Problem A. Candidate Problem (Div. 1 Only)

Input file:	`parties.in`
Output file:	`parties.out`
Time limit:	5 seconds
Memory limit:	256 megabytes

Berland is a federation. It is divided into square provinces of equal size. They form a rectangle $n \times m$, which contains n rows with m provinces in each row.

For each province, there is a single political party which dominates in this province. On the eve of the presidential election, one of the candidates is deciding which parties he should support. He has already decided on their number: he will support exactly two political parties.

He expects that if he supports a party, the provinces where this party dominates will vote for him. The provinces which vote for him will form several connected components on the map of Berland. Here, a *connected component* is such a set of provinces that it is possible to get from any province of the set to any other one, moving to the adjacent (by side) provinces belonging to the same set.

The candidate wants to maximize the area of the largest component by choosing two parties to support. If several answers are possible, he wants to maximize the number of components. In case of a tie, he chooses the lexicographically smallest pair of parties a, b (the one which has the smallest a or the smallest b if a are equal).

Input

The input contains one or more tests.

Each test description starts with a line containing two integers n and m ($1 \le n, m \le 500$), where n is a number of provinces rows and m is a number of provinces in each row. The following n lines contain m integers each — identifiers of political parties. Each identifier is between 1 and 10^9, inclusive. It is guaranteed that there are at least two different numbers in the given matrix.

The number of tests in the input doesn't exceed 100, the total number of provinces in all tests doesn't exceed $2.5 \cdot 10^5$.

Output

For each test in the input, write a single line containing the required pair a, b of parties' identifiers ($a < b$).

Examples

parties.in	parties.out
3 3 1 1 1 2 2 3 2 1 1 5 5 1 5 3 1 5 5 3 1 1 5 3 4 1 1 5 3 4 3 1 5 5 3 4 4 1 3 3 26 26 26 26 6 22 22 17 17	1 2 1 5 22 26

Problem B. Formula simplification

Input file:	`formula.in`
Output file:	`formula.out`
Time limit:	5 seconds
Memory limit:	256 megabytes

Given a mathematical formula you are to simplify it. The formula satisfies the following grammar:

- $\langle formula \rangle \rightarrow (\langle formula \rangle)$
- $\langle formula \rangle \rightarrow -\langle formula \rangle$
- $\langle formula \rangle \rightarrow \cos(\langle formula \rangle)$
- $\langle formula \rangle \rightarrow \sin(\langle formula \rangle)$
- $\langle formula \rangle \rightarrow \langle formula \rangle + \langle formula \rangle$
- $\langle formula \rangle \rightarrow \langle formula \rangle * \langle formula \rangle$
- $\langle formula \rangle \rightarrow \langle formula \rangle - \langle formula \rangle$
- $\langle formula \rangle \rightarrow x|0|1|2|3|4|5|6|7|8|9$

where x is a variable on the segment $[a, b]$. Your task is to find a formula which:

- satisfies the grammar described above;
- defines the same function as defined by the given formula on the segment $[a, b]$;
- has the string representation which has the minimum possible length;
- has the string representation which is lexicographically smallest among all such formulas.

It is guaranteed that the answer contains at most 8 characters.

Input

The first line of the input contains the formula. The second line contains the range «`[a,b]`», where a and b are integer numbers ($-10 \le a \le b \le 10$). The formula consists of at most 28 characters and doesn't contain spaces.

Output

Write the required formula in a single line. It should not contain spaces. It is guaranteed that it contains at most 8 characters.

Examples

formula.in	formula.out
`cos(x)*sin(x)*2` `[0,1]`	`sin(2*x)`

Problem C. Frog (Div 1 Only!)

Input file:	`frog.in`
Output file:	`frog.out`
Time limit:	10 seconds
Memory limit:	256 megabytes

A frog is sitting at the very end of a two-dimensional cave. The floor of the cave can be represented as a segment $[0, n]$ on the OX axis. The frog initially sits at position 0. The ceiling of the cave is a bounded by polyline comprising k nodes px_i, py_i ($px_1 = 0$, $px_k = n$, $px_i < px_{i+1}$ for $i = 1 \ldots k-1$). The interior of the ceiling is the set of points strictly above the polyline.

There are m mosquitoes hanging under the ceiling of the cave. All mosquitoes are strictly below the ceiling. The frog has a tongue of a given length s. After the frog eats one mosquito, its tongue increases in its length by 2. The frog can eat a mosquito only if the distance between them does not exceed the length of the frog's tongue and the segment connecting the frog and the mosquito has no common points with the interior of the ceiling.

The frog can jump from one integer position to other integer positions on the floor, that is, the frog can jump from position i to position j where $0 \le i, j \le n$ and $i \ne j$. The flight path of the frog when jumping from i to j ($i \ne j$) is a half of a circle that has its diameter on the segment $[0, n]$ and connects positions i and j. The flight path can't have common points with the interior of the ceiling. The frog can not eat mosquitoes during the flight.

The frog is very greedy, so it wants to maximize the number of mosquitoes eaten. At the same time, the frog is very lazy, so it wants to minimize the number of jumps required to eat the maximum number of mosquitoes and to get to position n.

Input

The first line of the input contains four integers n, k, m, s ($1 \le n \le 200$, $2 \le k \le 10\,000$, $1 \le m \le 10$, $1 \le s \le 200$), denoting the length of the segment the frog moves along, the number of nodes in the polyline, the number of mosquitoes, and the initial length of the frog's tongue, respectively.

Each of the next k lines contains two real numbers px_i, py_i denoting the coordinates of the ceiling polyline nodes ($1 \le py_i \le 200$). The coordinates are given with accuracy of two or less digits after the decimal point.

At last, the next m lines contain two real numbers qx_i, qy_i ($1 \le qy_i \le 200, 0 \le qx_i \le n$), these numbers denote the coordinates of the mosquitoes. The coordinates of the mosquitoes are given with an accuracy of at most two digits after the decimal point. All mosquitoes are strictly below the ceiling and no two mosquitoes are at the same point.

Output

The output file should contain two integers which denote the maximum number of mosquitoes the frog can eat and the minimum number of jumps it needs to do it. Remember that the frog should finish at position n.

Examples

frog.in	frog.out
4 5 1 1 0.00 2.00 1.00 1.00 2.00 2.00 3.00 1.00 4.00 2.00 2.00 1.00	1 2
5 6 2 1 0.00 3.00 1.00 4.00 2.00 3.00 3.00 1.00 4.00 2.00 5.00 1.00 1.00 3.00 4.00 1.00	2 6

Problem D. Best gifts (Div 1 Only!)

Input file:	gifts.in
Output file:	gifts.out
Time limit:	2 seconds
Memory limit:	256 megabytes

Helen loves to receive letters. She also loves to give presents to her friends. She decided that an envelope is a best gift. Clearly, it greatly increases the probability of receiving a letter from her friend!

Helen has n friends and each of them is characterized by a_i — number of letters written to Helen. Of course, those friends who have a higher value of a_i deserve better gifts!

Helen had already decided that for the i-th friend, she will give at least a_i envelopes. Entering the store, Helen found that the envelopes are sold in bundles of m envelopes in each. Buying a bundle of envelopes she wants to give away each envelope in it (she doesn't want to keep envelopes — it is so boring to write letters).

There will be long line of holidays soon and she wants to prepare several gift sets. Each gift set will contain m envelopes (envelopes from a single bundle). A gift set contains n gifts, one per friend. Helen wants to determine how many envelopes she will give to each friend. There is a constraint: the i-th friend should receive a_i or more envelopes in each gift.

According to her memories, not all friends greatly enjoyed her gifts. They said that her gifts are monotonous. Helen does not want to repeat her past mistakes. In other words, she does not want to make two sets such that there will be a friend who will receive an equal number of envelopes in each of them.

Helen does not know how many bundles of envelopes she has to buy. Help Helen understand what is the maximum number of sets she can make for her friends.

Input

The first line of the input contains the number of Helen's friends n and the number of envelopes in the single bundle m ($1 \le n \le 1000$; $1 \le m \le 10\,000$). The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 1000$), where a_i is the minimal number of envelopes in each gift set for the i-th friend.

Output

Write the maximal number of gift sets t on the first line of the output. Following t lines should contain the description of gift sets. Print n integer numbers in each line, where the i-th number means the number of envelopes for the i-th friend in the correspondent gift set.

Examples

gifts.in	gifts.out
2 3 1 1	2 1 2 2 1

Problem E. Leo's fours

Input file:	fours.in
Output file:	fours.out
Time limit:	5 seconds
Memory limit:	256 megabytes

Once upon a time there lived a man called Leo. He was very famous among mathematicians all over the world because of his strange game. The rules of this game were very easy: Leo gave player n integers $a_1, a_2, \ldots, a_n$ and asked him or her to write all fours of distinct indices i, j, k, p such that $a_i + a_j = a_k + a_p$. Many people became mad while searching for missing fours. Association of Clever Mathematicians (ACM) decided that Leo cheats and wants you to count the number of required fours.

Input

The first line of the input contains an integer n ($4 \le n \le 3000$). The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$).

Output

Write the only integer — the number of ways to choose four distinct indices i, j, k, p such that $a_i + a_j = a_k + a_p$.

Examples

fours.in	fours.out
4 1 2 3 4	8
4 1 1 1 1	24

Note

Fours (i_1, j_1, k_1, p_1) and (i_2, j_2, k_2, p_2) are the same only if $i_1 = i_2$, $j_1 = j_2$, $k_1 = k_2$ and $p_1 = p_2$.

Problem F. Delivery problem (Div 1 Only!)

Input file:	ndice.in
Output file:	ndice.out
Time limit:	2 seconds
Memory limit:	256 megabytes

Mr. Beetle is a postman. His routine work is to deliver new issues of the local magazine "N-dice" to its subscribers. Every Monday exactly at 8 a.m. he takes his bicycle and starts working. He is supposed to deliver one copy of the new issue to each subscriber.

Copies of the new issue may have covers with background of different colors, i.e., the cover background of a particular copy may be yellow, green, purple, etc. Imagine that for each color i from 1 to N, Beetle's bag contains infinitely many copies of the issue with cover background of i-th color. He knows that it does not matter to the subscriber what color the cover is. That is why for each subscriber he acts as follows: he puts his hand into the bag, takes out one randomly chosen copy, and finally puts it into the subscriber's pillar box.

Mr. Beetle likes days when he delivers at least one copy of each color. If this may happen with probability at least x, then he says that it is a "lucky" day.

Mr. Beetle doesn't like calculations. But he wants to know the minimal number of subscribers he should serve to be sure that the day is lucky. Your task is to find this number for Mr. Beetle.

Input

The input contains an integer number N ($1 \le N \le 7000$) and a real number x given with exactly six digits after the decimal point ($0.1 \le x < 1$).

Output

Output the minimal number of subscribers Mr. Beetle should serve to be sure that the day is lucky.

Examples

ndice.in	ndice.out
1 0.999999	1
2 0.777770	4

Problem G. Psychologist Problem

Input file:	psychologist.in
Output file:	psychologist.out
Time limit:	5 seconds
Memory limit:	256 megabytes

Professor Neurosis is a practicing psychologist. He is extremely popular in Berland. On arrival at the clinic, each client tells to the secretary the duration t_i of the coming session with psychologist. The i-th client enters the clinic at time s_i and the secretary immediately reports to the professor about new patient and desired session time t_i.

The professor knows that many of his patients are busy people and appreciates their time. For this reason, he wants to minimize the total time all patients spend in the walls of his clinic. For each patient, the time in clinic consists of waiting time and the duration of the session.

Neurosis recently decided that breaks in sessions are not as bad as it might seem at first glance, because in the interval the patient can come up with a new psychological problem. So he can make breaks in the sessions to switch between patients.

Help Professor Neurosis to arrange work with patients in such a way that the total time all patients spend in the clinic is minimal.

Input

In the first line of the input there is the only integer number n ($1 \le n \le 10^5$), where n is the number of patients during the day. Following n lines contain patient descriptions. Each patient is described by two integer numbers s_i, t_i ($1 \le s_i, t_i \le 10^8$), where s_i is the time when the i-th patient entered the clinic and t_i is the session duration for the i-th patient. The lines are ordered in such a way that $s_i \le s_{i+1}$ for each $1 \le i < n$.

Output

Write the required minimal total time in the first line of the output. The following lines should contain the professor's schedule. Each line should contain three integers $j, from, to$ ($1 \le j \le n$, $from < to$) and it means that during the time period $[from, to)$ the professor should work with the j-th patient. Obviously, it is impossible to work with the patient before he or she will come. The sum of period lengths for each patient j should be equal to t_j. Order the lines chronologically. If there are many solutions, write any of them. The output should not contain more than 10^6 lines of the professor's schedule.

Examples

psychologist.in	psychologist.out
2 4 4 10 10	14 1 4 8 2 10 20

Problem H. Round Table

Input file:	table.in
Output file:	table.out
Time limit:	10 seconds
Memory limit:	256 megabytes

Heads of Berland provinces have been involved in a summit in Bremlin. The meeting was held at the highest level and took two days. Each day n heads were sitting around the table and discussing very important problems. Of course, the table was round. The order in which the heads were sitting around the table did not change throughout each day, but probably altered between the days.

It is known that the heads of the provinces are very conservative and cannot have a constructive dialogue in changing conditions. Two heads can have a constructive discussion if and only if the number of persons between them did not change between the days. Since there are two possible ways to count people who separate a particular pair of heads, they are counted clockwise from the first head to the second. This number may be equal to 0.

During the closing ceremony, Berland government decided to reward some pair of heads who have had a constructive discussion. However, the government found that it was not an easy problem to distinguish such pairs. Help them to solve this problem and find the required pair of heads.

Input

The first line of the input contains an integer number n ($3 \le n \le 10^6$), the number of heads of Berland provinces. Each head is assigned an integer number (from 1 to n). The second line of the input contains a sequence of numbers which describes the arrangement of the heads as they were sitting around the table during the first day. The third (last) line of the input contains a sequence of numbers describing the order in which they were sitting around the table during the second day. Each of the last two lines of the input file is a permutation of integers from 1 to n, inclusive.

Output

Write a pair of integer numbers — the numbers assigned to any pair of heads who have had a constructive discussion. If there are several pairs of this kind, write any of them. Write "`-1 -1`" (without quotes) in case of no such pair.

Examples

table.in	table.out
5 3 4 2 1 5 1 2 3 4 5	3 1

Problem I. K-th heap (Div. 1 Only)

Input file:	`heaps.in`
Output file:	`heaps.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Consider all permutations of numbers from 1 to n. A permutation $[p_1, p_2, \ldots, p_n]$ is called a *heap* if for each index $i > 1$ the inequality $p_i < p_{i/2}$ holds true. Here, $i/2$ is the result of dividing i by two rounded down to the nearest integer.

For example, there are three heaps for $n = 4$: $[4, 2, 3, 1]$, $[4, 3, 1, 2]$ and $[4, 3, 2, 1]$.

As you may know, a permutation p *lexicographically precedes* a permutation q if there exists an index j, $1 \le j \le n$, such that $p_i = q_i$ for any $1 \le i < j$ and $p_j < q_j$. Let us sort all heaps for a given n lexicographically in increasing order. Your task is to find the k-th heap in this sorted list, where k is a 1-based index.

Input

The input file contains two integer numbers n and k ($1 \le n \le 1000$; $1 \le k \le 10^{15}$).

Output

Output the k-th heap with its elements separated by a single space. Output integer "`-1`" if the k-th permutation does not exist.

Examples

heaps.in	heaps.out
4 2	4 3 1 2

Problem J. Bindians Wigwams

Input file:	wigwams.in
Output file:	wigwams.out
Time limit:	10 seconds
Memory limit:	256 megabytes

Many years ago Bindian tribes lived on the territory of modern Berland. Bindians built tetrahedral wigwams and used them as their houses. How to construct a wigwam? It is very easy — you need to find six wooden sticks and construct a tetrahedron using the sticks as its edges. Sure enough, sticks should be very strong and it is forbidden to cut them or use part of a stick as a tetrahedron edge. Also, a wigwam should have positive volume.

Ancient Bindians liked to organize contests where the most popular problem was as follows: given a pile of sticks, find the number of ways one can choose six sticks from the pile so that these six sticks may be used to construct a wigwam. Ancient Bindians knew how to solve this problem. What about you? Remember that the sticks are different even if they have equal lengths.

Input

The first line of the input contains an integer number n ($1 \le n \le 30$), where n is the number of sticks in the pile. The second line contains a list of lengths of the sticks in the pile separated with spaces. Each length is an integer number between 1 and 100.

Output

Write the only integer — the number of ways to choose six sticks which may be used to construct a wigwam.

Examples

wigwams.in	wigwams.out
7 4 3 4 4 4 4 4	7
6 1 1 1 1 1 1	1

Problem K. Gophers (Div. 1 Only)

Input file:	gophers.in
Output file:	gophers.out
Time limit:	6 seconds
Memory limit:	256 megabytes

A long time ago there was a big gopher settlement to the south of Berland. Last year poor gophers were totally exhausted by a terrible drought.

However, recently, new generous regent of Berland decided to create a colony of gophers in the Berland National Park (BNP). To prevent the extinction of gophers, you were hired to construct exactly two watering places on the BNP territory (the government cannot afford constructing the third one).

Let us introduce a Cartesian coordinate system in BNP. First of all, consider BNP as a square with two opposite angles $(-20\,000, -20\,000)$ and $(20\,000, 20\,000)$. Every gopher has its own hole on the BNP territory inside the square with the opposite angles $(-10\,000, -10\,000)$ and $(10\,000, 10\,000)$. Each watering place should have the shape of a line segment located within the territory of BNP (therefore, it can be located on its border). A watering place may pass through one or more holes.

There are many predators in BNP, so the distance to be covered by any gopher on his way to the nearest watering place and back should be as small as possible. Note that gophers cannot jump outside the BNP area!

Input

The first line contains an integer N ($1 \le N \le 120$) — the number of gophers in the polulation. Then follow N lines; each of them describes the location of the hole of the i-th gopher by two integer coordinates x and y.

Output

The first line of the output should contain a single real number — the minimal possible maximal distance to be covered by any gopher from his hole to the nearest watering place and back home. Print this number with at least nine digits after the decimal point.

Examples

gophers.in	gophers.out
5 -10 0 0 0 10 0 0 10 0 -10	0.000000000000
7 -10 0 0 0 10 0 0 10 1 10 0 -10 1 -10	1.000000000000

Problem L. Candidate Problem (Div. 2 Only)

Input file:	`parties.in`
Output file:	`parties.out`
Time limit:	5 seconds
Memory limit:	256 megabytes

Berland is a federation. It is divided into square provinces of equal size. They form a rectangle $n \times m$, which contains n rows with m provinces in each row.

For each province, there is a single political party which dominates in this province. On the eve of the presidential election, one of the candidates is deciding which parties he should support. He has already decided on their number: he will support exactly two political parties.

He expects that if he supports a party, the provinces where this party dominates will vote for him. The provinces which vote for him will form several connected components on the map of Berland. Here, a *connected component* is such a set of provinces that it is possible to get from any province of the set to any other one, moving to the adjacent (by side) provinces belonging to the same set.

The candidate wants to maximize the area of the largest component by choosing two parties to support. If several answers are possible, he wants to maximize the number of components. In case of a tie, he chooses the lexicographically smallest pair of parties a, b (the one which has the smallest a or the smallest b if a are equal).

Input

The input contains one or more tests.

Each test description starts with a line containing two integers n and m ($1 \le n, m \le 100$), where n is a number of provinces rows and m is a number of provinces in each row. The following n lines contain m integers each — identifiers of political parties. Each identifier is between 1 and 10^9, inclusive. It is guaranteed that there are at least two different numbers in the given matrix.

The number of tests in the input doesn't exceed 100, the total number of provinces in all tests doesn't exceed 10^4.

Output

For each test in the input, write a single line containing the required pair a, b of parties' identifiers ($a < b$).

Examples

parties.in	parties.out
3 3 1 1 1 2 2 3 2 1 1 5 5 1 5 3 1 5 5 3 1 1 5 3 4 1 1 5 3 4 3 1 5 5 3 4 4 1 3 3 26 26 26 26 6 22 22 17 17	1 2 1 5 22 26

Problem M. K-th heap (Div. 2 Only)

Input file:	`heaps.in`
Output file:	`heaps.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Consider all permutations of numbers from 1 to n. A permutation $[p_1, p_2, \ldots, p_n]$ is called a *heap* if for each index $i > 1$ the inequality $p_i < p_{i/2}$ holds true. Here, $i/2$ is the result of dividing i by two rounded down to the nearest integer.

For example, there are three heaps for $n = 4$: $[4, 2, 3, 1]$, $[4, 3, 1, 2]$ and $[4, 3, 2, 1]$.

As you may know, a permutation p *lexicographically precedes* a permutation q if there exists an index j, $1 \le j \le n$, such that $p_i = q_i$ for any $1 \le i < j$ and $p_j < q_j$. Let us sort all heaps for a given n lexicographically in increasing order. Your task is to find the k-th heap in this sorted list, where k is a 1-based index.

Input

The input file contains two integer numbers n and k ($1 \le n \le 15$; $1 \le k \le 10^7$).

Output

Output the k-th heap with its elements separated by a single space. Output integer "`-1`" if the k-th permutation does not exist.

Examples

heaps.in	heaps.out
4 2	4 3 1 2

Problem N. Gophers (Div. 2 Only)

Input file:	gophers.in
Output file:	gophers.out
Time limit:	2 seconds
Memory limit:	256 megabytes

A long time ago there was a big gopher settlement to the south of Berland. Last year poor gophers were totally exhausted by a terrible drought.

However, recently, new generous regent of Berland decided to create a colony of gophers in the Berland National Park (BNP). To prevent the extinction of gophers, you were hired to construct exactly two watering places on the BNP territory (the government cannot afford constructing the third one).

Let us introduce a Cartesian coordinate system in BNP. First of all, consider BNP as a square with two opposite angles $(-20\,000, -20\,000)$ and $(20\,000, 20\,000)$. Every gopher has its own hole on the BNP territory inside the square with the opposite angles $(-10\,000, -10\,000)$ and $(10\,000, 10\,000)$. Each watering place should have the shape of a line segment located within the territory of BNP (therefore, it can be located on its border). A watering place may pass through one or more holes.

There are many predators in BNP, so the distance to be covered by any gopher on his way to the nearest watering place and back should be as small as possible. Note that gophers cannot jump outside the BNP area!

Input

The first line contains an integer N ($1 \le N \le 50$) — the number of gophers in the polulation. Then follow N lines; each of them describes the location of the hole of the i-th gopher by two integer coordinates x and y.

Output

The first line of the output should contain a single real number — the minimal possible maximal distance to be covered by any gopher from his hole to the nearest watering place and back home. Print this number with at least nine digits after the decimal point.

Examples

gophers.in	gophers.out
5 -10 0 0 0 10 0 0 10 0 -10	0.000000000000
7 -10 0 0 0 10 0 0 10 1 10 0 -10 1 -10	1.000000000000

Problem O. Lucky number 7 (Div. 2 Only)

Input file:	`seven.in`
Output file:	`seven.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Petya has one lucky number — 7. He always tries to find the signs of this number everywhere, and feels really happy if he manages to do it. And now he is figuring out how to find some connection between the number 7 and the matrix of digits. Probably he should count the number of 7-s in the matrix? No, there will be so few such digits... And then really bright idea comes to his mind: he will consider paths of the fixed length in this matrix, which start at the cell $(1, 1)$. He will examine the numbers, obtained by successfully writing down all digits from the path. And if such number is divisible by 7, he will consider that path lucky! There will be a huge number of such paths! The last thing he has to do is to count this number, and he asks you to take care of it. Adjacent cells in the path should have common side. Path can contain some cells more than once.

Input

First line contains three integer numbers n, m and k — sizes of the matrix and the length of the every path ($1 \le n, m \le 50, 1 \le k \le 100$). Next n lines contain m digits from 1 to 9 each.

Output

Output the number of lucky paths of the length k. This number can be very large, so output it modulo 10^9.

Examples

seven.in	seven.out
2 2 3 7 1 7 4	2

Problem P. Car numbers (Div. 2 Only)

Input file:	numbers.in
Output file:	numbers.out
Time limit:	2 seconds
Memory limit:	256 megabytes

After the recent reform all car numbers in Berland consist of two blocks: the first block is three english letters ($A \dots Z$), and the second block is three digits. Initially police department was planning to give car numbers to the car owners in the following way: every driver will receive lexicographically next car number, starting from the number $AAA000$. For instance, second driver will receive number $AAA001$ and 1001-th driver will receive number $AAB000$. But police department understood that some numbers are more beautiful than the other, and many drivers would like to have them. Many drivers, for example, want to have car number $XXX007$. Police has created the list of M beautiful numbers, ordered from the most beautiful to the least beautiful. So, now every driver has a choice: he can pay money and get the most beautiful car number from remaining in the list or he can get lexicographically next car number which is not in the list without any fee. Somehow you got the list of the beautiful numbers. Also you know who of the first N drivers actually paid for the beautiful number. You should find out car numbers given for the first N drivers.

Input

First line of the input contains two integer numbers N and M — the number of drivers and the number of beautiful car numbers in the list ($1 \le N \le 10000, 0 \le M \le 1000$). The next M lines contain one car number each. This is the list of the beautiful car numbers, given from the most beautiful to the least beautiful. The last line of the input contains string s of N characters. $s_i = 1$ if the i-th driver had paid for the beautiful number, and $s_i = 0$ otherwise. This string contains not more than M ones.

Output

Output N lines. In the i-th line print car number for the i-th driver.

Examples

numbers.in	numbers.out
4 2 AAA001 XXX007 1010	AAA001 AAA000 XXX007 AAA002

Problem Q. Lights (Div. 2 Only)

Input file:	`lights.in`
Output file:	`lights.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

The government is going to buy a very expensive automatic system to control the lights on the main street of the Berland capital. This system controls lights state in a quite strange way, so it was decided that you will write a program to simulate it's behaviour. Lights on the main street are placed in a line and are numbered fron 1 to N. Every light has two states: on and off. Control system allows to specify the segment $[l_i, r_i]$ and change the state of every light from this segment to the opposite. Also your simulating program should be able to report the state of the given light with the number q_i at any moment of time. Initially all lights are switched off.

Input

First line contains two integer numbers N and Q — the number of lights and the number of queries respectively ($1 \le N \le 100000, 1 \le Q \le 200000$). Next Q lines contains queries, one query per line. First number in each line is the type the query: 1 for state change on the segment and 2 for reporting the state. In the first case this number is followed by the pair l_i, r_i ($1 \le l_i \le r_i \le N$), while in the second case it is followed by q_i ($1 \le q_i \le N$) — the number of the light.

Output

For every query of the second type output one line. Print 0 in it if the light was switched off, and print 1 otherwise.

Examples

lights.in	lights.out
4 5 1 2 4 2 2 1 1 3 2 1 2 3	1 1 0