that if $b = 0$ in some instructions, $r[0]$ may be set more than once. Its value is the value set last after the cycle.

You need to set the initial value for each unit such that the machine can stop in finite cycles and output the result of expression module 2^{32}.

Since there is a time limit for a problem. In this problem, the machine can execute at most 10^8 cycles.

Input

The only line contains a 32-bit unsigned integer - the seed for the expression generator.

You do not need to use it. It is just for the checker to generate the expressions.

Output

Output 2^{10} 32-bit unsigned integers in the only line - the initial values of the memory units.

Example

standard input	standard output
0	0 0 ... 0 (1024 zeros)

Note

The output in the example is wrong, just for explaining the output format.

Problem I. Equivalence in Connectivity

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

Two undirected graphs of size n are equivalent in connectivity when there is a path from u to v in one graph if and only if there is a path from u to v in the other graph for all $1 \le u < v \le n$.

Given k graphs G_1, G_2, $\cdots$, G_k of size n, for each $i = 2, 3, \cdots, k$, there exists $p_i < i$ such that G_i can be derived from G_{p_i} by adding or removing an edge. Divide them into groups, such that two graphs are in the same groups if and only if they are equivalent in connectivity.

Input

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

The first line contains three integers k, n and m ($1 \le k, n \le 10^5$, $0 \le m \le \min(10^5, \frac{n(n-1)}{2})$) - the number of graphs, the number of vertices of each graph, and the number of edges of G_1.

Each of the following m lines contains two integers u and v ($1 \le u < v \le n$), denoting an edge of G_1 connecting u and v. It is guaranteed that there are no multiple edges in G_1.

The i-th of following $k-1$ lines contains an integer p_{i+1}, a string t_{i+1}, and two integers x_{i+1} and y_{i+1} ($1 \le p_{i+1} \le i$, $1 \le x_{i+1} < y_{i+1} \le n$). t_i is one of "add" and "remove".

If t_{i+1} ="add", G_{i+1} is derived from $G_{p_{i+1}}$ by adding an edge connecting x_{i+1} and y_{i+1}. It is guaranteed that there does not exist the edge in $G_{p_{i+1}}$.

If t_{i+1} ="remove", G_i is derived from $G_{p_{i+1}}$ by removing an edge connecting x_{i+1} and y_{i+1}. It is guaranteed that there exists the edge in $G_{p_{i+1}}$.

It is guaranteed that the sum of n, the sum of m, and the sum of k in all test cases do not exceed 10^5.

Output

For each test case:

Output an integer r - the number of the groups in the first line.

For each group, output an integer k and k integers - the size of the group and the numbers of graphs in one line.

You can output the groups and the graphs in any order.

Example

standard input	standard output
2 15 11 8 6 11 1 6 6 9 6 8 1 2 1 5 9 10 2 5 1 add 3 11 1 add 2 3 3 add 5 8 4 add 5 11 3 add 7 10 1 add 6 10 3 add 3 10 1 remove 6 8 5 add 4 9 1 add 2 9 8 add 7 8 3 add 2 4 1 remove 6 9 10 remove 6 9 14 5 2 1 5 1 4 1 add 2 4 1 add 3 4 1 add 2 4 4 add 3 4 4 add 1 3 5 add 1 3 2 add 2 3 1 add 1 2 4 add 3 4 3 add 4 5 9 add 2 3 3 remove 1 5 3 remove 3 4	7 2 10 13 5 2 3 4 5 8 3 1 7 11 1 14 2 6 12 1 9 1 15 5 3 2 4 9 6 5 6 7 8 10 12 2 1 14 2 3 11 1 13

Problem J. Symmetry: Tree

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

Given a tree with n vertices, find an integer point $p_i = (x_i, y_i)$ for each node i ($i = 1, 2, \cdots, n$) and connect the p_u and p_v with a line segment for each edge (u, v), such that:

1. No two points coincide.

2. No two line segments have common points except at both endpoints.

3. There exists a line such that the shape formed by the points is symmetric about the line and the shape formed by the line segments is symmetric about the line.

Input

There are multiple test cases. The first line of input contains an integer $T(1 \le T \le 10^3)$, the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^3$), the number of vertices of the tree.

Each of the following $n - 1$ lines contains two integers u and $v(1 \le u, v \le n, u \ne v)$, denoting an edge connecting u and v.

Note that there are **no constraints** related to the sum of n.

Output

For each test case:

If there is no answer, output the "NO" in the only line.

Otherwise, output "YES" in the first line, and two integers x_i, y_i ($0 \le |x_i|, |y_i| \le n$) in the i-th of the following n lines.

In the $n+2$ lines, output three integers a, b, c ($0 \le |a|,|b|,|c| \le n$), denoting that the shapes are symmetric about the $ax + by + c = 0$.

If there are multiple answers, output any of them.

Example

standard input	standard output
5 4 3 2 1 3 4 1 4 2 4 1 4 3 4 9 9 7 4 9 8 4 4 6 1 8 2 6 5 1 3 4 10 5 3 4 5 6 4 2 5 5 8 4 9 7 8 1 2 10 6 7 2 7 7 4 7 5 6 2 4 3 2 1	YES 1 0 -2 0 -1 0 2 0 1 0 0 YES 1 0 0 1 -1 0 0 0 1 0 0 YES 0 3 -2 0 0 0 0 1 0 4 -1 0 2 0 0 2 1 0 1 0 0 NO NO

Problem K. Symmetry: Convex

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

Given a strictly convex polygon with n vertices p_1, p_2, $\cdots$, p_n in counterclockwise. Denote C_i as the polygon with i vertices p_1, p_2, $\cdots$, p_i. For each $i = 3, 4, \cdots, n$, find the lines which C_i is symmetric about.

Input

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

The first line contains an integer n $(3 \le n \le 3 \times 10^5)$, the number of vertices.

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

It is guaranteed that the vertices are given counterclockwise, and the polygon is strictly convex, i.e., no three vertices are colinear.

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

Output

For each test case: s For each $i = 3, 4, \cdots, n$, output an integer k – the number of lines which C_i is symmetric about in the first line, and k lines follow.

In each of the following k lines, output three integers a, b, c $(-2 \times 10^{18} \le a, b, c \le 2 \times 10^{18})$, denoting that C_i is symmetric about the line $ax + by + c = 0$.

If there are multiple answers, you can output any of them. For each i, you can output the lines in any order.

Example

standard input	standard output
3 4 0 0 1 0 1 1 0 1 3 0 0 3 0 1 1 4 -1000000000 -1000000000 1000000000 -1000000000 1000000000 1000000000 -1000000000 1000000000	1 1 1 -1 4 1 -1 0 0 2 -1 2 0 -1 1 1 -1 0 1 1 1 0 4 1 -1 0 0 1 0 1 0 0 1 1 0

Problem L. Symmetry: Closure

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

A point set S is symmetric about a line l if and only if there exists $s' \in S$ satisfying that s' and s are symmetric about the line l for all $s \in S$.

Denoting the distance between two points a and b as $d(a, b)$. The distance between two non-empty point sets A and B is $\inf\{d(a, b) : a \in A, b \in B\}$. The infimum of a non-empty real number set S is the maximum value of x which satisfies $x \leq s$ for all $s \in S$.

n lines $l_1, l_2, \cdots, l_n$ are given, where two lines may coincide. For a point s, define $C(s)$ as the intersection of all sets S satisfying $s \in S$ and S is symmetric about l_i for all $i = 1, 2, \cdots, n$. There are q queries. For each query, given two points A and B, find the distance between $C(A)$ and $C(B)$.

Input

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

The first line contains an integer n and q ($1 \leq n, q \leq 10^5$) – the number of lines and the number of points.

The i-th of the following n lines contains four integers x_{P_i}, y_{P_i}, x_{Q_i} and y_{Q_i} – the coordinates of P_i and Q_i such that l_i passes through P_i and Q_i. It is guaranteed that $x_{P_i} \neq x_{Q_i}$ or $y_{P_i} \neq y_{Q_i}$. Two lines may coincide.

The i-th of the following q lines contains four integers x_{A_i}, y_{A_i}, x_{B_i} and y_{B_i} – the coordinates of A_i and B_i.

It is guaranteed that the absolute value of all coordinates in the input does not exceed 10^9.

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

Output

For each test case:

For each query, output the distance between $C(A)$ and $C(B)$.

The distance you output is correct if the relative error or absolute error to the jury does not exceed 10^{-9}.

Examples

standard input	standard output
4 1 1 0 0 1 0 -1 -2 2 1 2 1 0 0 1 0 0 0 0 1 -1 -2 2 1 3 1 0 0 1 0 0 0 0 1 0 0 1 1 -1 -2 2 1 3 1 0 0 1 0 0 0 0 1 0 0 1 2 -1 -2 2 1	3.162277660168 1.414213562373 0.000000000000 0.000000000000
5 1 1 -8 1 -8 10 -7 -5 -4 -6 2 2 -1 -10 -1 -8 10 9 9 10 2 10 -10 5 -4 4 -3 -3 3 1 -5 -10 -5 6 6 10 8 8 7 -2 4 -5 0 -9 -6 -3 3 3 9 8 10 7 1 5 -9 5 4 -2 -3 -9 6 6 -6 -8 2 -7 10 -3 3 -8 8 -9 1 3 10 -9 10 -7 -2 -7 -2 6 -2 9 -9 2 -6 -7 -7 -9	3.162277660168 7.810249675907 7.071067811865 7.211102550928 0.000000000000 0.000000000000 0.000000000000 13.000000000000 9.899494936612 2.236067977500