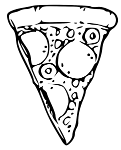
Croatian Open Competition in Informatics

Round 4, January 16th 2021

Tasks

Task	Time limit	Memory limit	Points
Pizza	1 second	512 MiB	50
Vepar	1.5 seconds	512 MiB	70
Hop	1 second	512 MiB	110
Janjetina	1.5 seconds	512 MiB	110
Patkice II	2 seconds	512 MiB	110
Total			450

Task Pizza



After a long day and miserable at work, Mirko decided to order a pizza for dinner to cheer himself up. In a big pile of papers on his desk, he found a flyer of a nearby pizza restarant.

The restarant offers m different pizzas. Pizza toppings are labeled with positive integers. i-th pizza has k_i toppings, with labels $b_{i,1}, b_{i,2}, \ldots, b_{i,k_i}$.

Mirko is very picky when it comes to food. He doesn't like n toppings, those with labels $a_1, a_2, \ldots, a_n$, so he wants to order a pizza that doesn't contain any of those toppings. Determine the number of pizzas that Mirko can order.

Input

The first line contains an integer n ($1 \le n \le 100$), the number of toppings, followed by n distinct integers a_i ($1 \le a_i \le 100$), the labels of toppings Mirko dislikes.

The second line contains an integer m ($1 \le m \le 100$), the number of pizzas.

The following m lines describe the pizzas. The i-th line contains an integer k_i ($1 \le k_i \le 100$), the numer of toppings, followed by k_i distinct integers $b_{i,j}$ ($1 \le b_{i,j} \le 100$), the labels of toppings on the i-th pizza.

The pizzas, i.e. the sets of toppings, will be distinct.

Output

Output the number of pizzas that Mirko can order.

Scoring

In test cases worth 20 points it holds $n = 1$ and $k_1 = k_2 = \cdots = k_m = 1$.

Examples

input

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

output

```
2
```

input

```
2 1 2
4
2 1 4
3 1 2 3
2 3 4
3 3 5 7
```

output

```
2
```

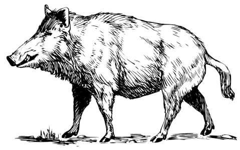
input

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

output

```
3
```

Task Vepar



Two intervals of positive integers $\{a, a+1, \ldots, b\}$ and $\{c, c+1, \ldots, d\}$ are given. Determine whether the product $c \cdot (c+1) \cdots d$ is divisible by the product $a \cdot (a+1) \cdots b$.

Input

The first line contains a single integer t ($1 \le t \le 10$), the number of independent test cases.

Each of the following t lines contains four positive integers a_i, b_i, c_i, d_i ($1 \le a_i \le b_i \le 10^7$, $1 \le c_i \le d_i \le 10^7$).

Output

Output t lines in total. For the i-th test case, output `DA` (Croatian for *yes*) if $a_i \cdot (a_i+1) \cdots b_i$ divides $c_i \cdot (c_i+1) \cdots d_i$, and output `NE` (Croatian for *no*) otherwise.

Scoring

In test cases worth 10 points it holds $a_i, b_i, c_i, d_i \le 50$.
In test cases worth additional 20 points it holds $a_i, b_i, c_i, d_i \le 1000$.
In test cases worth additional 10 points it holds $a_i = 1$.

Examples

input

```
2
9 10 3 6
2 5 7 9
```

output

```
DA
NE
```

input

```
6
1 2 3 4
1 4 2 3
2 3 1 4
1 3 2 4
19 22 55 57
55 57 19 22
```

output

```
DA
NE
DA
DA
DA
DA
```

Clarification of the first example:

We have $9 \cdot 10 = 90$ and $3 \cdot 4 \cdot 5 \cdot 6 = 360$. The answer is `DA` because 90 divides 360.

We calculate $2 \cdot 3 \cdot 4 \cdot 5 = 120$, which doesn't divide $7 \cdot 8 \cdot 9 = 504$. Thus the second answer is `NE`.

Task Hop

♩ Jeremiah was a bullfrog
Was a good friend of mine ♩

There are n water lilies, numbered 1 through n, in a line. On the i-th lily there is a positive integer x_i, and the sequence $(x_i)_{1 \le i \le n}$ is strictly increasing.

Enter three frogs.

Every pair of water lilies (a, b), where $a < b$, must belong to frog 1, frog 2, or frog 3.

A frog can *hop* from water lily i to water lily $j > i$ if the pair (i, j) belongs to it, and x_i **divides** x_j.

Distribute the pairs among the frogs such that no frog can make more than 3 consecutive hops.

Input

The first line contains a positive integer n ($1 \le n \le 1000$), the number of water lilies.

The second line contains n positive integers x_i ($1 \le x_i \le 10^{18}$), the numbers on the water lilies.

Output

Output $n - 1$ lines. In the i-th line, output i numbers, where the j-th number is the label of the frog to which $(j, i + 1)$ belongs.

Scoring

Subtask	Points	Constraints
1	10	$n \le 30$
2	100	No additional constraints.

If in your solution some frog can make k consecutive hops, where $k > 3$, but no frog can make $k + 1$ consecutive hops, your score for that test case is $f(k) \cdot x$ points, where

$$f(k) = \frac{1}{10} \cdot \begin{cases} 11 - k & \text{if } 4 \le k \le 5, \\ 8 - \lfloor k/2 \rfloor & \text{if } 6 \le k \le 11, \\ 1 & \text{if } 12 \le k \le 19, \\ 0 & \text{if } k \ge 20, \end{cases}$$

and x is the number of points for that subtask.

The score for some subtask equals the minimum score which your solution gets over all test cases in that subtask.

Examples

input

```
8
3 4 6 9 12 18 36 72
```

output

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

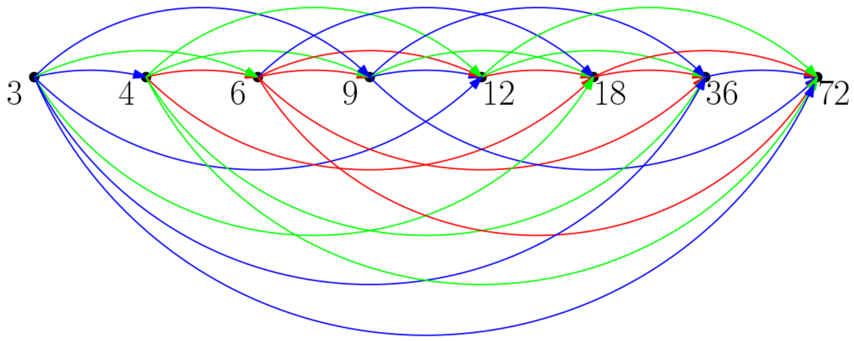
input

```
2
10 101
```

output

```
1
```

Clarification of the first example:



The frogs are marked blue (1), green (2), and red (3).

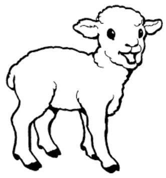
The blue frog can hop from water lily $x_1 = 3$ to water lily $x_4 = 9$, then to water lily $x_7 = 36$, and then to $x_8 = 72$. These are the only three consecutive hops any frog can make.

The green frog can hop from water lily $x_2 = 4$ to water lily $x_5 = 12$, and then to $x_7 = 36$, because 4 divides 12, and 12 divides 36. Those are two consecutive hops.

The red frog cannot hop from water lily $x_2 = 4$ to water lily $x_3 = 6$ because 6 is not divisible by 4.

No frog can make more than three consecutive hops.

Task Janjetina



After restaurants all over Croatia closed because of the lockdown, Mr. Malnar was first overwhelmed with sadness. But he soon realized that there was no point in being sad, and he decided that as soon as the restaurants reopen, he will travel around Croatia and treat himself with the best lamb Croatian restaurants can offer.

Mr. Malnar knows about n cities he could visit, that he labeled with integers from 1 through n. Also, he knows about $n - 1$ two-way roads that connect those cities, in such a way that it is possible to travel between any two cities.

On every road there is a restaurant that serves lamb, and Mr. Malnar knows how many kilograms of lamb he can order in each one.

He will choose two different cities x and y, and travel from x to y via the **shortest** path, i.e. the path that uses the minimal number of roads. He will stop at exactly **one** restaurant, the one where he can order the **maximum** amount of lamb (if there are multiple such restaurants, he will choose any of them), and he will of course eat all the lamb he orders.

Mr. Malnar considers a path of length l on which he eats w kilograms of lamb to be **satisfactory** if $w - l \geq k$. The length of a path is equal to the number of **roads** that it goes through.

He wonders how many ordered pairs of distinct cities (x, y) there are, such that the shortest path from x to y is satisfactory. He is very busy, so he asks you to calculate the answer for him.

Input

The first line contains integers n and k ($1 \leq n, k \leq 100\,000$), the number of cities, and the satisfaction threshold.

Each of the following $n-1$ lines contains three integers x, y and w ($1 \leq x, y \leq n$, $x \neq y$, $1 \leq w \leq 100\,000$), which means that there is a road that connects x and y, and there is a restaurant on that road where Mr. Malnar can order w kilograms of lamb.

Output

Output the number of ordered pairs of distinct cities (x, y), such that the shortest path from x to y is satisfactory.

Scoring

Subtask	Points	Constraints
1	15	$1 \leq n \leq 1000$
2	35	Cities i and $i + 1$ ($1 \leq i \leq n - 1$) are directly connected.
3	60	No additional constraints.

Examples

input

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

output

```
6
```

input

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

output

```
6
```

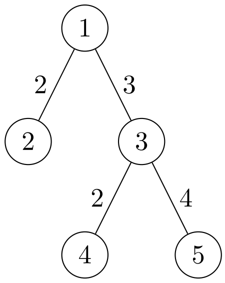
input

```
5 2
1 2 2
1 3 3
3 4 2
3 5 4
```

output

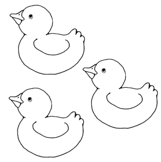
```
8
```

Clarification of the third example:



The pairs are $(1,3)$, $(3,1)$, $(1,5)$, $(5,1)$, $(3,5)$, $(5,3)$, $(4,5)$ and $(5,4)$.

Task Patkice II



After Holywood got its hands on the fascinating story of the successful umbrella voyage between two islands, Netflix executives decided to make a series adaptation of the three ducks' travels.

As you may remember from the first round of COCI 20/21, the ducks have a map of ocean currents. The ducks travel together. The island where the ducks live is marked by the letter `'o'`. The ducks can start their voyage in any of the four directions. Ocean currents in these seas move in one of the four directions, and are marked in the following way: west-east `'>'`, east-west `'<'`, north-south `'v'` and south-north `'^'`. When the ducks are located on a cell with a current, they will move one cell in the direction of the current.

Calm sea is marked by a dot `'.'`. If the currents bring the ducks to a cell with calm sea, outside the map, or back to the starting island, they will stop their voyage. The island that the ducks want to visit is marked by `'x'`.

In order to make the series more appealing, Netflix made a few changes to the story: the sea now may contain wild vortexes (the ducks can get stuck in a cycle) and sea currents that carry the ducks outside the map. *

Therefore, the original map of currents has been changed. But under heavy deadline pressure, the director has made some mistakes: the ducks cannot arrive from the initial to the target island via sea currents anymore.

Netflix directors are very important persons, so they don't really spend time contemplating plot holes. Thus it is your task now to **replace as few as possible characters** on the map, so that the ducks **can go from the initial to the target island**.

For story purposes, the cells with (`'o'` and `'x'`) **cannot be modified**. All other cells are either sea currents or calm sea (characters `'<>v^.'`). You can replace characters in those cells with characters from the same set `'<>v^.'`.

Input

The first line contains integers r and s ($3 \leq r, s \leq 2000$), the number of rows and columns of the map.

Each of the following r lines contains s characters from the set `'o<>v^.x'`, that represent the map of ocean currents. There will always be exactly one character `'o'` and exactly one character `'x'` on the map, and they will not be adjacent.

Output

In the first line output k, the minimum number of changes so that the ducks can go from the initial to the target island.

In the each of the next r lines, output s characters, describing a map which differs from the input map in exactly k cells, satisfying the requirements of the problem.

If there are multiple valid maps, output any of them.

*The ducks also form a heartbreaking love triangle, but that is not important right now.

Scoring

Subtask	Points	Constraints
1	30	$3 \leq r, s \leq 20$
2	80	No additional constraints.

If in all test cases in some subtaks the first line (minimum number of changes) is correct, but the map in some test case is not valid, you will get half of the points for that subtask.

Examples

input

```
3 3
>vo
vv>
x>>
```

output

```
1
>vo
vv>
x<>
```

input

```
3 6
>>vv<<
^ovvx^
^<<>>^
```

output

```
2
>>vv<<
^o>>x^
^<<>>^
```

input

```
4 4
x.v.
.>.<
>.<.
.^.o
```

output

```
4
x<<.
.>^<
>.<^
.^.o
```