Information

Memory limit

The limit is 512 MiB for each problem.

Source code limit

The size of each solution source code can't exceed 256 KiB.

Submissions limit

You can submit at most 50 solutions for each problem.

You can submit a solution to each task at most once per 30 seconds. This restriction does not apply in the last 15 minutes of the contest round.

Scoring

Each problem consists of several subtasks. The subtask score is awarded if all tests in the subtask are passed.

The number of points scored for the problem is the total number of points scored on each of its subtasks. The score for the subtask is the maximum number of points earned for this subtask among all the solutions submitted.

Feedback

To get feedback for your solution, go to "Runs" tab in PCMS2 Web Client and use "View Feedback" link. In each problem of the contest you will see the score for each subtask, or the verdict for the first failed test.

Scoreboard

The contestants' scoreboard is available during the contest. Use "Monitor" link in PCMS2 Web Client to access the scoreboard. The standings provided in PCMS2 Web Client are not final.

Problem A. Alphabet Contest

Time limit: 1 second

Kindergarten graduates participate in English alphabet Olympiad. The main task is to pronounce English letters in the alphabet order without repeats.

Children can start to pronounce letters at any moment, even when the other participant has not finished yet. At the same time, the teacher writes all pronounced letters into one common string. The task is not easy for children and sometimes they make mistakes such as skipping letters. For each participant the number of mistakes they make is the number of skipped letters. The total number of mistakes for all children doesn't exceed k. If at some point participant is tired to pronounce letters, (s)he stops, and all the remaining letters are not counted as mistakes.

You know k and the final string, your task is to find the minimum possible number of participants, or say that data is not correct.

Input

The first line contains a single integer k ($0 \le k \le 1000$) — the maximum number of skipped letters.

The second line contains teacher's string s — all pronounced letters. s consists of capital English letters and the length of the string does not exceed 1000.

Output

Print single word “`Impossible`” in a single line if the data is incorrect (it isn't possible to get this line with missing only k letters), otherwise print single integer — the minimum number of participants in the Olympiad.

Scoring

Subtask	Score	Constraints
1	16	$k = 0$
2	16	all letters in s are in alphabet order
3	11	s_i equals ‘`a`’ or ‘`b`’
4	11	it's possible to get a string by skipping no more than k letters, and the minimum number of participants doesn't exceed 3
5	46	no additional constraints

Examples

standard input	standard output
5 ABDBCBADB	4
100 INNOPOLIS	4
0 ABB	Impossible

Explanations

In the first example, the rows separately for each participant might look like this:
Participant 1: `ABD` — 1 mistake;

Participant 2: `BC` — 1 mistake;
Participant 3: `BD` — 2 mistakes;
Participant 4: `AB` — 0 mistakes;

In the second example:
Participant 1: `INO` — 12 mistakes;
Participant 2: `NOP` — 13 mistakes;
Participant 3: `LS` — 17 mistakes;
Participant 4: `I` — 8 mistakes;

Problem B. Need More T-shirts!

Time limit: 1 second

The Innopolis Open 2020 is coming. And this means that it's time to order T-shirts for the participants of the Olympiad.

The organizers decided that it would be great to buy T-shirts of different colors so that you can easily see who is the organizer, jury or participant of the Olympiad.

For this task, two people were assigned, they decided how many T-shirts of each color should be ordered. However, they did not agree among themselves, and one of them wrote down the number of T-shirts of a certain color that needs to be ordered, and the other wrote the percentage of T-shirts of a certain color. As a result, they got a list of length n, where each element of the list a_i is either the number of T-shirts of color i or the percentage of T-shirts of color i of the total number of T-shirts. Moreover, it is known that in the list at least one element a_i is the number of T-shirts of the color i.

Now the organizers need to understand, what is the total of T-shirts they need to buy for the olympiad. Help them find all possible options for the total number of T-shirts for which such a list could be obtained.

Input

The first line contains a single integer n, the number of different colors ($1 \le n \le 10^5$).

The second line contains n integers a_i, either the number of T-shirts or the percentage of t-shirts for the color i ($1 \le a_i \le 10^9$).

Output

In the first line print the number of options for the total number of T-shirts for which such a list could be obtained.

In the second line print all possible options for the total number of T-shirts, in sorted order.

Scoring

Subtask	Points	Constraints
1	13	$1 \le n \le 15$, $1 \le a_i \le 100$
2	21	$1 \le n \le 100$, $1 \le a_i \le 100$
3	38	$1 \le n \le 100$, $1 \le a_i \le 10^9$
4	28	$1 \le n \le 10^5$, $1 \le a_i \le 10^9$

Examples

standard input	standard output
2 1 50	2 2 51
3 20 30 70	4 120 125 140 300
4 2 40 90 5	3 137 470 840
7 10 20 5 15 30 17 3	1 100

Explanation

Explanation of the first example:

If the total number of T-shirts is 2: one of color 1, and one of color 2, we can obtain the given list if the first element (1) is the number of T-shirts of color 1, and the second element (50) is the percentage of T-shirts of color 2. $2 = 1 + 2 \cdot \frac{50}{100}$.

If the total number of T-shirts is 51: one of color 1, and 50 of color 2, we can obtain the given list, if both elements are the numbers of T-shirts of colors 1 and 2. $51 = 1 + 50$.

Explanation of the second example:

$120 = 20 + 30 + 70$

$125 = 125 \cdot \frac{20}{100} + 30 + 70$

$140 = 140 \cdot \frac{20}{100} + 140 \cdot \frac{30}{100} + 70$

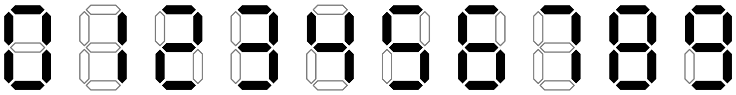
$300 = 300 \cdot \frac{20}{100} + 30 + 300 \cdot \frac{70}{100}$

Problem C. The Final Countdown

Time limit: 2 seconds

Rumor has it that IOI 2048 is going to be held in Innopolis! In this regard, a large display counter was installed in Innopolis, indicating the number of nanoseconds before the start of the Olympiad! Initially, a certain number was set to be displayed, and every nanosecond this number decreases by one. Leading zeros are not displayed.

The digits are displayed using standard seven-segment indicators. The way the digits look is shown in the picture:



Nanoseconds change so quickly that it's almost impossible to see the number the display shows at the moment. But, by connecting a high-precision sensor attached to the power cord of the display, it was possible to obtain the values a_i — the number of segments turned on during the each of n nanoseconds in a row. Since there is still time before IOI 2048, the number on the timer at any time was positive.

Write a program that calculates the number of possible initial values (corresponding to the measurement a_1), and any m of these values. If the number of possible initial values are less than m, you should print all of them.

Input

The first line contains two numbers n and m ($1 \le n \le 10^5$, $0 \le m \le 10$) — the number of nanoseconds and the number of values to print. The next line contains n integers a_i ($2 \le a_i \le 1000$) — the number of segments that are turned on during the i-th nanosecond.

Output

In the first line print k — the number of possible initial values of the counter modulo 1 000 000 007. Then, print m different initial values of the number on the display, consistent with the given measurements. If the actual number of values (before taking it modulo 1 000 000 007) is less than m, print all the suitable values. You can print these numbers in any order, each one in a separate line.

Scoring

Subtask	Points	Additional constraints
1	17	$n = 1$, $m = 0$, $a_i \le 100$
2	10	$n = 1$, $m \le 10$, $a_i \le 100$
3	19	$n \le 100$, $a_i \le 10$
4	22	$n \le 100$, $a_i \le 20$
5	32	no additional constraints

Examples

standard input	standard output
5 1 11 15 14 15 11	3 1151
10 1 13 10 14 12 13 9 12 11 10 11	1 102
4 10 29 28 29 29	108637 448765 812225 19541715 417773115 161177415 117347175 206215 8127315 12761415 54110175
1 6 6	6 111 41 14 6 77 9

Explanations

In the first example, if initial value on the display is 1151, then it changes in the following way:

Number on the display	Number of segments
1151	$2 + 2 + 5 + 2 = 11$
1150	$2 + 2 + 5 + 6 = 15$
1149	$2 + 2 + 4 + 6 = 14$
1148	$2 + 2 + 4 + 7 = 15$
1147	$2 + 2 + 4 + 3 = 11$

Other possible initial values are 451 and 761, you can print any of them.

Problem D. Painting

Time limit: 1.5 seconds

Mitya has a rectangular canvas, which can be represented as a table of size $n \times m$, and k robots. We number the rows of the table from 1 to n from top to bottom, and the columns from 1 to m from left to right. Cell (x, y) is at the intersection of row x and column y.

Initially, all the cells of the table are painted with white color, which has the number 0. Robot number i is filled with paint with color number i $(1 \le i \le k)$. For each robot, Mitya chose a rectangle, which is described by four integers $x_{i,1}$, $y_{i,1}$, $x_{i,2}$, and $y_{i,2}$. It contains all cells (x, y), such that $x_{i,1} \le x \le x_{i,2}$ and $y_{i,1} \le y \le y_{i,2}$ $(1 \le x_{i,1} \le x_{i,2} \le n,\ 1 \le y_{i,1} \le y_{i,2} \le m)$. After that, the robots, one after another, in random order, painted their rectangle with their color. When painting, the robot changes the color of all cells on the rectangle to a new color. If the cell was painted before, the old color of this cell is changed to the new color, and can not be restored.

The next day, his friend Lesha saw the canvas. He noticed that for each color from 1 to k there is at least one cell painted with that color. He is trying to determine for each robot, what rectangle was assigned to it. Namely, he is interested in whether it is possible to assign a rectangle to each robot in such a way that there is an order of robots that would result the same picture. And if Mitya could choose such rectangles, Lesha wonders if he could do it the only way. Two ways are considered different if there is a robot which has different rectangles in these methods. Two rectangles are considered different if they are different in at least one of the four coordinates x_1, y_1, x_2 or y_2.

Input

The first line contains three integers n, m, and k, the height and width of the canvas, and the number of robots $(1 \le n, m \le 2\,000;\ 1 \le k \le 1\,000)$.

Each of the next n lines contains m integers c_{ij}, the colors of the cells of the table $(0 \le c_{ij} \le k)$.

It is guaranteed that for every color from 1 to k there is at least one cell painted with it.

Output

If there is no way that Mitya could assign rectangles to robots, so that there is an order of robots that would result in the same picture, print "`No solution`".

If there is only one way, print "`Single solution`" in the first line. In the next k lines print four integers $x_{i,1}$, $y_{i,1}$, $x_{i,2}$, and $y_{i,2}$, which describe the rectangle for i-th robot $(1 \le x_{i,1} \le x_{i,2} \le n,\ 1 \le y_{i,1} \le y_{i,2} \le m)$. In the last line print a permutation of numbers from 1 to k, the order of the robots, which would result in the given picture.

If there is more than one way, print "`Multiple solutions`" in the first line. In the next line print "`First`". In the next lines, print the description of the first way in the same format as in the previous paragraph. In the next line print "`Second`". And in the next lines print a description of the second way. The first and second ways should be different. If there are more than two different ways, you can print any two different ones.

Scoring

Subtask	Points	Additional constraints
1	5	$k = 1;\ n, m \le 300$
2	5	$k \le 2;\ n, m \le 300$
3	10	$k \le 5;\ n, m \le 300$
4	25	$n = 1$
5	25	$n, m, k \le 300$
6	30	no additional constraints

Examples

standard input	standard output

3 4 3
0 3 3 0
1 3 2 1
1 1 2 1
	Single solution
	2 1 3 4
	2 3 3 3
	1 2 2 3
	1 3 2

2 2 2
1 2
2 1
	No solution

2 1 2
2
1
	Multiple solutions
	First
	2 1 2 1
	1 1 2 1
	2 1
	Second
	1 1 2 1
	1 1 1 1
	1 2

2 3 3
0 3 0
2 0 1
	Single solution
	2 3 2 3
	2 1 2 1
	1 2 1 2
	1 2 3

Problem E. Chicken Farm

Time limit: 1 second

Askhat is a young businessman. He realized that programming is not a very profitable business. Therefore, he decided to start a chicken farm.

On his farm, there are n chickens that stand in a row. The i-th chicken can eat no more than a_i grains. There are m feeders, each one is described by the numbers l_j, r_j and c_j. Chicken i can eat from feeder j, if $l_j \le i \le r_j$, and all chickens in total can eat no more than c_j grains from the j-th feeder.

As in any business, problems arise from nowhere. This time the government inspector came to his farm. He said that by current regulations, there should be at least two chickens that can eat from all feeders. In other words, there should be an integer i such that $1 \le i \le n-1$, and for all feeders $l_j \le i$ and $i+1 \le r_j$. All feeders that don't satisfy this property will be destroyed. Askhat asks you to find for each i what the maximum number of grains can be fed to chickens if you leave only feeders, from which chickens i and $i+1$ can eat.

Input

The first line contains a single integer t — the number of tests in the input ($1 \le t \le 2\,000$).

The following lines describe the given tests. The first line of each test contains two integers n and m — the number of chickens and the number of feeders, respectively ($1 \le n \le 2\,000$, $1 \le m \le 100\,000$). The following line contains n integers a_i — the maximum number of grains that the i-th chicken can eat ($0 \le a_i \le 10^9$). The following m lines contain three integers each l_j, r_j, and c_j describing the j-th feeder ($1 \le l_j \le r_j \le n$, $0 \le c_j \le 10^9$).

The total sum of all n in the input doesn't exceed $2\,000$.

The total sum of all m in the input doesn't exceed $100\,000$.

The total sum of all $n \cdot m$ in the input doesn't exceed 10^7.

Output

For each test print $n-1$ integers: the i-th of these integers should equal to the maximum number of grains that can be fed to chickens if you leave only feeders with $l_j \le i$ and $r_j \ge i+1$.

Scoring

Subtask	Points	$\sum n$	$\sum m$	Additional constraints
1	5	$\sum n \le 100$	$\sum m \le 100$	—
2	10	$\sum n \le 500$	$\sum m \le 500$	—
3	25	$\sum n \le 1\,000$	$\sum m \le 1\,000$	—
4	10	$\sum n \le 2\,000$	$\sum m \le 100\,000$	$l_i \le l_{i+1}$, $r_i \le r_{i+1}$
5	20	$\sum n \le 500$	$\sum m \le 100\,000$	—
6	30	$\sum n \le 2\,000$	$\sum m \le 100\,000$	—

Example

standard input	standard output

1
5 3
5 2 2 3 1
1 4 4
3 5 4
3 4 1
	4 4 9 4

Explanations

If you leave the feeders, from which chickens 1 and 2 can eat, then only the first feeder will remain. In this case, you can feed the first chicken all the grains from it, and four grains will be fed.

Similarly, if you leave the feeders, from which the second and third chickens can eat.

If you leave the feeders, from which chickens 3 and 4 can eat, then all the feeders will remain. Then you can feed the first chicken grains from the first feeder, and the third and fourth chickens grains from the remaining feeders. Thus, nine grains will be fed.

In the last case, you leave the feeders, from which chickens 4 and 5 can eat. Only the second feeder will remain. You can feed all grains from it.