Problem 1. Ski race

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	1 second
Memory limit:	256 megabytes

Winter has come to the town of N, and it's time for the first cross-country skiing race. This year, participants registered through the Internet — they entered their data, and each picked a number which had not yet been picked by other skiers. Due to the high number of participants, the organizers decided to split the race into several starts. To pick the lucky skiers for the first start, they've come up with a simple rule — the skier with the number X comes to the start if no other skier's number is divisible by X.

Help the organizers write a program to define the numbers of those who will start first.

Input

The first line of the input file contains an integer K — the number of registered participants ($1 \leq K \leq 10^5$). The second line contains K space-separated integers A_i — the numbers chosen by the participants at the registration ($1 \leq A_i \leq 10^7$). All the numbers A_i are distinct.

Output

The output file must contain a single line containing the numbers of all participants starting first, in the ascending order. Numbers must be space-separated.

Example

input.txt	output.txt
3 4 8 12	8 12

Problem 2. Chairs

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	1 second
Memory limit:	256 megabytes

Ostap and Kisa found themselves at a chair sale. They are facing two problems. First, they must leave the sale as soon as possible, because their rival company, represented by father Theodore, is breathing down their necks; second, they must get all chairs at the sale.

The sale site is a rectangular table of N rows and M columns, with some of its cells occupied by chairs that need to be collected. Initially our enterpreneurs are in the top-left corner — the cell with the coordinates $(1, 1)$, and the exit is located in the bottom-right corner — the cell with the coordinates (N, M). A single move can transfer them from a given sell to any of adjacent by a side cells. Passing through a cell with a chair, they take the chair with them. Find a shortest path from the starting to the ending cell. Among all such paths, find one passing through all cells with chairs, or find out if such a path doesn't exist.

Input

The first line contains three integers: N, M, K — size of the table and the number of chairs, respectively ($2 \le N, M \le 100$, $0 \le K \le 1000$).

The following K lines of the input data each contain two integers: X_i — the number of the row in which the i-th chair is located, and Y_i — the number of the column in which the i-th chair is located ($1 \le X_i \le N$, $1 \le Y_i \le M$). A single cell cannot contain multiple chairs.

Output

If collecting all chairs along any single shortest path is impossible, print a single word «`Impossible`» (without brackets) in the output file.

If such a path exists, print it as a line containing the sequence of moves, with each move coded by a single symbol according to the following:

- `R` — move right;
- `D` — move down;

If several solutions are possible, print the lexicographically smallest solution.

Example

input.txt	output.txt

3 3 2
1 2
3 3
	RDDR

3 3 2
1 2
2 1
	Impossible

Problem 4. Wires

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	1 second
Memory limit:	256 megabytes

Employees of a very large and very secret agancy work in a large rectangular room. N employees from the first division are seated by the windows of one wall and the same number of employees from the second division are seated along the opposite wall. One day, a very important and very secret memo came in — computers of all employees of the two divisions were to be connected in such a manner that each first division employee's computer were linked with the corresponding second division employee's computer with a separate wire.

A technical assignment including the room blueprints was drafted. This blueprints showed the room as a A by B rectangle: its left and right sides are of the length A, and the top and bottom sides are of the length B. There are N input contacts on the left wall, corresponding to positions of computers of the first division employees, and N output contacts on the right side for computers of the second division employees. Connect each input with the corresponding output by a wire based on the mutually unambiguous input-output correspondence.

There are rules regarding wires:

1. Wires cannot fork, i.e. each wire begins at the input contact and ends at the output contact.
2. Each wire can pass both inside and outside the rectangle (all contacts are accessible from both sides of the rectangle wall).
3. A wire **cannot** cross the rectangle wall.
4. Wires **cannot** intersect with each other, i.e. a wire cannot go above another one.

Find the minimum total length of wire necessary to connect the contacts in the desired manner, if it is possible. The wire thickness can be considered negligibly small: wires can pass infinitely close to each other.

Write a program which calculates the minimum required length of wires.

Input

The first line of the input file contains three integers: A — the length of the left side of the rectangle, B — the length of the upper side of the rectangle and N — the number of input (and output) contacts ($1 \le A, B \le 10^8$, $1 \le N \le 10^5$).

The second line describes the positions of all N input contacts. For each k-th input number an integer L_k is given — the distance from the lower left corner of the rectangle to the contact ($0 \le L_k \le A$). It is guaranteed that all L_k are different.

The third line contains the positions of N output contacts. For each k-th output contact an integer R_k is provided — the distance from the lower right corner to the contact ($0 \le R_k \le A$). It is guaranteed that all R_k are different.

Connect each k-th (description-wise) input contact with the k-th (description-wise) output contact.

Output

The first line of the output file must contain a single real number — the minimum total length of all wires in a correct connection scheme. The absolute or relative error must not be greater than

10^{-9}.

If there are no correct ways to do the wiring, print the number -1.

Example

input.txt	output.txt

6 7 3
1 3 5
5 1 3
	27.560219778561036542194604983054

Commentary

Strictly mathematically speaking, the minimum total length of wires may fail to be achieved with any of the correct wiring plans due to the infinitely small thickness of wires. In this case, find the precise lower margin(infimum) of all possible total lengths.

Problem 5. Voting

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds (3 seconds for Java)
Memory limit:	256 megabytes

The political situation in Berland has changed. With the opposing party candidate having won the election, the multi-level voting system has finally been canceled. Now the president of Berland is elected by a single total voting. But the conservative zealots are busy peddling the idea to the masses that the new voting is even more prone to tampering with the results than ever before. To refute these calumnies, the president requested to evaluate the costs of fixing voting results by bribing voters.

There is a total of N voters and K candidates. Each of the voters can either cast his voice for a single candidate or abstain from voting, for example, by not going to the election. Once all voters have voted (one way or another), the number of voices collected by each candidate is counted. The candidate who gets strictly the most voices wins. If there is no such candidate, the elections are deemed null and void.

You are asked to write a program based on the following statements. For each individual voter the candidate for whom he or she is going to vote is known. It is allowed to change the voter's preference to any other variant by spending a certain sum of money. The goal is for the necessary candidate to win the elections. Minimize the amount of money necessary to complete this task.

Input

The first line of the input file contains three integers: N — the count of voters in Berland, K — the count of candidates running for presidency, T — the index of candidate who needs the elections fixed in his favor ($1 \le N \le 100$, $1 \le K \le 10$, $1 \le T \le K$). Both all voters and all candidates are numbered in succession beginning from the number one.

This is followed by a costs matrix of N lines and $K + 1$ columns. The element $C_{i,j}$ of the matrix defines the amount of money to be spent in order to assure that the i-th voter votes for the j-th candidate (with $1 \le i \le N$, $1 \le j \le K$). The last element $C_{i,K+1}$ in the line defines the amount of money to be spent in order to keep the voter from going to the elections.

It is guaranteed that all costs $C_{i,j}$ are integers and fall within the range of $0 \le C_{i,j} \le 10^9$. In addition, for each i strictly one of the numbers $C_{i,1}, C_{i,2}, \ldots, C_{i,K+1}$ equals zero: the zero means that the given voter has been planning to vote in the corresponding way.

Output

The first line of the output file must contain a single integer — the minimum required amount of money to be spent in order to change the voters' preferences.

The second line of the input file must contain N integers. The i-th of these numbers V_i means that the i-th voter must vote for the V_i-th candidate($1 \le V_i \le K + 1$). The special value $V_i = K + 1$ means that the i-th voter must skip the elections.

If there are several optimal solutions, print any of them.

Example

input.txt	output.txt

5 2 2
0 9 2
0 10 1
7 8 0
0 2 1
3 0 1
	3
	1 3 3 2 2

Example explanation

The example suggests that the second voter's preference must be changed to skipping the elections (costing 1 unit of money), and that the fourth voter must be persuaded to vote for the desired candidate (costing 2 units). As the result, only the first voter is going to vote for the first candidate, with the fourth and fifth voters voting for the second candidate.

Problem 6. Finite automaton

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 second
Memory limit:	256 megabytes

Today Vasya learned what a «deterministic finite automaton» (DFA) is, and he's aching to tell everyone about it.

As it turns out, there are N states in a DFA. The automaton can be in any one of these state at any given moment during its work. The input of the automaton is an arbitrary string, and after its work, the automaton tells whether the string is *acceptable*.

The automaton works in the following manner:

1. In the beginning of its work, the automaton is in the *start* state, which is always marked as such.
2. The automaton *reads* all symbols of the string one by one from left to right. After reading each symbol, the automaton can switch to a different state (described in detail below).
3. After the string is read completely, the automaton defines the answer based on the state in which it ended up.

For each state u of the automaton and each possible symbol c, automaton defines in which state will it be afer reading the symbol c, if it was in the state u beforehand. This new state can either be the same state u or be a different state. Moreover, for each state the automaton defines the answer it will give (whether the string is accepted or not) if it finished in that state.

In the first seminar on the subject, Vasya constructed all sorts of DFA's, and he was given the following problem for «homework». Build a DFA which, given a non-negative integer written in the B-ary numeral system, accepts those and only those integers that are divisible by the given module M.

To simplify, Vasya assumes that the input number:

- begins from high-order digits (they're written on the left, big-endian);
- can have leading zeroes;
- can be empty: in this case it is equal to zero and is definitely divisible by M.

Vasya is a born perfectionist, and he wants to learn how to build DFA's that meet the problem requirements with the **smallest** possible number of states, He asked you to help him.

Input

The only line of the input file contains two integers: B — the base of the positional numeral system in which the input number is given and M — the module which all acceptable and only acceptable numbers must be divisible by ($2 \le B \le 16, 2 \le M \le 10^5$).

Output

Print the description of the smallest DFA meeting the problem requirements to the output file.

The first line of the output file must contain two integers: N — the count of states in the automaton ($N \geq 2$) and S — the start state number ($0 \leq S < N$). All states are numbered successively beginning from zero.

The second line must contain N space-separated symbols. The k-th of these symbols defines the answer the automaton gives if it ends up in the k-th state upon the completion of its work ($0 \leq k < N$). A symbol equals '`G`' if the string should be deemed acceptable, and '`B`' otherwise.

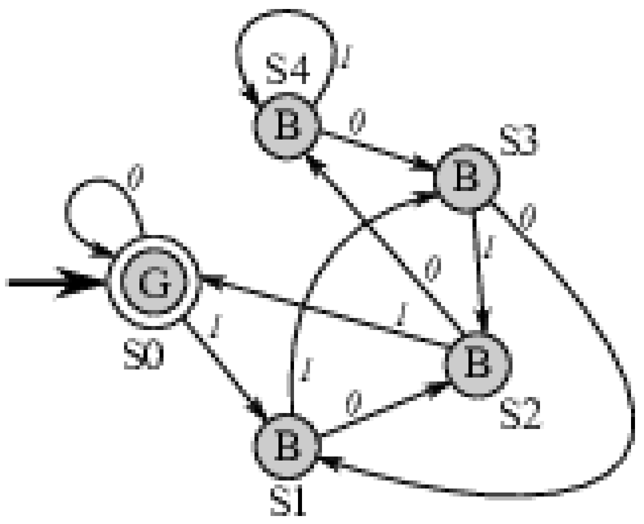
There must be N line following, each containing B integers. The k-th number in the i-th of these lines contains the state number in which the automaton ends up after reading digit k, if before that it was in the state i ($0 \leq i < N$, $0 \leq k < B$). This number can be any integer between 0 and $N - 1$ inclusively.

Example

input.txt	output.txt
2 5	5 0 G B B B B 0 1 2 3 4 0 1 2 3 4

Example explanation

Shown below is the DFA used in the sample. The letter "`S`" and the number are shown near each state. Bold arrow points to the start state. Each normal arrow describes the state the DFA goes to after reading the digit written near the arrow.



Problem 8. A system of balance scales

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	1 second
Memory limit:	256 megabytes

A complicated system of balance scales and weights is set out on the floor. Each set of balance scales consist of a stand, a beam and two cups. The beam is **not** fixed and rests on the stand with a single point in such a manner that it can rotate freely around it in the vertical plane. Precise bearing point selection allows a state of nonstable equilibrium in horizontal position. Cups are attached to the beam ends, and they usually hold objects whose weights are to be compared. The distance from the bearing point to the cup is called a shoulder. The rule states that upon reaching the equilibrium shoulders correlate in the same way as do the weights of the objects in the cups. The system of balance scales and weights works in the following way: Each of the cups of all scales holds either a balance weight or another pair of balance scales. There is strictly one pair of balance scales standing directly on the floor, with all other scales standing in cups of other scales. The weights of all balance weights are known, and the weight of the scales themselves are negligibly small compared to the balance weights. All scales are always in the state of equilibrium owing to correct choice of bearing points. The sizes of balance weights, cups and stands are also negligibly small compared to the length of beams.

Process a sequence of queries of two types.

1. Change the weight of a given balance weight.

2. Learn the position of the bearing point of a given pair of scales.

After each weight change of any balance weight equilibrium must be restored in all scales in the system, with some of its bearing points shifting in the process.

Input

The first line contains two integers: N — the number of scales in the system($1 \le N \le 5 \cdot 10^4$) and K — the number of queries ($1 \le K \le 10^5$).

All scales are numbered with integers beginning from one and up. Scales with the number 1 stand on the floor. All balance weights are also numbered with integers beginning from one and up.

The second line contains $(N + 1)$ integers: the t-th of these numbers W_t defines the initial weight of the balance weight with the number t ($1 \le W_t \le 10^9$).

The following N line describe the scales. The i-th of these lines contains three integers: S_i — length of the beam of the i-th pair of scales ($1 \le L_i \le 10^4$), L_i — number of the scales standing in the left cup of the i-th scales and R_i — number of the scales standing in the right cup of the i-th scales. If the left cup is occupied by scales, then $i < L_i \le N$, if it is occupied by a balance weight, then L_i equals the number of the balance weight with a «minus» sign, with $1 \le -L_i \le N + 1$. Similarly, R_i defines either the number of the standing scales ($i < R_i \le N$) or the number of the balance weight with a «minus» ($1 \le -R_i \le N + 1$).

This is followed by K lines, with every j-th line containing a single query. The query description begins with an integer t_j, defining the query type ($1 \le t_j \le 2$). If $t_j = 1$, it is followed by two integers: k_j — number of the balance weight with its weight being changed ($1 \le k_j \le N+1$), V_j — new weight of the balance weight($1 \le V_j \le 10^9$). If $t_j = 2$, it is followed by a single integer k_j — the number of scales for which the position of its bearing point must be found ($1 \le k_j \le N$). There are no other types of queries.

Output

For each query to define the bearing point a single integer must be printed: the distance from the left cup of the scales to the bearing point. Answers must be produced in the order of occurence of the corresponding queries in the input data. The absolute or relative error of each answer should be less than or equal 10^{-13}.

Example

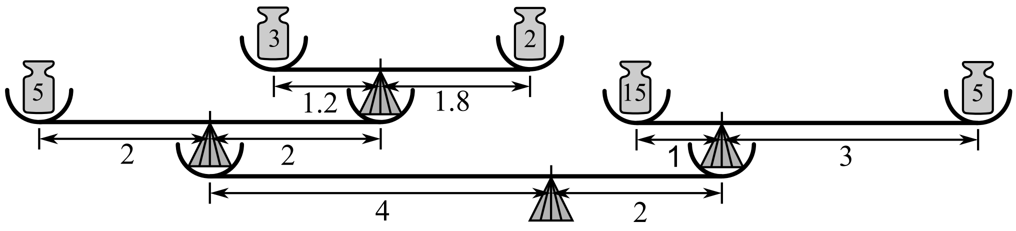
input.txt	output.txt

`1 1`
`15 20`
`10 -1 -2`
`2 1`
	`5.7142857142857142857142857142857`

`4 9`
`3 2 5 5 15`
`6 2 3`
`4 -3 4`
`4 -5 -4`
`3 -1 -2`
`2 1`
`2 2`
`2 3`
`2 4`
`1 4 45`
`2 3`
`1 5 45`
`2 3`
`2 1`
	`4`
	`2`
	`1`
	`1.2`
	`3`
	`2`
	`5.4`

Example explanation

Initial state of the system from the second sample:



Problem 9. Karmon be ill

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	1 second
Memory limit:	256 megabytes

Vasya loves catching karmons. Every karmon has a numerical parameterBP (battle power). The larger the parameter value, the stronger the karmon — and the more valuable.

Vasya's feeling ill and he's asked his friend Peter to go karmon trapping instead of him. Peter has agreed, but Vasya also asked for another thing: he wants Peter to report the BP sum of the K most powerful currently caught karmons every time Peter catches another karmon. Peter has found this request a bit weird, but there's nothing he wouldn't do for an ill friend. Nevertheless he decided to clarify things and asked Vasya what should be done if he hasn't yet collected K karmons. Vasya gave it a thought and decided that in this case, Peter shouldn't report anything at all.

Help Peter write a program that is fed a list of BP's of the caught karmons and produces the values to be reported to Vasya.

Input

The first line contains two integers: N — the total number of the trapped karmons and K — the number of karmons for which the sum of their BP must be reported ($10 \le N \le 100\,000$, $2 \le K \le \min(N, 1000)$).

The second line contains N integers defining the karmons' BP in order of catching. All these numbers lie within the range of 1 to 10 000 inclusively.

Output

The single line of the output file must contain $(N - K + 1)$ integers — the sums of BP for the K most powerful karmons after catching each karmon (beginning from the K-th one).

Example

input.txt	output.txt
14 4 1 2 3 4 5 6 7 8 9 10 1 1 1 1	10 14 18 22 26 30 34 34 34 34 34

Problem 11. Test generation

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

A while ago Pasha came up with a simple problem for a programming contest training session. The input data in the problem consists of the line S containing N digits, and three integers L, R and P ($1 \leq L \leq R \leq N$, P being a prime number). The requested output was the remainder from the division of the subnumber formed by digits at the positions from L through R, inclusively, by the number P. It should be noted that this subnumber may contain leading zeroes. Pasha prepared the problem description, wrote a solution and prepared lots of tests to check the solutions.

Before a practice session, Pasha discovered that T files with input test data were gone, and only the corresponding answer files remained. He remembers that the line S in all these tests was identical, moreover, he remembers that line perfectly well. Similarly, he remembers the value of P, which was also identical in all missing tests. To recover the lost input data, Pasha is asking for your help. Write a program which is given a line S of the length N, the numbers P and T, as well as T values of A_i — the answers for the lost test data. For each A_i, the program must figure out the number of different pairs $\{L_i, R_i\}$ ($1 \leq L_i \leq R_i \leq N$) — the pairs of acceptable values from the input file, as well as find one of these pairs.

Input

The first line of the input file contains the line S, consisting of N decimal digits ($1 \leq N \leq 10^5$). The second line of the input data contains two integers T and P — the number of the lost tests and the prime number, for which the remainder from the division by that number was to be calculated.($1 \leq T \leq 100$, $11 \leq P \leq 10^9 + 33$, P — the prime number). This is followed by T lines, with the i-th line containing a single integer A_i — the answer for the i-th test input dataset ($0 \leq A_i < P$).

Output

For each of these T solutions, the output file must receive three integers C_i, L_i and R_i — the number of different acceptable pairs of input values, and the values of one of those pairs, respectively. If Pasha has made an error when preparing the tests, and there are no acceptable pairs for a solution, three zeroes must be printed.

Example

input.txt	output.txt

923813
4 17
5
3
15
13
	2 1 3
	3 3 3
	0 0 0
	3 4 5

Problem 12. Tournament

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds
Memory limit:	256 mebibytes

Each player in a tournament plays six games. There are no ties. After those games players are distributed between groups based on the results of games as follows:

- if a player wins more than 4 games, they are placed in Group 1;
- if a player wins 4 or 3 games, they are placed in Group 2;
- if a player wins 2 or 1 games, they are placed in Group 3;
- if a player does not win any games, they are eliminated from the tournament.

Write a program to determine which group a player is placed in.

Input

The input consists of six lines, each with one of two possible letters: '`W`' (to indicate a win) or '`L`' (to indicate a loss).

Output

The output will be either 1, 2, 3 (to indicate which Group the player should be placed in) or −1 (to indicate the player has been eliminated).

Examples

input.txt	output.txt
L W W W L W	2
L L L L L L	-1

Problem 13. Palindromes

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds
Memory limit:	256 mebibytes

A palindrome is a word which is the same when read forwards as it is when read backwards. A word which has just one letter is also a palindrome. Given a word, what is the longest palindrome that is contained in the word? That is, what is the longest palindrome that we can obtain, if we are allowed to delete characters from the beginning and/or the end of the string?

Input

The input will consist of one line, containing a sequence of at least 1 and at most 40 lowercase English letters.

Output

Output the total number of letters of the longest palindrome contained in the input word. Sample Input 1

Example

input.txt	output.txt
`ababahalamaha`	`5`
`ukkonen`	`3`
`palindrome`	`1`

Problem 14. Nessie and Salesman

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds
Memory limit:	256 mebibytes

Nessie the Monster lives in the some lake. Every day, McJohn, the traveling salesman, brings his shipment of goods across the lake. On a good day, Nessie manages to dump McJohn's shipment into the lake, making all the goods s loss. On a bad day, however, McJohn gets his shipment past Nessie successfully.

Nessie good and bad days are determined by its strength. Its strength starts at a number s, but this number is cut in half (rounding down) after each day's shipment. Nessie is having a good day if his strength is an odd number, and a bad day otherwise. McJohn transports n packs of goods the first day, and then doubles his shipment size each following day to compensate for Nessie's interference (regardless of the outcome of the day).

McJohn wants to know how many packs of goods he can lose. Can you help him?

Input

First line of input contains single positive integer, t $1 \leq t \leq 60$, representing the number of situations to process. Each of the following t lines will describe a situation, consisting of two non-negative integers, s ($s < 2^{31}$) and n ($n < 2^{31}$), representing Nessies's initial strength and the number of McJohn's packs at the first day.

Output

For each situation, output the total number of packs Nessie can dump into the lake in that situation. The answer will always be less than 2^{31}.

Example

input.txt	output.txt

2
1 7
27 35
	7
	945