Problem A. Tourism

Time limit:	1 second
Memory limit:	512 megabytes

In Lineland the Ministry of Tourism has decided to prepare a sightseeing route for tourists. There are n cities in Lineland, they are located at a straight road. The cities are enumerated from 1 to n along this road. Some of these cities contain sights interesting for tourists. Let us introduce the value a_i, if the i-th city has a sight then $a_i = 1$, otherwise $a_i = 0$.

The sightseeing route for tourists will be chosen by the Ministry of Tourism. You have to provide them with two *similar* routes to choose from. Each route must visit a continuous segment of cities along the line: $l, l+1, \ldots, r$. Two routes are similar if they visit the same number of cities, and also they visit the same number of cities that have sights.

Your task is two find two similar routes that have the maximum number of visited cities.

Input

Input contains a string that consists of characters "0" and "1". The i-th character of the string represents a_i. The length of this string is from 3 to 10^6. Input contains no spaces.

Output

Output four integers: s_1, t_1, s_2 and t_2 — the numbers of the first and the last cities of the first route, and the numbers of the first and the last cities of the second route, respectively. It is guaranteed that the answer always exists.

Example

standard input	standard output
10101	1 4 2 5

Problem B. Open cup

Time limit:	2 seconds
Memory limit:	512 megabytes

Year 3017. Human race has become so good in space flights, that one of the most popular sport activities is now racing between asteroids. In two days the next stage of Asteroid Races Open Cup will take place. It is known that the Open Cup is organized by a famous space races evangelist Spark.

Spark likes to keep track of statistics about top participants, and he likes to tell interesting stories about previous races and their results. In one of such stories he has accidentally disclosed, that the stage that will take place in two days is named "Grand Prix of Alpha Centauri". Alpha Centauri is a well-studied system, so anybody familiar with the rules of the races can guess that participants will have to fly from asteroid ICM-2017 to asteroid YOY-2018.

For each race a two-dimensional plane is chosen and all racing space ships movements always take place in that plane. Let's assume that asteroids don't change their mutual positions while race is going on. The plane where the race takes place intersects the two asteroids via convex non-intersecting and non-touching polygons, specified by their vertices in counterclockwise order.

Racing space ship can depart and land only perpendicular to the surface of an asteroid. Participant has to choose the starting point of the race strictly inside one of the edges of the polygon. After taking off the asteroid, the space ship is going in the direction perpendicular to the edge that it started from, moving away from the asteroid. When the space ship lands to the other asteroid, it should land to a point strictly inside some edge of the polygon, its movement when landing must be perpendicular to that edge.

Elbrus is a novice racer, and he plans to participate in the race, but he has not trained enough yet, so he can't maneuver in space. Fortunately, it can happen that there is a direct route between asteroids, so a space ship can fly from one asteroid to another without turning at all. In other words, it is possible to choose start and finish points so that the segment between them is perpendicular to the edges that contain the points, and the segment doesn't intersect the interior of the asteroids.

Help Elbrus to figure out whether he can take part in the race or not.

Input

The first line contains one integer n — the number of vertices in the polygon of the asteroid ICM-2017 ($3 \le n \le 200\,000$). The following n lines contain pairs of integer x_i, y_i — coordinates of vertices of the polygon ICM-2017 ($-10^9 \le x_i, y_i \le 10^9$).

The following line contains one integer m — the number of vertices of the polygon of the asteroid YOY-2018 ($3 \le m \le 200\,000$). The following m lines contain pairs of integer x'_i, y'_i — coordinates of vertices of polygon YOY-2018 ($-10^9 \le x'_i, y'_i \le 10^9$).

It is guaranteed that both polygons are convex, their vertices are given in counterclockwise order, and no three vertices of one polygon belong to the same line.

Output

If it is possible to find direct route between polygons, output "`YES`" in the first line, and output numbers of edges of asteroids ICM-2017 and YOY-2018 that the direct route must connect in the second line.

You can assume that the edge i connect vertices i and $i + 1$ of the polygon, the edge n of the polygon ICM-2017 connect vertices n and 1, the edge m of the polygon YOY-2018 connect vertices m and 1.

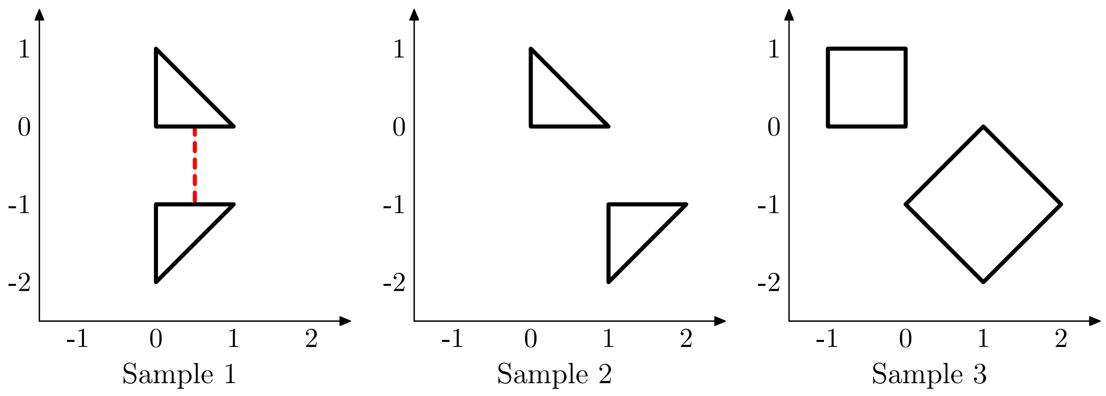
It there is no direct route between polygons, output a word "`NO`".

Examples

standard input	standard output
3	YES
0 0	1 3
1 0	
0 1	
3	
0 -1	
0 -2	
1 -1	
3	NO
0 0	
1 0	
0 1	
3	
1 -1	
1 -2	
2 -1	
4	NO
-1 0	
0 0	
0 1	
-1 1	
4	
1 0	
0 -1	
1 -2	
2 -1	

Note

The picture shows the polygons of the asteroids in sample tests.



Problem C. Configuration file

Time limit:	2 seconds
Memory limit:	512 megabytes

Vadim develops a parser for configuration files of his project. A configuration file consists of blocks delimited with braces: "`{`" marks the beginning of the block, and "`}`" marks the end of the block. Blocks can be nested. One block can contains several other blocks.

There are variables in the configuration file. Each variable has a name that consists of at most ten lowercase English letters. Each variable has an integer value, initially all variables are set to 0.

New values can be assigned to variables. Assigning a constant value to a variable is specified as "`<variable>=<number>`", here `<variable>` is the variable's name, and `<number>` is an integer, its absolute value doesn't exceed 10^9. The parser reads the file line after line. As it reads the assignment operation, it assigns the new value to the variable. This value is used until the end of the current block, after that the previous value is restored. If the current block has some nested blocks following the assignment, the new value of the variable is used inside those blocks.

A variable can also be assigned the value of another variable. Such operation is specified as "`<variable1>=<variable2>`". When the parser reads such line, it assigns the current value of `variable2` to `variable1`. Similarly to the case of a constant value assignment, the new value is used until the end of the current block. After the end of the current block, the value that the variable had at the beginning of the block is restored.

For debugging purpose Vadim wants to print all values that are assigned when processing lines of the form "`<variable1>=<variable2>`". Help him to debug the parser.

Input

Input data contains at least 1 and at most 10^5 lines. Each line has one of four types:

- `{` — beginning of the block;
- `}` — end of block;
- `<variable>=<number>` — assigning a constant value to a variable;
- `<variable1>=<variable2>` — assigning one variable to another, here `<variable1>` and `<variable2>` can be the same.

It is guaranteed that the input is correct and corresponds to the statement. Input doesn't contain spaces.

Output

For each line that has the form "`<variable1>=<variable2>`" print the value that is assigned.

Example

standard input	standard output

a=b
b=123
var=b
b=-34
{
c=b
b=1000000000
d=b
{
a=b
e=var
}
}
b=b
	0
	123
	-34
	1000000000
	1000000000
	123
	-34

Problem D. Equal Maximums

Time limit:	1 second
Memory limit:	512 megabytes

Sasha is preparing for programming contests, so he is studying data structures and corresponding problems. One common problem he has noticed is Range Maximum Query.

That problem is defined as follows. There is an array a of n integers: $a_1, a_2, \ldots, a_n$. One has to answer queries "find the maximum value in a range from the i-th element to the j-th one", so the problem is to calculate $\max\{a_i, a_{i+1}, \ldots, a_j\}$.

Of course, this task is not difficult for Sasha, and soon there was a very fast program that answers such queries. Looking at the answers, he has noticed that the answers for different queries are often the same.

Now Sasha is wondering, how many ways are there to choose a pair of non-overlapping ranges that have equal maximum elements.

Your task is to help Sasha. Find the number of quadruples i, j, k, l, such that $1 \le i \le j < k \le l \le n$ and $\max\{a_i, a_{i+1}, \ldots, a_j\} = \max\{a_k, a_{k+1}, \ldots, a_l\}$. Since that number can be quite large, you have to output it modulo 1 000 000 007.

Input

The first line of input contains one integer n: the length of Sasha's array ($2 \le n \le 100\,000$). The second line of input contains n integers: array elements (they are positive and don't exceed 10^9).

Output

Output the number of pairs of non-overlapping ranges with equal maximums. Print it modulo $10^9 + 7$.

Examples

standard input	standard output
6 3 3 4 4 3 2	16
12 1 3 2 3 4 1 3 4 3 2 2 5	177

Note

Pairs of ranges from the first sample:

$[\mathbf{3}], [\mathbf{3}], 4, 4, 3, 2$	$i = 1$	$j = 1$	$k = 2$	$l = 2$
$[\mathbf{3}], 3, 4, 4, [\mathbf{3}], 2$	$i = 1$	$j = 1$	$k = 5$	$l = 5$
$[\mathbf{3}], 3, 4, 4, [\mathbf{3}, 2]$	$i = 1$	$j = 1$	$k = 5$	$l = 6$
$[\mathbf{3}, \mathbf{3}], 4, 4, [\mathbf{3}], 2$	$i = 1$	$j = 2$	$k = 5$	$l = 5$
$[\mathbf{3}, \mathbf{3}], 4, 4, [\mathbf{3}, 2]$	$i = 1$	$j = 2$	$k = 5$	$l = 6$
$3, [\mathbf{3}], 4, 4, [\mathbf{3}], 2$	$i = 2$	$j = 2$	$k = 5$	$l = 5$
$3, [\mathbf{3}], 4, 4, [\mathbf{3}, 2]$	$i = 2$	$j = 2$	$k = 5$	$l = 6$
$[3, 3, \mathbf{4}], [\mathbf{4}], 3, 2$	$i = 1$	$j = 3$	$k = 4$	$l = 4$
$[3, 3, \mathbf{4}], [\mathbf{4}, 3], 2$	$i = 1$	$j = 3$	$k = 4$	$l = 5$
$[3, 3, \mathbf{4}], [\mathbf{4}, 3, 2]$	$i = 1$	$j = 3$	$k = 4$	$l = 6$
$3, [3, \mathbf{4}], [\mathbf{4}], 3, 2$	$i = 2$	$j = 3$	$k = 4$	$l = 4$
$3, [3, \mathbf{4}], [\mathbf{4}, 3], 2$	$i = 2$	$j = 3$	$k = 4$	$l = 5$
$3, [3, \mathbf{4}], [\mathbf{4}, 3, 2]$	$i = 2$	$j = 3$	$k = 4$	$l = 6$
$3, 3, [\mathbf{4}], [\mathbf{4}], 3, 2$	$i = 3$	$j = 3$	$k = 4$	$l = 4$
$3, 3, [\mathbf{4}], [\mathbf{4}, 3], 2$	$i = 3$	$j = 3$	$k = 4$	$l = 5$
$3, 3, [\mathbf{4}], [\mathbf{4}, 3, 2]$	$i = 3$	$j = 3$	$k = 4$	$l = 6$

Problem E. School Olympiad

Time limit:	1 second
Memory limit:	512 megabytes

School olympiad in informatics will take place in the capital of Lineland tomorrow.

All houses in the capital are located at integer points along the straight Main street. The Olympiad will take place at three locations at the Main street. Every location has a limit on the maximum number of participants.

The first location is at the point with coordinate a, it's limit is n_a participants. The second location is at the point with coordinate b, it's limit is n_b participants. The third location is at the point with coordinate c, it's limit is n_c participants.

There are n students that are going to participate in the Olympiad, the i-th of them lives in the house that is located at the point with coordinate x_i. Organizers need to choose a participation location for each student. It's forbidden to exceed the limit of a location. It's guaranteed that the total limit is enough for all students to participate.

If a student lives at a point with coordinate p, and a location that he would participate at is at a point with coordinate q, they must walk a distance of $|p - q|$ before the Olympiad. Help organizers to find the minimum total distance that the students would have to walk before the Olympiad in case of optimal assignment of students to the three locations.

Input

The first line contains two integers a and n_a — the coordinate of the first location and its participation limit, the second line contains two integers b and n_b — the coordinate of the second location and its participation limit, the third line contains two integers c and n_c — the coordinate of the third location and its participation limit ($-10^9 \le a, b, c \le 10^9$; $1 \le n_a, n_b, n_c \le 100\,000$).

The fourth line contains an integer n — the number of students ($1 \le n \le 100\,000$, $n \le n_a + n_b + n_c$).

The next line contains n integers x_i — the coordinates of the students' houses ($-10^9 \le x_i \le 10^9$).

Output

Output one integer — the minimum possible total distance that the students would have to walk before the Olympiad.

Example

standard input	standard output

0 1
3 2
6 3
4
-2 1 3 2
	8

Problem F. Pandemic 2

Time limit:	1 second
Memory limit:	512 megabytes

Vasya's friends are always playing the same board game called "Pandemic". Vasya is tired of this game, so he has decided to create his own version.

He chooses n cities on the map of Byteland and $n-1$ bidirectional roads that connect them. Cities are enumerated by integers from 1 to n, and roads are enumerated by integers from 1 to $n-1$. The i-th road has length of l_i kilometers and connects cities u_i and v_i. There is a path by the roads between any pair of the cities. During the game roads and cities become infected. A city can be either entirely infected, or not infected at all, and a road can contain both infected and not infected parts.

At the beginning of the game, the cities $a_1, a_2, \ldots, a_m$ become infected. Then the infection spreads along the adjacent roads. A city becomes infected at the moment when the infection reaches it. At the same time the infection begins to spread along roads adjacent to the city that has just been infected. The city becomes infected instantly, and the infection spreads along any road at the constant speed of one kilometer per minute.

At each moment of the game the yet uninfected cities and parts of roads form *uninfected connected components*. An uninfected city and adjacent uninfected parts of roads are always in the same component. Two uninfected cities are in the same component if and only if there is a path of uninfected roads that connects them. Uninfected connected component can contain no cities at all, in this case it consists of a single uninfected part of the road that connects already infected cities.

The game ends when all cities and roads become infected. Vasya has not yet come up with the roles for players, but first he wants to know what is the maximum number of uninfected connected components that would exist on the board at some moment of the game.

Input

The first line of input consists of one integer n — a number of chosen cities ($2 \le n \le 10^5$). The following $n-1$ lines contain descriptions of chosen roads, the i-th line contains three integers u_i, v_i and l_i — the cities connected by the i-th road and its length ($1 \le u_i, v_i \le n$; $u_i \ne v_i$; $1 \le l_i \le 10^9$).

The next line contains one integer m — the number of cities, that are infected at the beginning of the game ($1 \le m \le n$). The next line contains $a_1, a_2, \ldots, a_m$ — the numbers of these cities ($1 \le a_i \le n$, all a_i are distinct).

Output

Print one integer — the maximum number of uninfected connected components on the board at some moment of the game.

Example

standard input	standard output

8
1 2 1
1 3 1
2 4 1
2 5 1
3 6 1
3 7 1
1 8 4
2
1 8
	5

Problem G. Puzzle

Time limit:	1 second
Memory limit:	512 megabytes

Andrew is going to open his own factory for manufacturing puzzle pieces. He needs to order a special device that would cut pieces out of cardboard sheets. Also he needs to order a set of tips for it. Each tip allows to cut pieces of a particular form.

A puzzle piece is a square, each of its four sides can contain a rounded tab, a blank cut, or be smooth. The pieces can be of three different types:

- «*Corners*» — such pieces have exactly two adjacent smooth sides forming a corner;
- «*Borders*» — such pieces have exactly one smooth side;
- «*Normal*» — such pieces have no smooth sides.

Rounded tabs and blank cuts can each be of k types. So, there are $2k+1$ options for a puzzle piece side — rounded tab of one of k types, blank cut of one of k types, a smooth side.

Andrew needs to find out how many different tips he need to order. Pieces, such that one of them can be rotated to become equal to another one, can obviously be cut out using the same tip.

Help Andrew to find the number of different tips he needs to order so that he would be able to cut any possible puzzle piece.

Input

The only line of input contains an integer k — the number of different rounded tabs and blank cuts types ($1 \le k \le 10^4$).

Output

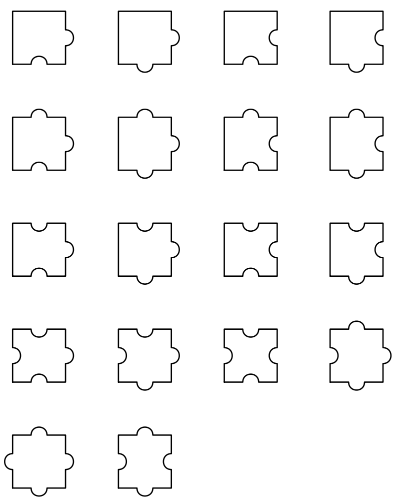
Output the number of different tips Andrew need to order.

Example

standard input	standard output
1	18

Note

All 18 tips for $k = 1$ are presented at the picture below:



Problem H. Game of 2-SAT

Time limit:	1 second
Memory limit:	512 megabytes

This is an interactive problem.

Today Petya has an exam on Computational Complexity. He has the following task.

There is a boolean formula in *2-CNF*. The task is to determine whether it is possible to assign values `true` or `false` to its variables so that the formula evaluates to `true`.

A boolean formula has several variables. Each of them can be equal to either `true` or `false`. A formula in *2-CNF* is defined as follows:

- It is a *conjunction* (i.e. «and» operation) of one or more *disjuncts*.
- Each *disjunct* is an «or» operation applied to two distinct variables or their negations.
- Each variable can be in several disjuncts.
- Some variables may not appear in any disjuncts.

There are a few examples of boolean formulas in *2-CNF* (symbols «&», «|», «!» mean «and», «or», «not» respectively):

- $(a|b)\&(c|!d)$
- $(!a|!b)$
- $(a|b)\&(a|!b)\&(b|!c)$

Petya thinks that the answer is «impossible». Now he has to prove it to the teacher. To do this the teacher has proposed him an interesting game.

In the beginning of the game the variables are not assigned any values. Petya and the teacher take alternating turns, starting with Petya. Each turn the current player assigns `true` or `false` to any unassigned variable. The game ends when there are no unassigned variables left.

If the formula equates to `false` for the final assignment of variables, Petya wins the game and therefore passes the exam. Otherwise, Petya was obviously wrong, and he fails the exam.

Your task is to help Petya pass his exam. Write a program to play the game against the teacher. If Petya has no winning strategy, you should inform them not to spend time for a hopeless game.

Interaction Protocol

In the beginning of the input, a formula in *2-CNF* is given. The first line contains two integers n and m — number of variables and the number of disjuncts ($2 \le n \le 10^4$, $1 \le m \le 2 \cdot 10^4$).

Then disjuncts follow in m lines. Each of them is defined by two integers — indices of variables. If a variable appears in the disjunct with a negation then its index is given with a minus sign. Variables are numbered from 1 to n.

If Petya has no winning strategy, you should output “`-1 -1`” and terminate.

Otherwise, the game starts. Your program plays as Petya and jury's program plays as the teacher. Each turn the current player prints two integers in a single line. An assignment of value v to the x-th variable is denoted by a line “`x v`”. Here $v = 0$ means `false` and $v = 1$ means `true`.

Your program should print each Petya's turn to stdout. After each turn, if the game is not over yet, your program should read teacher's turn from stdin.

When the game ends, your program should terminate.

Note

After each turn you should output a line end character. If you use "`writeln`" in Pascal, "`cout << ... << endl`" in C++, "`System.out.println`" in Java or "`print`" in Python then the output stream flushes automatically. It is recommended to call "`flush`" if you output in some other way. Note that you should always print a line end character.

There are common reasons of unsuccessful outcomes.

If your program follows the protocol but in the end of the game the formula evaluates to `true`, you will get "Wrong Answer".

If your program prints "`-1 -1`" instead of the first move but you have a winning strategy, you will get "Wrong Answer".

If your program prints incorrect messages, you will also get "Wrong Answer".

If your program violates the protocol and waits for input when the jury's program is also waiting for it, you will get "Idleness Limit Exceeded". Note that you'll also get "Idleness Limit Exceeded" if you don't print a line end character or output in some different way than described above and don't call "`flush`".

Examples

standard input	standard output
`3 2`	
`1 2`	
`-1 -2`	
	`3 1`
`2 0`	
	`1 0`
`2 2`	
`1 2`	
`-1 -2`	
	`-1 -1`

Problem I. Circuit

Time limit:	1 second
Memory limit:	512 megabytes

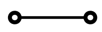
Zhenya studies electricity at Physics practice lessons in his school. His next task is to analyze an electrical network.

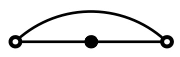
Electrical network is the set of nodes, connected by wires. One of the nodes is a source node, and another one is a sink node. The electrical network in this task is *correct*.

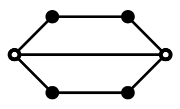
Electrical network is called *correct* iff it can be constructed using following rules:

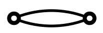
- Basic network, that has a source node and a sink node connected by a single wire, is correct.
- If there are two correct networks A and B, then the network $S(A, B)$, created by joining the sink node of A and the source node of B, is correct. The source node of the resulting network is the source node of A, the sink node is the sink node of B.
- If there are two correct networks A and B, then the network $P(A, B)$, created by joining the source node of A with the source node of B and the sink node of A with the sink node of B is correct. The source node of the resulting network is the joined source node, the sink node is the joined sink node.
- Every correct network can be created from basic networks using a finite number of the two previous rules.

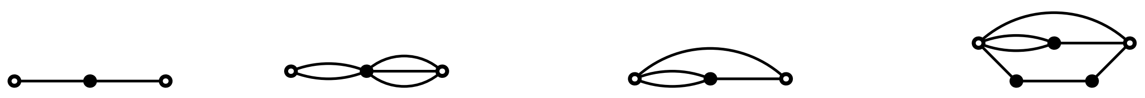
The picture shows examples of correct networks.


Basic network


Network from example 1


Network from example 2


Network from example 3


Other examples of correct networks

Zhenya doesn't like Physics, but he likes Graph Theory. So, instead of doing his task, he wants to count the number of ways to remove some wires from the network, so that the remaining wires formed a *tree*: there was exactly one path by wires from any node to any other node.

That number could be very big, so Zhenya wants to find its reminder modulo 998 244 353.

Input

The first line of input contains two itnegers n and m — number of nodes in the network and the number of wires ($1 \le n, m \le 100\,000$).

The following m lines contain the description of wires. Each of them contains two integers a and b, they mean that nodes a and b are connected by a wire ($1 \le a, b \le n$; $a \ne b$).

It's guaranteed, that the given network is correct. The source node is the node number 1, and the sink node is the node number n. Note, that correct networks can have several wires between a pair of nodes.

Output

Print one integer — the number of ways to remove the wires, modulo 998 244 353.

Examples

standard input	standard output
3 3	3
1 2	
2 3	
3 1	
6 7	15
1 2	
1 3	
1 6	
2 4	
3 5	
4 6	
5 6	
2 2	2
1 2	
1 2	

Note

In the first and in the third exmaple any single wire can be removed.

In the second example either wire $(1, 6)$ and any other wire can be removed, or one wire from each of the chains $1 - 2 - 4 - 6$ and $1 - 3 - 5 - 6$.

Problem J. Entertainment with Javelins

Time limit:	3 seconds
Memory limit:	512 megabytes

Sometimes the Arithmetics classes are cancelled in Arithmetics University. To occupy the students during cancelled classes the dean has decided to introduce javelin throwing. So he bought a target for throwing and a machine that sells javelins.

The target has multiple layers. Each javelin can break through several layers. The goal of the javelin throwing is to break through all layers.

Each javelin is characterized by its diameter, its strength and its cost. The selling machine proposes to buy javelins in some fixed order. Each proposed javelin can either be bought and immediately thrown, or can be discarded. In both cases the machine then proposes to buy the next javelin. When the machine has proposed all the javelins, it turns itself off.

The first thrown javelin makes a round hole with the diameter equal to the diameter of the thrown javelin, and the depth equal to the strength of the javelin. Each of the followign javelins thrown hits the center of the hole. If the diameter of the last thrown javelin does not exceed the diameters of all previously thrown javelins, then this javelin breaks through layers, starting from the first layer that was not broken through yet. Otherwise the javelin breaks through layers starting from the deepest layer, that has the hole with the diameter strictly less than the diameter of the javelin.

Each javelin breaks through the number of layers equal to its strength. The number of broken through layers by a javelin does not depend on the layers broken through by javelins with smaller diameter. If a strength of a javelin is enough to break through the last layer, then the javelin passes through the target and the javelin throwing finishes.

You know the order that the machine will propose the javelins. Find out which javelins to buy to break through all layers and spend the minimum cost.

Input

The first line of input contains two integers n and m — the number of javelins and the number of target layers ($1 \le n, m \le 2000$).

Each of the following n lines contains a description of a javelin. The description consists of three integers d_i, s_i and c_i — the diameter, the strength and the cost of the javelin ($1 \le d_i, c_i \le 10^9$, $1 \le s_i \le 2000$). The javelins are given in the order they are proposed by the machine.

Output

The first line of the output should contain two integers c and k — the total cost and the number of the javelins in the optimal set. The second line should contain k integers — the identifiers of the chosen javelins. These identifiers should be provided in increasing order, as they are proposed by the machine.

If the required sequence of javelins does not exist, output "−1". If there are several optimal sequences, output any of them.

Examples

standard input	standard output

2 2
1 1 1
2 3 2
	2 1
	2

2 4
1 1 1
2 3 2
	-1

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

Problem K. Misha's Product

Time limit:	1 second
Memory limit:	512 megabytes

Misha really loves to invent new operations. Recently he has invented the new operation, that he has named *Misha's product* after himself.

Misha's product of the set of different integer numbers $a_1, a_2, \ldots, a_n$ is defined as follows. Consider all possible ordered pairs (a_i, a_j) of different numbers from this set. For each pair write these numbers one after another without a space, and get a new number b_{ij}. Misha's product of the initial set is the sum of all values b_{ij}.

Help Misha to calculate Misha's product for the given set of integers. Misha wants to calculate it modulo $10^9 + 7$.

Input

The first line contains one integer n — the number of elements in the set ($2 \le n \le 10^5$).

The second line contains n different integers $a_1, a_2, \ldots, a_n$ — numbers from the set ($1 \le a_i \le 10^8$).

Output

Output one number — Misha's product of the given set of numbers, modulo $10^9 + 7$.

Example

standard input	standard output
3 1 3 10	668

Note

There are six possible pairs in sample test, the following values b_{ij} are obtained: 13, 31, 101, 103, 110, 310, their sum is equal to 668.

Problem L. Prime Suffixes

Time limit:	1 second
Memory limit:	512 megabytes

Let us call a number y a *suffix* of a number x, if we can remove digits from the beginning of a decimal representation of x to get y. Note that if there are no zeroes in a decimal representation of x, all suffixes of x contain no leading zeroes. For example, suffixes of 283 are 283, 83 and 3.

An integer is called prime, if its has exactly two positive integer divisors. The number 1 is not prime — it has only one positive integer divisor.

Senya likes prime numbers that have no zeroes in their decimal representation, such that all of their suffixes are also prime.

You are given integers a and b. Help Senya to find out, how many integers between a and b are there that he likes.

Input

Input contains two integers a and b ($1 \le a \le b \le 10^{11}$).

Output

Print the number of primes between a and b inclusive, such that they have no zeroes in their decimal representation, and if any number of their leading digits are removed, the resulting number is still prime.

Examples

standard input	standard output
4 13	3
101 109	0
281 286	1

Note

In the first example Senya likes integers 5, 7 and 13.

In the second example all integers in the range contain 0 in their decimal respresentation.

In the third example Senya likes integer 283, since 283, 83 and 3 are all prime.

Problem M. Restoring the Sequence

Time limit:	1 second
Memory limit:	512 megabytes

It's school time, and today the first graders are learning about positive integers. The teacher wrote numbers from 1 to n in ascending order on the blackboard, for demonstration.

Unfortunately, young hooligan Kolya from the 11-th grade ran into the classroom and ruined the beautiful sequence written on the blackboard. Standing before the principal he repented and claimed that he only erased one number from the sequence.

Help the teacher to figure out whether this can be true, and if it can, which number was erased by Kolya.

Input

The first line of input contains one integer n — the number of integers written by the teacher on the blackboard ($2 \le n \le 1000$).

The second line of input contains the number m — the number of integers on the blackboard after Kolya's actions ($1 \le m \le 1000$).

The third line contains m integers $a_1, a_2, \ldots, a_m$ — the integers on the blackboard after Kolya's actions, in the order they are written on the board ($1 \le a_i \le 1000$).

Output

If Kolya's explanations are plausible, and the numbers written on the blackboard could be obtained from the sequence $1, 2, \ldots, n$ by erasing exactly one number, output the word «`Yes`» at the first line. The next line must contain the number that was erased by Kolya.

Otherwise output one word «`No`».

Examples

standard input	standard output

`4`
`3`
`1 3 4`
	`Yes`
	`2`

`4`
`3`
`3 3 3`
	`No`

`4`
`2`
`1 2`
	`No`

`4`
`4`
`1 2 3 4`
	`No`

`4`
`3`
`4 3 1`
	`No`