Problem A. Knot Knowledge

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Sonja the scout is taking a test to see if she knows all the knots a scout is supposed to know. The Scout's Big Book of Knots has descriptions of 1 000 different knots, conveniently numbered from 1 to 1 000. For the test, Sonja needs to learn a specific set of n of these knots. After some intense studying, she has learned all except one of them, but she has forgotten which knot she does not yet know.

Given the list of knots Sonja needs to learn, and the ones she has learned so far, find the remaining knot to learn.

Input

The first line of input consists of an integer n ($2 \leq n \leq 50$), the number of knots Sonja needs to learn. This is followed by a line containing n distinct integers $x_1, \ldots, x_n$ ($1 \leq x_i \leq 1\,000$), the knots that Sonja needs to learn. Finally, the last line contains $n-1$ distinct integers $y_1, \ldots, y_{n-1}$ ($1 \leq y_i \leq 1\,000$), the knots that Sonja has learned so far. You may assume that each knot Sonja has learned is one of the n knots she was supposed to learn.

Output

Output the number of the remaining knot that Sonja needs to learn.

Examples

standard input	standard output
4 1 2 4 3 4 2 3	1
4 10 101 999 1 1 999 101	10

Source

Nordic Collegiate Programming Contest 2021. Rephrased.

Problem B. Hot Springs

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Iceland is famous for its geothermal activity, which supplies the country with much of its electricity and heating. It is also a source of pleasure, in the form of hot springs.

Kalle is visiting one of Iceland's famous hot springs. It contains n pools of water, where the ith one has temperature t_i. Although staying in one of the warmer pools for a long time sure is relaxing, Kalle is on a very tight schedule and just wants a quick dip in each of the pools. As you may know, the nicest thing about hot baths is the contrast between hot and cold. Therefore, to get the most out of his stay, Kalle wants to find an ordering of the pools so that the difference in temperature between subsequent pools is increasing.

Given a sequence of pool temperatures $t_1, t_2, \ldots, t_n$, rearrange them into a new sequence $t'_1, t'_2, \ldots, t'_n$ such that for all $2 \leq i \leq n-1$ it holds that

$$|t'_{i-1} - t'_i| \leq |t'_i - t'_{i+1}|.$$

Input

The input consists of:

- One line with an integer n ($2 \leq n \leq 10^5$), the number of pools.
- One line with n integers $t_1, \ldots, t_n$ ($-10^5 \leq t_i \leq 10^5$ for each i), the temperatures in each of the n pools.

Output

Output a rearrangement of the sequence satisfying the given requirement. If no solution exists, output "`impossible`". If there are multiple valid solutions, you may output any one of them.

Examples

standard input	standard output
3 1 3 4	4 3 1
6 0 0 1 -1 -6 3	0 1 3 -1 -6 0

Source

Northwestern Europe Regional Contest 2020. Rephrased.

Problem C. Atomic Energy

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

The *Next Wave Energy Research Club* is looking at several atoms as potential energy sources, and has asked you to do some computations to see which are the most promising.

Although an atom is composed of various parts, for the purposes of this method only the number of neutrons in the atom is relevant[1]. In the method, a laser charge is fired at the atom, which then releases energy in a process formally called *explodification*. Exactly how this process proceeds depends on the number of neutrons k:

- If the atom contains $k \leq n$ neutrons, it will be converted into a_k joules of energy.
- If the atom contains $k > n$ neutrons, it will decompose into two atoms with i and j neutrons respectively, satisfying $i, j \geq 1$ and $i + j = k$. These two atoms will then themselves explodificate.

When an atom with k neutrons is explodificated, the total energy that is released depends on the exact sequence of decompositions that occurs in the explodification process. Modern physics is not powerful enough to predict exactly how an atom will decompose—however, for explodification to be a reliable energy source, we need to know the minimum amount of energy that it can release upon explodification. You have been tasked with computing this quantity.

Input

The input consists of:

- One line with two integers n and q ($1 \leq n \leq 100$, $1 \leq q \leq 10^5$), the neutron threshold and the number of experiments.
- One line with n integers $a_1, \ldots, a_n$ ($1 \leq a_i \leq 10^9$ for each i), where a_i is the amount of energy released when an atom with i neutrons is explodificated.
- Then q lines follow, each with an integer k ($1 \leq k \leq 10^9$), asking for the minimum energy released when an atom with k neutrons is explodificated.

Output

For each query k, output the minimum energy released when an atom with k neutrons is explodificated.

[1] In fact, for this problem you might want to forget everything you thought you knew about chemistry.

Examples

standard input	standard output
4 5 2 3 5 7 2 3 5 6 8	3 5 8 10 13
1 3 10 1 2 100	10 20 1000

Source

Northwestern Europe Regional Contest 2020. Rephrased.

Problem D. Antenna Analysis

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

Åke has heard that there may be some suspicious radiation in his city. To test this, he uses the antenna on his roof to measure the radiation level each day. However, he does not know how he should analyze the data.

We are given the measurements for n consecutive days as a list of numbers $x_1, \ldots, x_n$ (where x_i denotes the measurement for day i) and a constant c that measures how much Åke expects the radiation to vary from day to day. We want to find, for each day i, the most significant difference between the measurement on day i and any earlier day, after the expected variations are taken into account. More precisely, the goal is to find the maximum value of

$$|x_i - x_j| - c \cdot |i - j|$$

where $j \leq i$. I.e., we want to find a large difference in radiation level that has happened recently.

Input

The first line of input contains the two integers n and c ($1 \leq n \leq 4 \cdot 10^5$, $1 \leq c \leq 10^6$), the number of measurements and expected day-to-day variation. The second input line contains the n integers $x_1, x_2, \ldots, x_n$ ($1 \leq x_i \leq 10^6$ for $i = 1, 2, \ldots, n$), giving the measurements of the n days.

Output

Output n integers $y_1, \ldots, y_n$, where y_i is the most significant difference on day i.

Example

standard input	standard output
5 1 2 7 1 5 4	0 4 5 3 1

Source

Nordic Collegiate Programming Contest 2021. Rephrased.

Problem E. Customs Controls

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

With lifted restrictions, the border trade between Norway and Sweden will surely be back to its former glory. But the authorities are worried that this will also mean an increase of illegal smuggling of goods. The customs authorities of Norway and Sweden must cooperate to prevent this from becoming too big of a problem.

To pass through the customs, one must visit a series of checkpoints, the Nordic Customs and Passport Control. There are n checkpoints in total, numbered from 1 to n, where 1 is the entrance and n is the exit. There are m pairs of bidirectional roads that connect distinct checkpoints. The ith checkpoint takes some amount of time t_i to pass through, and this is the bottleneck in crossing the border (the time it takes to walk the roads is negligible).

Each checkpoint can be watched by one customs unit, either a Norwegian one or a Swedish one. There are k Norwegian customs units available, and $n-k$ Swedish units. When a road has both of its endpoints watched by customs units from the same country, any smugglers using that road will be caught. Smugglers are of course always in a hurry, and will always attempt to go from 1 to n in as short amount of time as possible.

Your task is to decide where to put the n customs units, so that any smugglers who take a fastest possible route from 1 to n will be caught.

Input

The first line of input contains three integers n, m, and k ($2 \le n \le 10^5$, $1 \le m \le 2 \cdot 10^5$, $0 \le k \le n$), the number of checkpoints, roads, and Norwegian customs units. The second line of input contains n positive integers $t_1, \ldots, t_n$ ($1 \le t_i \le 10^4$), the time it takes to pass through each checkpoint. Then follow m lines of input each containing two integers u and v ($1 \le u, v \le n$), meaning that there is a road between checkpoints u and v.

It is guaranteed that it is possible to go from any checkpoint to any other checkpoint using the roads. There is also at most one road between each pair of checkpoints, and no road connects a checkpoint to itself.

Output

If there is a way to place the customs units so that every smuggler is caught, output a string of length n, where the ith character indicates which type of customs unit to put at the ith checkpoint (an '`N`' for a Norwegian customs unit, and an '`S`' for a Swedish customs unit). Otherwise, if there if there is no way to catch every smuggler, output "`impossible`".

Examples

standard input	standard output
3 2 0 1 1 1 1 2 2 3	SSS
2 1 1 1 1 1 2	impossible
8 9 4 3 3 1 2 2 3 2 1 1 2 1 3 1 4 2 5 3 6 4 7 5 8 6 8 7 8	SNSNSSNN

Source

Nordic Collegiate Programming Contest 2021. Rephrased.

Problem F. Great Expectations

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

A speedrun is a playthrough of a game with the intention to complete it as quickly as possible. When speedrunning, you usually follow a pre-planned path through the game. Along this path, there may be some places where you have to pull off a difficult technique, or trick, which may cause a delay if you fail to pull it off successfully. Luckily you can *reset* the game at any time: if you have made a few mistakes, you can start a new run, losing your progress but instantaneously starting over with a clean slate. You can do this as often as you like.

The game you are currently speedrunning has a record of r seconds, which you intend to beat. You have discovered a path through the game that, in the best case, takes $n < r$ seconds. There are some tricks along the way, though: you know exactly where along the run they occur, what the probability is that you will pull them off successfully, and how many seconds you have to spend to recover if they fail.

Given this data, you want to find the optimal strategy for when to reset the game to minimise the expected time to set a new record. Write a program to determine what this smallest possible expected time is.

Input

The input consists of:

- One line with three integers n, r and m ($2 \leq n < r \leq 5\,000$, $1 \leq m \leq 50$), where n and r are as described above and m is the number of tricks.
- m lines, each containing three numbers describing a trick:
  - An integer t ($1 \leq t < n$), the time in the route (assuming no failed tricks before) at which the trick occurs,
  - a real number p ($0 < p < 1$ and p has at most 6 digits after the decimal point), the probability that the trick succeeds, and
  - an integer d ($1 \leq d \leq 1\,000$), the number of seconds required to recover in case the trick fails.

The tricks are given in sorted order by t, and no two tricks occur at the same time t in the route.

You may assume that, without resetting, a single playthrough has a probability of at least 1 in 50 000 to succeed at improving the record.

Output

Output the expected time you will have to play the game to set a new record, assuming an optimal strategy is used. Your answer should have an absolute or relative error of at most 10^{-6}.

Examples

standard input	standard output
100 111 5 20 0.5 10 80 0.5 2 85 0.5 2 90 0.5 2 95 0.5 2	124
2 4 1 1 0.5 5	3
10 20 3 5 0.3 8 6 0.8 3 8 0.9 3	18.9029850746
10 50 1 5 0.5 30	15

Note

Explanation of Sample Input 1

The record for this game is 111 seconds, and your route takes 100 seconds if everything goes right.

After playing for 20 seconds, there is a trick with a 50% success rate. If it succeeds, you keep playing. If it fails, you incur a 10 second time loss: now the run will take at least 110 seconds. It is still possible to set a record, but every other trick in the run has to be successful. It turns out to be faster on average to reset after failing the first trick.

Thus you repeat the first 20 seconds of the game until the trick is successful: with probability 1/2, it takes 1 attempt; with probability 1/4, it takes 2 attempts; and so on. On average, you spend 40 seconds on the first 20 seconds of the route.

Once you have successfully performed the first trick, you want to finish the run no matter the result of the other tricks: it takes 80 seconds, plus on average 1 second loss from each of the remaining 4 tricks. So the expected time until you set a record is 124 seconds.

Source

Northwestern Europe Regional Contest 2020. Rephrased.

Problem G. Elevator Pitch

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

You are in charge of ensuring all building designs meet accessibility requirements. As law dictates, every part of your building should be reachable for wheelchair users, which means elevators will have to be installed. You are given the blueprints of the company's current project and have to determine the minimum number of elevators required.

The floor plan is laid out on a square grid and the blueprints tell you the number of floors above any given square. You can place an elevator at any square, which stops at all floors of that square. A wheelchair user can move up and down between floors using the elevators and can freely move to any of the four adjacent squares on the same floor. Buildings do not connect diagonally.

The image below shows the second sample input. Designs can consist of multiple buildings; this one contains three buildings. The design requires two elevators: one for the pyramid-shaped building and one for the tall tower. The small building of height one does not require an elevator, since it only has a ground floor.

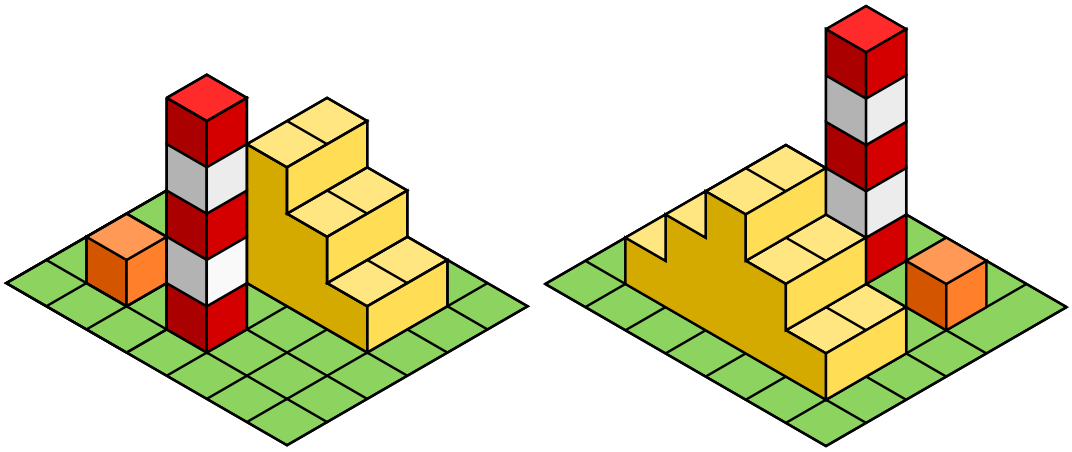

Рис. 1: A visualisation of the second sample input.

Input

- One line containing integers h and w ($1 \leq h, w \leq 500$), the height and width of the grid.
- h lines of w integers each, where $x_{i,j}$ ($0 \leq x_{i,j} \leq 10^9$), the jth integer on the ith line, denotes the number of floors at position (i, j) of the grid.

Output

Output the minimum number of elevators you need to build to be able to reach every part of the building(s) in the grid.

Examples

standard input	standard output
3 3 1 2 3 0 0 4 7 6 5	1
6 7 0 0 0 0 0 0 0 0 1 2 3 2 1 0 0 1 2 3 2 1 0 0 0 0 0 0 0 0 0 1 0 5 0 0 0 0 0 0 0 0 0 0	2
4 4 1 1 2 1 2 2 1 2 1 2 2 1 2 1 2 2	4

Source

UK & Ireland Programming Contest 2020. Rephrased.