A Digital Friends

Problem

Two positive integers are called *friends* if they consist of the same decimal digits. So 123 and 32331313323213 are friends, but 123 and 22121221 are not.

Two positive integers (that are not friends) are called *almost friends* if a single neighbour exchange in one of them results in a pair of friends. A *neighbour exchange* changes two neighbouring digits a and b into $a-1$ and $b+1$, or into $a+1$ and $b-1$, provided that these new digits are still in the range 0...9, and that no leading zero is generated. So 123 and 2223042 are almost friends (let $04 \rightarrow 13$), and 137 and 470 are neither friends nor almost friends (note that $13 \rightarrow 04$ is not allowed).

The problem is to determine if two given integers are friends or almost friends.

Input

The first line of the input contains a single number: the number of test cases to follow. Each test case has the following format:

- One line with two integers x and y, separated by a single space, with $0 < x, y < 10^{100}$. Both integers start with a non-zero digit.

Output

For every test case in the input, the output should contain a single line with the string `"friends"` or `"almost friends"` or `"nothing"`, reflecting the property of the two given integers.

Example

The examples below correspond to the four examples mentioned in the text.

Input

```
4
123 32331313323213
123 22121221
123 2223042
137 470
```

Output

```
friends
almost friends
almost friends
nothing
```

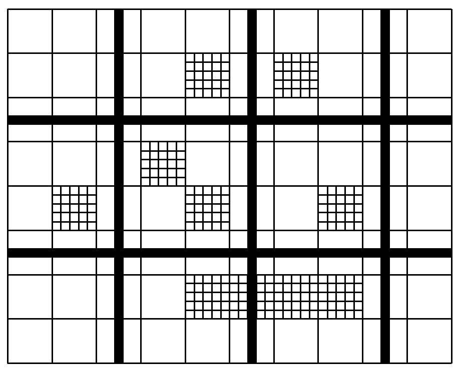
B Make it Manhattan

Problem

Chaos City has grown out of control. Buildings have been built everywhere and the city layout is a complete mess. The mayor decided that this must come to an end, and wants to create a nice, structured city.

After some research, he found the ideal way to make this happen. Inspired by the New York district of Manhattan, he wants all buildings organized in a rectangular grid, separated by avenues running from North to South and streets running from West to East. These streets and avenues should all be separated by the same distance D.

In the current situation, the buildings are already organized in a rectangular grid. In fact, each building fills exactly one square in this grid. However, with all the buildings randomly scattered across the city, it may be impossible to build the roads without demolishing a couple of buildings. To keep most citizens happy, the mayor wants to demolish as few buildings as possible. Given the current locations of the buildings, what is this minimum number?



The above picture illustrates the problem. The shaded squares are the initial locations of the buildings. If the roads should be separated by a distance of three, the thick lines indicate the optimal placement of the roads and one building has to be demolished.

Input

The first line of the input contains a single number: the number of test cases to follow. Each test case has the following format:

- One line with two integers D and N, separated by a single space, satisfying $1 \leq D \leq 1,000$ and $0 \leq N \leq 100,000$: the distance between two roads, and the number of buildings in the city, respectively.
- N lines with two integers x_i and y_i, separated by a single space, satisfying $-10^9 \leq x_i, y_i \leq 10^9$: the positions of the buildings.

Output

For every testcase in the input, the output should contain a single line with the minimum number of buildings that has to be demolished.

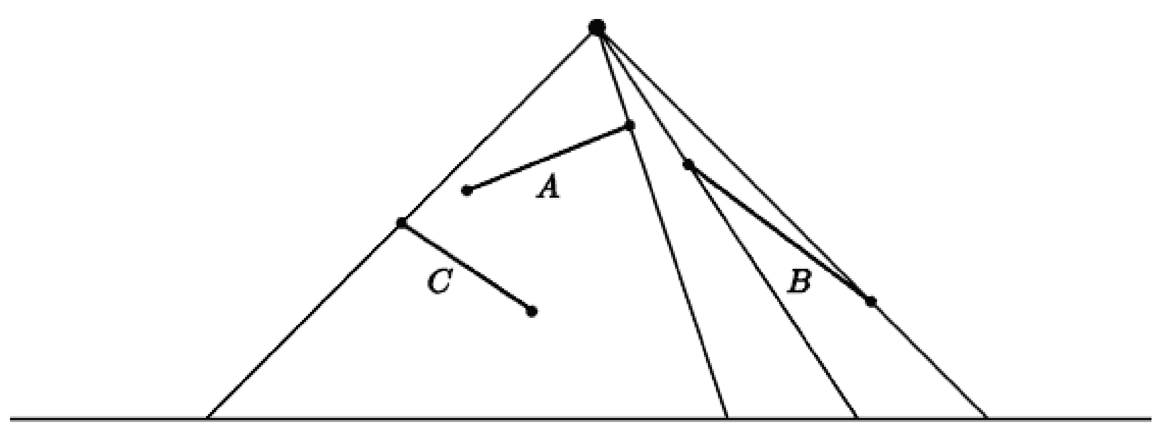
Problem C. Lucky Light

We have a (point) light source at position (x_L, y_L) with $y_L > 0$, and a finite series of line segments, all of finite non-zero length, given by the coordinates of their two endpoints. These endpoints are all different. The line segments are all situated above the x-axis ($y > 0$).

The segments cast their shadows onto the x-axis. We assume that the shadows of two segments either do not overlap at all, or have an overlap that has some non-zero width (they do not just touch). We also assume that for each segment its shadow is more than just one point, i.e., there is no segment that is directed toward the light source. The height of the light source (y_L) is at least 1 unit larger than the y-coordinates of the endpoints of the line segments. This guarantees that indeed each line segment has a bounded shadow on the x-axis.

The collection of shadows divides the x-axis into dark and lighted areas (intervals). The problem is to determine the number of lighted areas — which is at least 2 (if there is at least one line segment, otherwise it is 1).

In the picture below the three line segments A, B and C cause three lighted areas, as indicated.



Input

The first line of the input file contains a single number: the number of test cases to follow. Each test case has the following format:

- One line with one integer n with $0 \leq n \leq 100$: the number of line segments.
- One line with two integers x_L and y_L, the coordinates of the light source, separated by a single space. The coordinates satisfy $-100 \leq x_L \leq 100$ and $1 \leq y_L \leq 1000$.
- n lines, each containing four integers x_i, y_i, u_i and v_i, separated by single spaces, that specify x- and y-coordinates of the two endpoints $(x_i,\ y_i)$ and $(u_i,\ v_i)$ of the i-th line segment, where $-100 \leq x_i, u_i \leq 100$ and $0 < y_i, v_i < y_L$, for $1 \leq i \leq n$.

Output

For every test case in the input file, the output should contain a single number, on a single line: the number of lighted areas.

Examples

standard input	standard output
2 3 50 60 55 45 30 35 64 39 92 18 20 30 40 16 2 -10 50 -10 1 10 11 -10 11 10 1	3 2

Notes

The first test case below corresponds to the picture in the problem description. The second test case has two crossing line segments.

D The Bavarian Beer Party

Problem

The professors of the *Bayerische Mathematiker Verein* have their annual party in the local Biergarten. They are sitting at a round table each with his own pint of beer. As a ceremony each professor raises his pint and toasts one of the other guests in such a way that no arms cross.

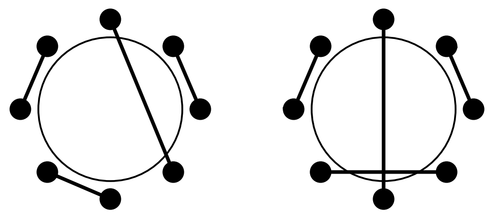


Figure 2: Toasting across a table with eight persons: no arms crossing (left), arms crossing (right)

We know that the professors like to toast with someone that is drinking the same brand of beer, and we like to maximize the number of pairs of professors toasting with the same brand, again without crossing arms. Write an algorithm to do this, keeping in mind that every professor should take part in the toasting.

Input

The first line of the input contains a single number: the number of test cases to follow. Each test case has the following format:

- One line with an even number p, satisfying $2 \leq p \leq 1,000$: the number of participants.
- One line with p integers (separated by single spaces) indicating the beer brands for the consecutive professors (in clockwise order, starting at an arbitrary position). Each value is between 1 and 100 (boundaries included).

Output

For every test case in the input, the output should contain a single number on a single line: the maximum number of non-intersecting toasts of the same beer brand for this test case.

Example

Input

```
2
6
1 2 2 1 3 3
22
1 7 1 2 4 2 4 9 1 1 9 4 5 9 4 5 6 9 2 1 2 9
```

Output

```
3
6
```

E. Exchange

time limit per test: 0.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Given a string of lowercase English letters. You are allowed to choose two letters that exist in any position in the string, replace all occurrences of the first letter you chose with the second one, and replace all occurrences of the second letter you chose with the first one.

Your task is to find the string that comes first in dictionary order among all possible strings that you can get by performing the above operation **at most once**.

For example, by exchanging letter 'a' with letter 'h' in string "hamza", we can get string "ahmzh".

Input

The first line of input contains a single integer T, the number of test cases.

Each test case contains a non-empty string on a single line that contains no more than 10^5 lowercase English letters.

Output

For each test case, print the required string on a single line.

Example

input

```
3
hamza
racecar
mca
```

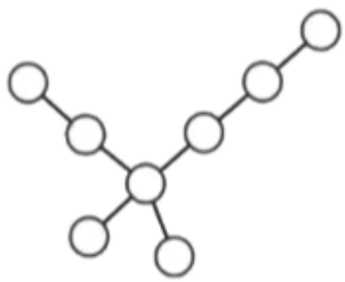
output

```
ahmzh
arcecra
acm
```

F. Starry Night

time limit per test: 6 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Reem likes to look up into the night sky and count how many stars there are. The sky is represented as a tree, and a star is a fixed node with three or more rays of possibly varying lengths leading out of it. A ray is a chain of connected nodes where each node except for the last one is connected to exactly two nodes, and the last one is connected to exactly one node.



A star with four rays.

Not a star.

If Reem can remove as many nodes as she wants, what is the maximum number of stars she can get in a given sky?

Input

The first line of input contains a single integer T, the number of test cases.

The first line of each test case contains a single integers N ($1 \leq N \leq 10^5$), the number of nodes in the sky.

Each of the next $N - 1$ lines contains two integers a and b ($1 \leq a, b \leq N$), and represents a link that connects two nodes.

It is guaranteed that the given graph is a tree.

Output

For each test case, print the maximum number of stars Reem can get after removing zero or more nodes, on a single line.

Example

input

```
1
9
1 2
3 2
5 1
3 7
1 4
3 8
1 6
3 9
```

output

```
2
```

Note

A tree is an undirected graph with exactly one path between any two nodes.

G. Topological Sort

time limit per test: 8 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Consider a directed graph G of N nodes and all edges $(u \to v)$ such that $u < v$. It is clear that this graph doesn't contain any cycles.

Your task is to find the lexicographically largest topological sort of the graph after removing a given list of edges.

A topological sort of a directed graph is a sequence that contains all nodes from 1 to N in some order such that each node appears in the sequence before all nodes reachable from it.

Input

The first line of input contains a single integer T, the number of test cases.

The first line of each test case contains two integers N and M $(1 \le N \le 10^5)$ $(0 \le M \le min(\frac{N(N-1)}{2}, 10^5))$, the number of nodes and the number of edges to be removed, respectively.

Each of the next M lines contains two integers a and b $(1 \le a < b \le N)$, and represents an edge that should be removed from the graph.

No edge will appear in the list more than once.

Output

For each test case, print N space-separated integers that represent the lexicographically largest topological sort of the graph after removing the given list of edges.

Example

input

```
3
3 2
1 3
2 3
4 0
4 2
1 2
1 3
```

output

```
3 1 2
1 2 3 4
2 3 1 4
```

H. Rectangles

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Given an $R \times C$ grid with each cell containing an integer, find the number of subrectangles in this grid that contain only one distinct integer; this means every cell in a subrectangle contains the same integer.

A subrectangle is defined by two cells: the top left cell (r_1, c_1), and the bottom-right cell (r_2, c_2) $(1 \le r_1 \le r_2 \le R)$ $(1 \le c_1 \le c_2 \le C)$, assuming that rows are numbered from top to bottom and columns are numbered from left to right.

Input

The first line of input contains a single integer T, the number of test cases.

The first line of each test case contains two integers R and C $(1 \le R, C \le 1000)$, the number of rows and the number of columns of the grid, respectively.

Each of the next R lines contains C integers between 1 and 10^9, representing the values in the row.

Output

For each test case, print the answer on a single line.

Example

input

```
1
3 3
3 3 1
3 3 1
2 2 5
```

output

```
16
```