Hong Kong Olympiad in Informatics 2016/17 Senior Group

Task Overview

ID	Name	Time Limit	Memory Limit	Subtasks
S171	TV Ratings	1.000 s	256 MB	7 + 8 + 12 + 21 + 9 + 17 + 26
S172	Degradion	1.000 s	256 MB	15 + 30 + 25 + 30
S173	Monster GO	1.000 s	256 MB	9 + 9 + 22 + 23 + 37
S174	Magic Triangle II	0.500 s	256 MB	11 + 15 + 10 + 18 + 25 + 21

Notice:
Unless otherwise specified, inputs and outputs shall follow the format below:

- One space between a number and another number or character in the same line.
- No space between characters in the same line.
- Each string shall be placed in its own separate line.
- Outputs will be automatically fixed as follows: Trailing spaces in each line will be removed and an end-of-line character will be added to the end of the output if not present. All other format errors will not be fixed.

C++ programmers should be aware that using C++ streams (`cin` / `cout`) may lead to I/O bottlenecks and substantially lower performance.

For some problems 64-bit integers may be required. In Pascal it is `int64`. In C/C++ it is `long long` and its token for `scanf` /`printf` is `%lld`.

All tasks are divided into subtasks. You need to pass all test cases in a subtask to get points.

S171 - TV RATINGS

Time Limit: 1.000 s / Memory Limit: 256 MB

TV Ratings are means to measure the audience of television programmes. The estimated number of viewers gives the station and advertisers an idea how popular a programme is. One way to measure the audience size is to install a remote control sensor near the TV in selected households. The sensor detects when the power button and channel buttons are pressed. Such data is sent back to the ratings agency regularly. For example, assuming the TV is off initially, if the power button is pressed two times, once at 2pm and and once at 3pm, that means the household watched one hour of TV programme. Your task is to write a program that processes sensor data from the button presses by households for one day.

In Byteland there are N TV channels, numbered $1, 2, \ldots, N$. Channel i has C_i programmes for the day and the j^{th} programme lasts for $D_{i,j}$ seconds. All channels start broadcasting their first programme at midnight $(t = 0)$. The last programmes end exactly at midnight next day $(t = 86400)$. There are no breaks between programmes.

On the remote control there are $N + 1$ buttons. Button 0 represents the power button and buttons 1 to N corresponds to the channels. When the power button is pressed, the TV turns itself on (if it was off) or off (if it was on). When channel button j is pressed, the TV changes to channel j **if and only if it is currently on**. This has no effect if the TV is already set to channel j. Pressing a button takes negligible time. A household is regarded as watching a programme if and only if the TV is on and is set to the channel showing the programme.

M households, numbered $1, 2, \ldots, M$, take part in the measurement. The initial state of the television at midnight in household i can be represented by two integers: P_i and Q_i. If the television is on initially, $P_i = 1$. Otherwise, $P_i = 0$. Q_i is an integer between 1 and N (inclusive) which is the channel the TV is set to initially.

Altogether there will be L sensor records listed in chronological order. Each record consists of three integers: T_i, H_i, and B_i. T_i is the number of seconds past midnight when the button is pressed (assume that it happened at the beginning of the second), H_i is the household number that pressed the button and B_i is an integer between 0 and N (inclusive) which stores the button pressed.

Based on the sensor data, your program should output a report that contains the average viewership for each programme. The average viewership is calculated as the total number of seconds spent watching the programme by all households divided by the duration of the programme. For example, for a 60-minute programme watched by 2 households for 30 minutes and 1 household for 45 minutes, the average viewership would be $\frac{1800+1800+2700}{3600} = 1.75$.

You are strongly recommended to complete the subtasks first to familiarise yourself with the problem and input format.

INPUT

The first line contains 3 integers N, M, and L -- the number of channels, number of households and number of sensor data records respectively.

The i^{th} line of the next N lines describes the programme on Channel i. The first integer on the line is C_i $(1 \le C_i \le 86400)$ -- the number of programmes scheduled on the channel for the day. C_i integers follow, the j^{th} of which is $D_{i,j}$ $(1 \le D_{i,j} \le 86400)$ -- the duration of the j^{th} programme in seconds. It is guaranteed that $\sum_{j=1}^{C_i} D_{i,j} = 86400$.

The next M lines describe the initial states of the TVs in the households. The i^{th} line contains two integers P_i $(P_i \in \{0, 1\})$ and Q_i $(1 \le Q_i \le N)$ -- whether the TV is on and the channel set to initially.

The next L lines describe the sensor data records. The i^{th} line contains three integers T_i $(0 \le T_i \le 86400)$, H_i $(1 \le H_i \le M)$, and B_i $(0 \le B_i \le N)$ -- the time when the button is pressed, the household number and the button pressed. The records are listed in chronological order, i.e. $T_i \le T_{i+1}$. If $T_i = T_{i+1}$, record $i + 1$ happens just after i.

OUTPUT

Output N lines -- one for each channel. On the i^{th} line output C_i numbers separated by spaces -- the average ratings of the programmes in order.

If for each pair of expected answer a and your output x, $\frac{|x-a|}{max(1,a)} \leq 10^{-6}$, your answer will be accepted. In other words, the absolute or relative error, whichever is less, should not exceed 10^{-6}. It is not allowed to output the numbers in scientific notation. If you output a decimal point, it must be preceded by and followed by at least 1 decimal digit.

SAMPLE TESTS

1

Input

```
1 1 4
12 7200 7200 7200 7200 7200 7200 7200 7200 7200 7200 7200 7200
0 1
14400 1 0
28800 1 0
36000 1 0
72000 1 0
```

Output

```
0 0 1 1 0 1 1 1 1 1 0 0
```

2

```
1 4 4
3 6780 60060 19560
1 1
1 1
1 1
1 1
960 4 0
49920 3 0
66840 2 0
66840 1 0
```

```
3.141593 2.718282 0
```

3

```
2 2 5
2 36000 50400
3 28800 28800 28800
0 2
1 1
9000 1 1
18000 2 2
36000 2 1
36000 1 0
54000 1 1
```

```
0.5 1.642857
0.375 0.875 0
```

SUBTASKS

For all cases: $1 \leq N, M, L \leq 100000$, $\sum_{i=1}^{N} C_i \leq 100000$

	Points	Constraints
1	7	There is only 1 channel. The channel has 12 programmes of 2 hours each. There is only 1 household. The household does not turn the TV on or off during a programme. The household does not press channel buttons. $N = M = 1$ $C_1 = 12$ $D_{1,j} = 7200$ T_i is a multiple of 7200 $B_i = 0$
2	8	There is only 1 channel. There is only 1 household. The household does not press channel buttons. $N = M = 1$ $B_i = 0$
3	12	There is only 1 channel. There can be many households. The household does not press channel buttons. $N = 1$ $B_i = 0$
4	21	The households do not press channel buttons. $B_i = 0$
5	9	There are not more than 100 channels. Each channel has 12 programmes of 2 hours each. There is only 1 household. The household does not turn the TV on or off or press channel buttons during a programme. $1 \leq N \leq 100$ $M = 1$ $C_1 = 12$ $D_{1,j} = 7200$ T_i is a multiple of 7200
6	17	There are not more than 100 channels. There is only 1 household. $1 \leq N \leq 100$ $M = 1$
7	26	No additional constraints

S172 - DEGRADION

Time Limit: 1.000 s / Memory Limit: 256 MB

Lord Manholition is the strongest student in HKOI School. In this semester, he has attempted in N courses and achieved score A_i in the i^{th} course. Of course, the grades are too superior and Lord Manholition isn't very happy with this (which is strange). To fulfill his wish to pretend weak (which is even more strange), he decides to cast some spells, called "Degradion", so that his scores become lower.

Lord Manholition's spells can be cast between 0 and 5×10^6 (inclusive) times. Every time he casts a spell, he can target one of the courses that has the highest score currently, and reduce its score by any positive integer x, which can be controlled by Lord Manholition by adjusting the spell's strength. Therefore, the values of x chosen each time can be described by a sequence $X_1, X_2, \ldots, X_K$, where K is the number of spell casts and X_i $(X_i > 0)$ is the value of x for the i^{th} cast.

In order to pity his poor classmate RB, which also attempted N courses in this semester and achieved score B_i score in the i^{th} course, Lord Manholition wants that, after casting a sequence of spells followed by sorting both his and RB's scores in ascending order, his scores will be the same as RB's. Can you help Lord Manholition to find a sequence of spell casts to fulfill his strange goal?

INPUT

The first line contains an integer N -- the number of courses.

The second line contains N integers, $A_1, A_2, \ldots, A_N$. $(1 \le A_1 \le A_2 \le \cdots \le A_N)$

The third line contains N integers, $B_1, B_2, \ldots, B_N$. $(1 \le B_1 \le B_2 \le \cdots \le B_N)$

OUTPUT

If there is no solution, output `Impossible` in a single line.

Otherwise, output a sequence of spell casts: The first line should contain a non-negative integer K $(0 \le K \le 5 \times 10^6)$, the number of spell casts. Then output K lines, the i^{th} line of which contains the positive integer X_i. It is guaranteed that if a solution exists, there exists a solution with $K \le 5 \times 10^6$.

SAMPLE TESTS

Input **Output**

1

```
2
3 4
1 2
```

```
4
1
1
1
1
```

25%: A correct sequence but is not of minimal length.

2

```
2
3 4
1 2
```

```
2
2
2
```

50%: The sequence is one of the minimal length sequences but not the lexicographically largest one.

3

2
3 4
1 2
	2
	3
	1

100%: The sequence is the lexicographically largest one among all minimal length sequences.

4

1 2 3	Impossible

SUBTASKS

For all cases: $1 \leq N \leq 100000$, $A_N, B_N \leq 10^9$

	Points	Constraints
1	15	$1 \leq N \leq 3$
2	30	$1 \leq N \leq 8$
3	25	$1 \leq N \leq 5000$
4	30	No additional constraints

SCORING

Within a subtask:

- If your program wrongly outputs `Impossible` when a solution exists, or fails to output `Impossible` when there is no solution, you lose all points in the subtask.
- If for each and every test case that has a solution, your program outputs the correct sequence of minimal length and lexicographical largest one among all the possible sequences with minimal length, you score 100% in the subtask. Sequence $A_1, A_2, \ldots, A_K$ is lexicographically larger than sequence $B_1, B_2, \ldots, B_K$ if and only if there exists i such that $A_i > B_i$ and $A_j = B_j$ for all $j < i$.
- Otherwise if for each and every test case that has a solution, your program outputs any correct sequence of minimal length, you score 50% in the subtask.
- Otherwise if for each and every test case that has a solution, your program outputs any correct sequence, you score 25% in the subtask.
- Otherwise, you lose all points in the subtask.

S173 - MONSTER GO

Time Limit: 1.000 s / Memory Limit: 256 MB

Alice is a monster trainer in Byteland. Her target is to catch all the monsters in Byteland. Unfortunately, she has caught none of them.

There is a total of N distinct monsters in Byteland and they live in N different caves individually. The caves are numbered from 1 to N. To help this poor trainer, Dr. Jones gives her access to his M radars. The i^{th} radar can detect the set of monsters living in L_i caves and output the set of monsters in an arbitrarily ordered list.

Before catching the monsters, it is necessary for Alice to know the locations of each monster. One of the way is to deduce their locations from the scanning results of Dr. Jones' radars. Help her to determine whether the radars are sufficient to deduce the locations of each monster.

More specifically, let $(p_1, p_2, \ldots, p_N)$ be a permutation of $(1, 2, \ldots, N)$ such that the i^{th} monster lives in the cave p_i. The radars are said to be sufficient if exactly one permutation $(p_1, p_2, \ldots, p_N)$ could satisfy the scanning results from the radars for all possible scanning results.

If the radars are sufficient, you should also find the minimum integer K such that the first K radars are already sufficient.

INPUT

The first line contains two integers, N and M, the number of monsters and number of radars respectively.

The i^{th} line of the next M lines describes the i^{th} radar. The first integer on the i^{th} line is L_i $(1 \le L_i \le N)$, the number of caves that the i^{th} radar can scan. The following L_i distinct integers on that line $C_{i,1}, C_{i,2}, \ldots, C_{i,L_i}$ $(1 \le C_{i,j} \le N$ for $1 \le j \le L_i)$ are the caves that the i^{th} radar can scan.

OUTPUT

If the M radars are sufficient, print the minimum integer K such that the first K radars in the input are already sufficient.

Otherwise, output `Impossible`.

SAMPLE TESTS

	Input	Output

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

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

3	6 3
	2 1 2
	2 1 3
	3 2 3 4
		Impossible

SUBTASKS

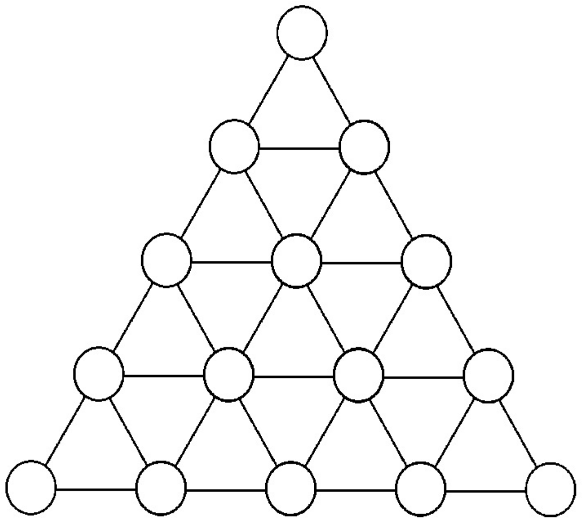
For all cases: $1 \leq N, M \leq 200000$, $\sum_{i=1}^{M} L_i \leq 200000$

	Points	Constraints
1	9	$1 \leq N, M \leq 2$
2	9	$1 \leq N \leq 2$ $N \times M \leq 200000$
3	22	$1 \leq N \leq 10$ $N \times M \leq 200000$
4	23	$N \times M \leq 200000$
5	37	No additional constraints

S174 - MAGIC TRIANGLE II

Time Limit: 0.500 s / Memory Limit: 256 MB

Consider a triangular grid with N layers. It has $\frac{N \times (N+1)}{2}$ vertices and its side length is $N-1$. The figure below shows a triangular grid with $N = 5$ layers.



Initially, a positive integer is written on each vertex of the grid. You want to make the triangular grid K-magical. To check whether a triangular grid is K-magical, we consider all triangles with side length K and edges parallel to the grid lines. For such a triangle, we consider the sum of numbers on its three vertices. The grid is considered to be K-magical if, for any two such triangles T_1 and T_2, the sum of numbers on the vertices of T_1 is the same as that on the vertices of T_2.

In case the given grid is not K-magical, you can modify the numbers on the vertices to make it so. You can either increase the number in a vertex by 1 with cost A, or decrease the number in a vertex by 1 with cost B. You can perform the two types of operations any number of times, but after each operation, all numbers must remain positive.

Your task is to find the minimal cost to make a given triangular grid K-magical. Output the cost, as well as the final, K-magical configuration.

INPUT

The first line of input consists of four integers N, A, B, and K.

For the next N lines, the i^{th} line consists of i integers, representing the initial numbers on the i^{th} layer, from left to right. The initial numbers are between 1 and 128 (inclusive).

OUTPUT

Output $N+1$ lines in total.

On the first line, output a single integer, the minimal cost to make the grid K-magical.

On the next N lines, output a final configuration of the grid, using the same format as that for the input. All numbers on the vertices must be between 1 and 512 (inclusive).

SAMPLE TESTS

Input **Output**

1

```
5 1 10 3
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

```
15
1
2 3
4 5 6
7 8 9 22
11 12 13 17 15
```

$1 + 7 + 22 = 2 + 11 + 17 = 3 + 12 + 15 = 30$

2

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

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

SUBTASKS

For all cases: $1 \leq K < N \leq 80$, $1 \leq A, B \leq 50$

	Points	Constraints
1	11	$N = 3$ $K = 1$ $A = B = 1$
2	15	$N = 3$ $K = 1$
3	10	$N \leq 6$ $K = 1$
4	18	$K = 1$
5	25	$N \leq 10$
6	21	No additional constraints

SCORING

Within a subtask:

- If for each and every test case, your program outputs the correct minimal cost and a minimal-cost K-magical final configuration, you score 100% in the subtask.
- Otherwise if for each and every test case, your program outputs the correct minimal cost and any final configuration in the correct format, you score 60% in the subtask.
- Otherwise, you lose all points in the subtask.