Problem A. Renovation

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

You want to renovate your room and hang wallpapers. The room has dimensions of $N \times M$ meters and the ceiling height is H meters. The next rules should followed when hanging wallpapers:

- Line parallel to the sides wallpaper roll should be vertical. The rotation on 90 or 270 degrees is forbidden.
- Each hanged stipe of the wallpaper should start at the ceiling and end at the floor.
- Each roll could be cut only perpendicularly to sides.
- The stripes are hanged next to each other. (Except for the last stipe that could slighlt cover the first stripe)
- The stripe could be flexed in the corners of the room. (So, the stripe could cover two or more walls of the room)

You are given the dimensions of the room N and M, the ceiling height H, the length of the roll L and its width W.

You have to calculate the smallest number of rolls to buy to paper all the walls.

Input

The only line of the input contains five integers N, M, H, L and W — the parameters of the room and the roll. All the integers are in $[1, 10000]$. It is guaranteed that $H \leq L$.

Output

The only line of the output should contain the minimal number of rolls.

Example

system input	system output
4 5 3 7 1	9

Problem B. Billboard

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

The small advertising agency was asked to build a new billboard. The agency has n plates of some height and width 1. The billboard is constructed from the vertical columns. Each column has width 1 and consists of two plates. The height of all the columns in the billboard should be the same. (In other words a billboard should look like rectangle) What is the maximal width of the billboard that could be constructed from the given plates?

Input

The first line of the input contains n ($2 \leq n \leq 10^6$) — the number of plates. The second line contains n integers $l_1, \ldots, l_n$ ($1 \leq l_i \leq 2000$) — the heights of the plates.

Output

The sole line of the output should contain two integers — the maximal width of the billboard and the number of different heights such billboard could have.

Example

system input	system output
4 2 4 3 5	2 1
4 1 3 9 27	1 6

Problem C. Polynomial

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

You have to find a polynomial $p(x)$ such that:

- for all i from 1 to n $p(i) = i$ modulo 998244353;
- $p(n+1) \neq n+1$ modulo 998244353.

Input

The sole line of the input contains one integer n ($0 \leq n \leq 2000$).

Output

The first line of the output should contain m ($0 \leq m \leq 2000$) — the degree of the polynomial. The second line should contain $m+1$ integers $p_m, p_{m-1}, \ldots, p_0$ — the coefficients of the polynomial near $x^m, x^{m-1}, \ldots, x^0$, respectively. If there are several answers, you could return any of them.

Example

system input	system output
1	1 2 998244352
0	1 1 1
2	2 1 998244351 2

Problem D. The door

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

Playing the game “Super-quest 2000” you found a door with the fraction $\frac{a}{b}$ written on it. The door has a minicomputer that contains two fractions x and y. At the beginning $x = \frac{0}{1}$ and $y = \frac{1}{1}$. Also, the door has two buttons ‘u’ and ‘d’. When the button ‘u’ is pressed then y is added to x, when the button ‘d’ is pressed then y is subtracted from x. If the pressed button differs from the button pressed before, then before the operation y is divided by 2.

For example, if you pressed `uududd`, then $x = \frac{1}{1} + \frac{1}{1} - \frac{1}{2} + \frac{1}{4} - \frac{1}{8} - \frac{1}{8} = \frac{12}{8}$.

To open the door you have to make x differ from $\frac{a}{b}$ by no more than 10^{-6}.

You have to find the sequence of buttons to press of length no more than 200 that opens the door.

Input

The only line of the input contains two integers a and b. $(1 \le a, b \le 100)$.

Output

The only line of the output should contain the sequence consisting of letters `u` and `d` — the sequence of buttons to press that opens the door. The length of the string should not exceed 200 characters.

Example

system input	system output
12 8	uud
1 3	udduudduudduudduudduudduudduudduud
1 1	u

Problem E. Parkour

Input file: system input
Output file: system output
Time limit: 6 seconds
Memory limit: 256 megabytes

The Mirror city is represented by a rectangular grid $N \times M$. Each square of this grid is a building with height $h_{i,j}$. You are located on the upper-left building and you have to reach the bottom-right building. You are already a good parker and you could jump from the higher building to the lower one, but to get from the lower building to the higher building you need a staircase. If the difference in heights is n, your staircase should have length not less than n. Because you have to carry the staircase with you, you want it to be as shorter as possible. What is the smallest length of staircase you need to reach a goal?

Input

The first line of the input contains two integers N and M $(1 \le N, M \le 1000)$. Each of the next N lines contains M integers. Each integer represents a height $h_{i,j}$ of the building located in the corresponding square $(0 \le h_{i,j} \le 10^9)$. The first integer in the first line corresponds to the start point and the last integer in the last line corresponds to the end point.

Output

The sole line of the output should contain the smallest length of the staircase that you need to reach the bottom-right building.

Example

system input	system output

7 5
20 21 22 23 24
21 30 26 27 26
22 20 28 31 34
24 20 24 24 32
26 28 30 30 35
35 34 32 20 35
36 37 38 31 35
	3

Problem F. Comparison of coins

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

You are doing homework on the comparison of n coins with a partner. Your partner has done all the weighings and gives you the results as the list of outcomes in the form: “the weight of the coin A is bigger than the weight of the coin B” or “the weight of the coin A equals to the weight of the coin B”.

You know that your partner could make a mistake. You have to check whether the results of the weighings are possible.

Input

The first line of the input contains two integers n and w ($2 \le n \le 5 \cdot 10^4$, $1 \le w \le 2.5 \cdot 10^5$) — the number of coins and the number of weighings.

Each of the next w lines has the format: one integer a, then the character ‘`>`’ or ‘`=`’ and one integer b ($0 \le a, b \le n$). a and b are the coins. The character ‘`>`’ means that the weight of the coin a is bigger than the weight of the coin b. The character ‘`=`’ means that the weight of the coin a equals to the weight of the coin b.

It is guaranteed that no pair of coins is compared twice and that each coin is compared at least once.

Output

If the results of the weighings is possible then the sole line of the output should contain “`Possible`”, otherwise, it should contain “`Impossible`”.

Example

system input	system output

`3 3`
`0 > 1`
`1 = 2`
`0 = 2`
	`Impossible`

`5 5`
`0 = 1`
`1 = 2`
`3 = 4`
`0 > 3`
`1 > 4`
	`Possible`

`6 5`
`0 > 1`
`1 > 2`
`3 = 4`
`4 = 5`
`5 > 3`
	`Impossible`

Problem G. Anti-advertisement

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

In the last decade the competition between big companies has evolved from the standard advertisement or anti-advertisement to more sophisticated strategies. Two big companies iByte and Byteoid are ready to release their new smartphones. Byteoid decides to uses anti-advertisement to reduce the activity of the iByte supporters in the internet. Each user is characterized by non-zero real preference x_i: users with negative value support iByte and users with positive value supports Byteoid; and by activity p_i: how often the user will defend his point of view in the internet.

Each anti-advertisement banner is characterized by the preference interval $[l_i, r_i]$ and by the coefficient of demotivation $0 \le d_i \le 1$. This means, that if the banner is shown to the person with $x_i \in [l_i, r_i]$, then his activity becomes $p_i \cdot d_i$.

You have m banners. You need to select some subset of them in the following way:

- The difference between the sum of activities of users, who support Byteoid, and the sum of the activities of users, who support iByte, should be maximized.
- Each person is affected by no more than one banner.

Input

The first line of the input contains k ($1 \le k \le 10$) — the number of test cases.

The first line of each test case contains two integers n ($1 \le n \le 200$) and m ($1 \le m \le 50$) — the number of users and the number of banners.

Each of the next n lines describes a user with two real numbers x_i and p_i ($-1 \le x_i \le 1$, $0 \le p_i \le 1$) — the preference of the user and his activity. The users are sorted by non-decreasing x_i.

Each of the next m lines describes a banner with three real numbers l_i, r_i and d_i ($-1 \le l_i \le r_i \le 1$, $0 \le d_i \le 1$) — left and right ends of the preference interval and the coefficient of demotivation. Banners are sorted by non-decreasing r_i.

All values in the input files are given with no more than 10 digits after the decimal point. It is known that no x_i is equal to l_j or r_j.

Output

For each of the test case output in the separate line the maximal difference between the sum of activities of users with positive preference and the sum of activities of users with negative preference with the precision of 10^{-2}.

Example

system input	system output

1
5 4
-0.94 0.34
-0.51 0.22
-0.21 0.82
0.33 0.43
0.82 0.82
-0.61 -0.41 0.96
-1 0 0.8
-0.35 0.46 0.52
-0.95 0.91 0
	0.15

Problem H. Treasure

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

The island is represented as a square $N \times N$, properly oriented: up — north, right — east, down — south, left — west. Some of the squares of the island are filled with the water.

You read in the old book that there is a treasure buried in the island. To find it you have to start in the center of some square then go one fourth of the path to the east, then one half of the path to the south and, finally, one fourth of the path again to the east. The length of the path is always divisible by four and the path cannot pass through water.

You need to find the number of possible pairs: the start square and the square with the treasure.

Input

The first line of the input contains one integer N ($1 \leq N \leq 2000$) — the size of the island. Each of the next N lines contains N symbols. i-th symbol in j-th row equals to '`.`', if the square is not filled with water, and equals to '`x`', otherwise.

Output

The sole line of the output should contain one integer — the number of possible pairs of the start square and the square with the treasure.

Example

system input	system output
5 ..... .x.x. .x...	3
6	20

Note

In the first example the following variants are possible:

```
.K+..                                .....                                K-+..
.x|x.                                .x.x.                                .x|x.
.x+K.                                .xK+.                                .x|..
.....                                ...|.                                ..|..
.....                                ...+K                                ..+-K
```

Problem I. Price of the collection

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

The employees of the "Bytis" auction calculates the price of the collection as follows.

Let the collection consists of k items with prices $b_1, \ldots, b_k$. Then the partial q-price is calculated as the sum of the prices of $\min(q, k)$ most valuable items from collection.

Then the full q-price of the collection $a_1, \ldots, a_n$ is calculated as the sum of the partial q-prices of all non-empty subsets of the collection.

For the given collection $a_1, \ldots, a_n$ calculate the full i-price for all i from 1 to n.

Input

The first line of the input contains one integer t ($1 \leq t \leq 10$) — the number of test cases.

The first line of each test case contains one integer n ($1 \leq n \leq 10^5$) — the number of the items in the collection. The next line contains n integers a_i ($0 \leq a_i \leq 10^9$) — the prices of the items in the collection.

Output

For each test case output in the separate line n integers; i-th integer corresponds to the full i-price modulo 998244353.

Example

system input	system output

2
3
2 2 2
5
5 4 3 2 1
	14 22 24
	129 201 231 239 240