# Problem A. Ladder

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Memory limit: | 64 mebibytes |

You have got $n$ sticks of lengths $d_1, d_2, \ldots, d_n$. To build a ladder, you need $k + 2$ sticks: two sticks of length $x$ and $k$ sticks of length $y$. You may shorten the sticks you already have, but you cannot divide one stick into two. Can you make a ladder?

## Input

The first line of the input contains a single integer $z$, the number of test cases. The descriptions of the test cases follow.

Each test case consists of four integers $n$, $k$, $x$, $y$ ($1 \leq n \leq 10^5$, $0 \leq k \leq 10^5$, $1 \leq x, y \leq 10^9$) followed by $n$ integers $d_1, d_2, \ldots, d_n$ ($1 \leq d_i \leq 10^9$).

## Output

For each test case, output a single line containing a single word "YES" if making a ladder is possible, or "NO" otherwise.

## Example

| standard input | standard output |
|---|---|
| 2<br>8 3 5 2<br>1 1 1 2 3 4 5 6<br>8 3 6 2<br>1 1 1 2 3 4 5 6 | YES<br>NO |

# Problem B. 36 Puzzle

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Memory limit: | 64 mebibytes |

A *36-puzzle* consists of 36 square tiles. These squares form one $6 \times 6$ square. Each of the tiles is labeled with either a letter of the Latin alphabet or an Arabic digit. The objective of the game is to place the squares in the following order:

```
abcdef
ghijkl
mnopqr
stuvwx
yz0123
456789
```

This must be achieved by performing a sequence of moves. In each move, all squares in one row or column can be shifted cyclically. For example, in the above position, after shifting the first row by two positions to the right, one gets:

```
efabcd
ghijkl
mnopqr
stuvwx
yz0123
456789
```

Given some order of the squares, find any sequence of moves that achieves the desired order.

## Input

The input consists of six lines. Each line contains six characters: lowercase letters or digits. The $i$-th line describes labels on squares in the $i$-th row. Each possible label will appear exactly once. You can assume that for the test data, there will be always exist a solution.

## Output

Print the descriptions of moves. A description of a single move consists of the row or column number $i$ ($1 \leq i \leq 6$), the direction of the shift $d$ ($d$ should be "L" for shifting a row to the left, "R" for shifting a row to the right, "U" for shifting a column upwards or "D" for shifting a column downwards) and $k$, the number of squares to shift by ($1 \leq k \leq 5$). Consecutive moves must be printed on separate lines.

## Example

| standard input | standard output |
|---|---|
| bciefa<br>ghojkl<br>mnupqr<br>st0vwx<br>yz6123<br>45d789 | 3 D 1<br>1 R 1 |

# Problem C. Message

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Memory limit: | 128 mebibytes |

Agent John knows some top secret information, encoded as the number $N$. In order to avoid revealing information to a third party, he wrote the digits of $N$ in base $b$ in the ancient language of ancient Byteotians. (Each digit in this language is written as a sequence of lowercase Latin letters.) He separated these digits with zero or more random letters (these random letters do not appear inside digit words, only at the beginning, at the end or between two consecutive digits). It is known that Byteotians didn't use zero, so there is no word for it in their language. Fortunately, no zero appears in the notation of this number.

Unfortunately, a message encoded this way can be ambiguous.

You are given the encoded message. Find the greatest number $N$ that could be encoded this way.

## Input

The first line of input contains the number of test cases $T$. The description of the test cases follows.

The description of each test case starts with a line containing the base of the system $b$ ($3 \le b \le 5 \cdot 10^4$). Then follow $(b-1)$ lines, each contains a non-empty string of lowercase letters. The $i$-th of them is the word for digit $i$ in the language of ancient Byteotians. The words are nonempty and pairwise different. The total sum of lengths of these words is not greater than $5 \cdot 10^5$.

The last line of the test case description contains agent John's encoded message. Its length is not greater than $3 \cdot 10^5$.

## Output

Print the answers to the test cases in the order in which they appear in the input. For each test case, print a single line containing the digits of the number separated by spaces. If no number matches the encrypted message, print a single number 0 on this line.

## Example

| standard input | standard output |
|---|---|
| 1<br>10<br>one<br>two<br>three<br>four<br>five<br>six<br>seven<br>eight<br>nine<br>twohundredsixtynine | 2 6 9 |

# Problem D. Arithmetic Sequences

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Memory limit: | 512 mebibytes |

An arithmetic sequence is a sequence of numbers such that the difference between consecutive elements is constant. For example, the sequence $5, 7, 9, 11, 13$ is an arithmetic sequence (the common difference is 2), but the sequence $1, 2, 4, 5$ is not (the differences between consecutive elements are 1, 2 and 1).

Given the set of integers $\{a_1, a_2, \ldots, a_n\}$, find the size of its largest subset that forms an arithmetic sequence.

The set $A$ is said to form an arithmetic sequence if there exists an ordering of $A$ that is an arithmetic sequence.

## Input

The first line of input contains a single integer $z$, the number of test cases. The descriptions of the test cases follow.

Each test case consists of a separate line containing an integer $n$ ($1 \leq n \leq 2000$) followed by $n$ integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i \leq 10^9$). The numbers $a_i$ are pairwise distinct.

## Output

For each test case, output a single line containing the size of the largest arithmetic sequence found in the given set.

## Example

| standard input | standard output |
|---|---|
| 2<br>4 1 2 3 4<br>6 0 1 2 4 5 6 | 4<br>4 |

# Problem E. Subsequence

Input file: *standard input*
Output file: *standard output*
Memory limit: 512 mebibytes

A nondecreasing subsequence $(b_1, b_2, \ldots, b_k)$ of a sequence $(a_1, a_2, \ldots, a_n)$ is said to be a *maximal nondecreasing subsequence* of $a$ if there is no nondecreasing subsequence $(c_1, c_2, \ldots, c_l)$ of $a$ such that $b$ is a subsequence of $c$ and $k < l$.

Given the sequence $(a_1, a_2, \ldots, a_n)$, find the length of its shortest maximal nondecreasing subsequence.

## Input

The first line of input contains the number of test cases $T$. The descriptions of the test cases follow.

The description of each test case starts with a line containing one integer $n$ ($1 \le n \le 10^6$). The second line contains $n$ integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$).

## Output

For each test case, print one integer: the length of the shortest maximal nondecreasing subsequence of the given sequence $a$.

## Example

| standard input | standard output |
|---|---|
| 1<br>5<br>1 2 8 4 9 | 4 |

# Problem F. Algebra is Awesome

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Memory limit: | 512 mebibytes |

Every permutation $\sigma$ can be composed with itself, which means $\sigma^2 = \sigma \circ \sigma$. More generally, for positive $k$, $\sigma^k = \sigma \circ \sigma^{k-1}$ and $\sigma^0$ is an identity permutation. For a permutation $\sigma$, the set of all its compositions is called $D(\sigma)$, which means $D(\sigma) = \{\sigma^k : k \in \mathbf{N}\}$.

You are given an $m$-element sequence of $n$-element permutations $\sigma_1, \sigma_2, \ldots, \sigma_m$. For each $i$, find the number of $j < i$ such that $D(\sigma_i) = D(\sigma_j)$.

## Input

The first line of input contains a single integer $z$, the number of test cases. The descriptions of the test cases follow.

The first line of each test case consists of two integers $n$ and $m$ ($1 \le n \le 10^2$, $1 \le m \le 10^4$).

In each of the next $m$ lines, you are given a pemutation as a sequence of $n$ positive distinct integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$).

## Output

For each test case, print $m$ numbers each on a separate line: how many different $j$'s satisfy the given condition.

## Example

| standard input | standard output |
|---|---|
| 1<br>3 3<br>2 3 1<br>3 1 2<br>1 2 3 | 0<br>1<br>0 |

# Problem G. Bus Lines

Input file: *standard input*
Output file: *standard output*
Memory limit: 64 mebibytes

There are $(n-1)$ bidirectional streets connecting $n$ crossroads (numbered 1 to $n$) in Byteozavodsk. There is only one way to travel between every pair of crossroads without going through any street more than once.

You are tasked with planning city's bus line routes. A bus line route starts and ends at a crossroad such that there is only one street that is incident with this crossroad. The starting crossroad must be different from the ending one.

Every street has a capacity: the number of bus lines that can pass through this street.

Calculate the maximum possible number of bus lines that can be arranged in Byteozavodsk. Not that it makes any sense, but there can be two bus lines that are connecting exactly the same pair of crossroads.

## Input

The first line contains one integer $z$, the number of test cases. Then $z$ test cases are described in the following way.

The first line of each test case description contains one integer $n$ ($2 \leq n \leq 10^5$), the number of crossroads in Byteozavodsk. Each of the following $(n-1)$ lines describing city streets contains three integers $a$, $b$, $c$ ($1 \leq a, b \leq n$, $a \neq b$, $1 \leq c \leq 10^6$) meaning that a particular street connects crossroad $a$ with crossroad $b$, and has a capacity of $c$.

## Output

For each test case, output a single line containing one integer: the maximum number of bus lines that can be arranged in Byteozavodsk.

## Example

| standard input | standard output |
|---|---|
| 3<br>4<br>1 2 2<br>1 3 2<br>1 4 2<br>2<br>1 2 2<br>3<br>1 2 2<br>2 3 1 | 3<br>2<br>1 |

# Problem H. Coal Mine

Input file: *standard input*
Output file: *standard output*
Memory limit: 64 mebibytes

There was a strike at a coal mine in Bytesia recently. Angry miners destroyed all the mine's data. You were hired to recover some data on what types of coal were mined in different parts of the mine.

The company mines $k$ different coal types. The coal mine is a rectangular grid. Every unit square of the grid is an area that mines exactly one type of coal. On some midpoints of edges of the unit squares, there are elevators, one for each of the $k$ coal types being mined in the mine. As a coal mine must be a work of art, the set of unit squares where coal of type $i$ is mined at is symmetric with respect to the point where the elevator assigned to coal type $i$ is located.

You are given the dimensions of the coal mine, the number $k$ and the locations of the $k$ elevators. Your task is to determine a way of assigning unit squares to coal types so the above conditions are met, or to determine that such an assignment does not exist.

## Input

The first line of input consists of one integer $z$, the number of test cases. Then, $z$ test cases proceed, described in the following way.

The first line of each test case description consists of three integers $n$, $m$ amd $k$ ($1 \le n, m$, $n \cdot m \le 1\,500$, $1 \le k \le 100$): the dimensions of the coal mine and the number of different coal types mined in the mine. The upper-left corner of the upper-left field of the mine has coordinates $(0, 0)$, and the lower-right corner of the lower-right field has coordinates $(n, m)$.

Then the locations of $k$ elevators follow, each one being represented as a line containing the coordinates of the elevator $i$, $j$ ($0 \le i \le n$, $0 \le j \le m$). Exactly one of $i, j$ is an integer and the other one is a half of an integer, representing a point on a grid's unit square edge. The $l$-th (1-based) of the elevators is assigned to coal type $l$.

## Output

For each test case, you must first output a single line containg a word "YES" if there exists a proper assignment of coal types to **all** grid's unit squares, or "NO" if such an assignment does not exist. Additionally, if such an assignment exists, you must then output $n$ lines, each containg $m$ integers separated by spaces. The $j$-th number in the $i$-th (1-based) line must be an integer describing the coal type mined at square with upper-left coordinate of $(i-1, j-1)$ and lower-right of $(i, j)$. Coal types are denoted by integers from 1 to $k$. If there are several possible answers, print any one of them.

## Example

| standard input | standard output |
|---|---|
| 2<br>1 6 2<br>0.5 1<br>0.5 4<br>1 4 2<br>0.5 2<br>0.5 3 | YES<br>1 1 2 2 2 2<br>YES<br>1 1 1 1 |

# Problem I. Chocolate is Tasty

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Memory limit: | 512 mebibytes |

A mother wishes to share a chocolate bar among her children. A child can be either a boy or a girl. The children sit in a circle and consecutively take a line of chocolate in clockwise order. At the beginning, the chocolate bar has $n$ rows and $m$ columns. Each boy is greedy and takes one line from the longer side of the chocolate bar. Each girl wishes to keep fit so she takes one line from the shorter side. Mother has to choose from which child she should start to maximize the number of kids that will share the chocolate.

## Input

The first line of input consists of $z$, the number of test cases. The description of the test cases follows.

Each line consists of two integers $n$ and $m$ separated by a space ($1 \le n, m \le 10^6$). They are followed by a space and a sequence of letters "B" and "G" which identify the children on the circle in clockwise order. The length of the sequence is not greater than $10^6$.

## Output

For each test case, print a line with single a interger: the maximal number of children that can eat a chocolate. Do not count the same kid twice even if he/she receive the chocolate twice.

## Example

| standard input | standard output |
|---|---|
| 3<br>4 4 GGGGGGG<br>4 4 BBBBBBB<br>4 4 GBGBBBG | 7<br>4<br>7 |

# Problem J. Game

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Memory limit: | 256 mebibytes |

Let us consider the following two-player game. Initially, there is some sequence of numbers. Then players in turns remove either the first or the last element of the sequence. If after some move, the sequence becomes one of several terminal sequences, then the game ends and the player who made the last move wins. If no player can make a move (because the sequence has length 0), then no player wins. Determine which player has a winning strategy in this game.

## Input

The first line of input contains the number of test cases $z$. The descriptions of the test cases follow.

The first line of each test case contains an integer $n$ ($1 \le n \le 10^6$) followed by $n$ integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$), the initial sequence itself. The second line contains one integer $k$, the number of terminal sequences. The next $k$ lines contain descriptions of terminal sequences. Each of these lines contains an integer $m$ ($m \ge 1$), the length of the sequence, followed by $m$ elements of the sequence: $w_1, w_2, \ldots, w_m$ ($0 \le w_i \le 10^9$). The total length of terminal sequences does not exceed $3 \cdot 10^6$.

Consecutive test cases will be separated by a single blank line.

## Output

For each test case, print "`FIRST`" if the player who makes the first move wins, "`SECOND`" if the other player wins, or "`DRAW`" if no player wins.

## Example

| standard input | standard output |
|---|---|
| 1<br>5 1 1 2 2 1<br>4<br>1 2<br>3 1 2 2<br>2 2 1<br>2 1 1 | SECOND |

# Problem K. Tic-tac-toe

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Memory limit: | 256 mebibytes |

A group of archaeologists has found an ancient clay tablet with a game of Tic-tac-toe engraved on it.

Tic-tac-toe is a paper-and-pencil game for two players, '`O`' and '`X`', who take turns marking the spaces in a $3 \times 3$ grid. The player who succeeds in placing three respective marks in a horizontal, vertical, or diagonal row wins the game.

You, as an employee of the State Historical Museum in Byteozavodsk, are to arbitrate if this state of game could have been created by two excellent players.

## Input

The first line of input contains a single positive integer $t$, the number of test cases. The descriptions of the test cases follow.

Each test case consists of three lines, three characters on each line. The $j$-th character of the $i$-th line denotes the state of the $j$-th square in the $i$-th row of the clay tablet. There are three possibilities:

- "`.`" denotes an empty square,
- "`O`" (big "o") denotes a square that was marked by first player,
- "`X`" denotes a square that was marked by second player.

Each test cases will be preceded by a single blank line.

## Output

For each test case, output a single line containing a single word: "`INVALID`", if there does not exist a valid sequence of alternating moves that leads to this game state, "`UNREACHABLE`", if there does exist a valid sequence of alternating moves that leads to this game state, but only when at least one of players is not excellent, and "`REACHABLE`" otherwise.

## Example

| standard input | standard output |
|---|---|
| 3<br><br>...<br>.X.<br>...<br><br>...<br>.OX<br>...<br><br>...<br>.O.<br>..X | INVALID<br>UNREACHABLE<br>REACHABLE |

## Note

Provided that it is possible, an excellent player always performs a move that lets him win regardless of further moves of his opponent. If it is not possible, then he performs a move that brings him to a draw. If worse comes to worst, he makes any move.