Problem A. Alice, Bob and Tree

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Alice lives in the forest, so nothing new that she have the tree with 99 edges connecting vertices labeled from 1 to 100. She wants to tell the integer X to the Bob.

Alice sequentially adds K extra edges to the tree such that there will be no self-loops nor multiple edges at any moment of the process.

After the process of edges adding is finished, vertices of the graph (i.e. 1 to 100) are renumerated randomly by the jury program and then the extra K edges are sent to Bob.

Bob lives in the desert, so he knows nothing about the initial tree structure. But his task is to reconstruct an integer X sent by Alice.

Input

Your solution will be ran twice independently. First time it will be ran for the data file for Alice, second time it will be ran for the data file for Bob, produced by jury program using the result of the first run. You are not allowed to pass extra data between the runs.

First line of the input contains one string — "`Alice`", if it is data for the first run, and "`Bob`", if the data are for the second run.

The data for Alice contain 100 lines. First 99 are denoting the Alice's tree. i-th of those 99 lines contains two integers c_i and d_i between 1 and 100 inclusively — indices of the vertices connected by that edge. You may assume that the graph is a tree. 100-th line contains one integer X that should be passed to Bob ($0 \le X \le 10^{18}$).

The data for Bob starts with line containing one integer K. i-th of the following K lines contains two integers a_i and b_i ($1 \le a_i, b_i \le 100$, $a_i \ne b_i$) and denotes the extra edges after shuffling, received by Bob.

Output

At the first run (Alice part) in the first line print one integer $0 \le K \le 100$ — number of the added edges. In each of the following K lines print the extra edges — two integers a_i and b_i in each ($1 \le a_i, b_i \le 100$, $a_i \ne b_i$). If you add more than 100 edges or introduce self-loops or multiple edges, your solution will be considered wrong.

If the output data are for Bob, print one integer — value of X.

Example

standard input	standard output
Alice 10 46 58 13 62 55 96 more lines for the tree 1234567890987	3 2 3 1 3 99 100
Bob 3 5 6 1 2 6 12	1234567890987

Note

The full data sample can be found in the `Samples ZIP` archive of the ejudge system.

Problem B. Build The Network

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

A network at the military base consists of several nodes and bidirectional cables, connecting them with the next properties.

- Each cable is connecting exactly two nodes.
- Cable cannot connect the node with itself.
- Any two nodes are connected directly with no more than one cable.
- The information between any two nodes can be passed through one or more cables and intermediate nodes (i.e. any two nodes are linked by this network).

Sergeant Pepper wants the network to be in the **perfect order**, namely, he wants to have exactly one node with exactly one cable connected to it, exactly two nodes with exactly two cables connected to it and so on till N, i.e. for any $1 \le i \le N$ the network must contain exactly i nodes with exactly i cables connected to it. Sergeant does not allow any other nodes and cables except for that.

Your task is to build perfect order network for given integer N or consider that it is impossible to do it.

Input

Input contains one integer N ($1 \le N \le 239$) — the parameter of the network.

Output

If it is impossible to build perfect order network for given N, print -1. Otherwise, list all the cables in the network. Each cable must be listed at the new line, description of one cable must contain two 1-based indices of the nodes connected by the cable.

Example

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

Problem C. Colored Points

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

The plane is painted in two colors in the following way: for any positive integer n the segments $((n, n)$ $(n, -n))$, $((n, -n)$, $(-n, -n))$, $((-n, -n)$, $(-n, n))$ and $((-n, n)$,$(n, n))$ are painted red; the point $(0, 0)$ is painted red; all the other points are painted white.

Given the segment connecting the points S and F inslusively, count number of the red points at this segment.

Input

First line of the input contains four integers x_s, y_s, x_f, y_f — coordinates of the points S and F respectively $(-10^9 \le x_s, y_s, x_f, y_f \le 10^9)$.

Output

Print one integer — number of red points at the segment SF. If the segment contains infinite number of the red points, print «oo» instead.

Examples

standard input	standard output
2 0 2 0	1
2 0 2 1	oo
4 2 -2 -1	7

Problem D. Dominating Powers

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

Given set of the positive integers $a_1, a_2, \ldots, a_n$ and two positive integers m and g.

Your task is to calculate $b_1, b_2, \ldots, b_n$ where b_k is defined as

$$b_k = g^{a_1 \cdot a_2 \cdot \ldots \cdot a_{k-1} \cdot a_{k+1} \cdot a_{k+2} \cdot \ldots \cdot a_n} \bmod m$$

I.e. the g powered to the product of all a_i, except for a_k, modulo m.

Input

First line of the input contains three positive integers n, m, g ($1 \le n \le 10^5, 1 \le g < m \le 10^9$).

Second line contains set of the positive integers $a_1, a_2, \ldots, a_n$ ($1 \le a_k < m$).

Output

Print n non-negative integers $b_1, b_2, \ldots, b_n$ ($0 \le b_k < m$).

Example

standard input	standard output
3 10 2 4 2 5	4 6 6

Note

The calculations from the first sample:

$b_1 = g^{a_2 \cdot a_3} \bmod m = 2^{2 \cdot 5} \bmod 10 = 1024 \bmod 10 = 4$
$b_2 = g^{a_1 \cdot a_3} \bmod m = 2^{4 \cdot 5} \bmod 10 = 1048576 \bmod 10 = 6$
$b_3 = g^{a_1 \cdot a_2} \bmod m = 2^{4 \cdot 2} \bmod 10 = 256 \bmod 10 = 6$

Problem E. Equivalence

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

There are n points in the plane. Two circles are equivalent if they contain (inside or on the border) the same set of the given n points. For the given set of points, how many non-equivalent circles are there?

Input

The first line contains the integer n ($0 \le n \le 500$).

Each of the next n lines contains two integers x_i and y_i — coordinates of i-th point ($-30\,000 \le x_i, y_i \le 30\,000$).

It is guaranteed that all points are distinct, no three points lie on a common line and no four points lie on a common circle.

Output

Print the answer to the problem.

Examples

standard input	standard output
1 0 0	2
4 0 0 1 0 0 1 -1 -1	15

Problem F. Faster Exponentiation

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Consider the following "usual" way of the fast exponentiation — the function `fastpow`:

- if $n = 0$, return 1
- if n is odd, return `a*fastpow(a,n-1)`
- if n is even, calculate `fastpow(a,n/2)` and square it.

Alice checked if there exists some n that the given algorithm is not the fastest one. She noticed that, for example, such an integer is 27: `fastpow` uses 7 multiplications, while Alice managed to find 6. Let ve need to calculate v_0^{27}. Then

1. $v_1 = v_0 \cdot v_0$
2. $v_2 = v_1 \cdot v_0$
3. $v_3 = v_2 \cdot v_2$
4. $v_4 = v_3 \cdot v_2$
5. $v_5 = v_4 \cdot v_4$
6. $v_6 = v_5 \cdot v_4 = v_0^{27}$

Now Alice wants for the given D find such n and such an way of exponentiation, that is faster than `fastpow` exactly by D exponentiations. Because she likes C++, she wants n to fit the **unsinged long long** data type.

Input

Input contains one integer D ($2 \le D \le 57$).

Output

In the first line print two integers — power n and number of multiplications P ($0 \le N \le 2^{64} - 1$, $1 \le P \le 1000$).

Then print P lines. i-th of those lines shall have the form `v`i=`v`k*`v`l, where $1 \le i \le P$, $0 \le k, l < i$.

`v0` denote the integer that shall be raised in the n-th power. The sequence v shall be a **strictly increasing**, i.e for all $1 \le i \le P$ the inequality `v`i > `v`$(i-1)$ shall be hold.

The value of `v`P shall coincide with v_0^n, and difference between number of multiplications in your algorithm P and in classical `fastpow` shall be exactly D.

Example

standard input	standard output
1	27 6 v1=v0*v0 v2=v1*v0 v3=v2*v2 v4=v2*v3 v5=v4*v4 v6=v5*v4

Problem G. Generate The Triangle

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Given positive integer r.

Your task is to find three integers a, b and c such as:

- There exists a triangle with sides a, b and c and non-zero area.
- The triangle is not right, i.e. $a^2 + b^2 \neq c^2$, $a^2 + c^2 \neq b^2$ and $b^2 + c^2 \neq a^2$ are held simultaneously
- The radius of the inscribed circle of this triangle is equal to r.

Input

Input contains one integer r ($2 \leq r \leq 10^9$).

Output

Print three integers a, b and c — answer to the problem. The integers shall be not less than 1 and not greater than $5 \cdot 10^{18}$.

Example

standard input	standard output
3	17 15 8

Note

Note that the sample shows only format — such triangle is right, so it will not be accepted.

Problem H. Haxi

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

The Byteland-City consists of n junctions and m bidirectional roads, connecting those junctions. The value of each road is a non-negative integer.

Hackers are fond of bit operations, so the cost of a Haxi (special taxi for hackers) ride is calculated as a bitwise XOR of values of all roads used. For example, if the ride used roads with values 3, 4 and 6 the price will be $3 \oplus 4 \oplus 6 = 1$. If the car uses the same road $k > 1$ times, this road is included in the price calculation k times.

Neo plans several travels by Haxi. For each travel starting and ending junctions are given, and Neo wants each travel to be as cheap as possible. Note that the taxi is allowed to go through ending junction arbitrary number of times before it will stop there.

Help Neo calculate the smallest possible price for each of the rides.

Input

The first line of the input contains two integers n and m ($1 \le n, m \le 2 \cdot 10^5$) — the number of junctions and the number of bidirectional roads, respectively.

Each of next m lines describes one road and contains three integers a, b and c ($1 \le a < b \le n$; $0 \le c \le 10^9$) — indices of the junctions connected by this road and its value. Each pair of junctions is connected by at most one road. It's guaranteed that it's possible to get from a junction to any other junction using only the provided roads.

The next line contains one integer q ($1 \le q \le 2 \cdot 10^5$) — the number of routes. Each route is described by two integers s and e ($1 \le s, e \le n$) — starting and ending junctions of the route.

Output

For each route print one integer — the minimum possible price for traveling from starting junction to ending junction by Haxi.

Example

standard input	standard output
5 5 1 2 3 2 3 9 3 4 2 4 5 6 3 5 7 1 1 2	0
4 6 1 2 7 1 3 2 1 4 0 2 3 13 2 4 19 3 4 31 3 4 1 4 2 4 3	0 6 2

Note

In the first sample, route 1, 2, 3, 4, 5, 3, 2 allows to achieve overall price of 0.

Problem I. iMarket

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

There are stocks of n types available for purchase at an electronic stock market called iMarket. One stock unit of type i costs c_i cent. Immediately after the purchase, a stock unit of type i starts generating profit of g_i cent per second.

Note that on iMarket one is allowed to purchase non-integer amounts of units of each stock, with costs and profits scaling accordingly. Furthermore, profits are gained gradually and uniformly over any time interval. For example, if Scrudge bought 0.7 stock units with a profit of 1 cent per second, then over the next 0.4 seconds it would generate 0.28 cent.

Any volume of any stock is considered *passive* immediately after purchase. However, turning passive stocks into *active* may help decrease costs on buying some other stock types. Formally, for certain pairs of stock types a, b we have that a *supports* b. In this case, whenever one purchases x units of stock type b, they can *activate* x units of stock type a to cut the costs of this purchase in half. Of course, the activated stock volume must be passive prior to the activation, and can never be activated again to save on a subsequent purchase.

For any stock type b there is at most one stock type a such that a supports b. Furthermore, cyclic chains of support between stock types are disallowed.

At the start of the trade on iMarket Scrudge has e cent and no stocks. He wants to perform some trade operations so that to reach profits of p cent per second as soon as possible. Note that Scrudge can perform operations as frequently as he likes.

Input

The first line of the input contains integers n, e and p ($1 \le n \le 1000$, $1 \le e, p \le 10^9$) — the number of stock types, the initial amount of money Scrudge has, and the target profit per second, respectively.

The i-th of the next n lines describes the stock type i and contains three integers c_i, g_i and s_i, ($1 \le c_i \le 10^9$, $0 \le g_i \le 10^9$, $0 \le s_i \le n$) — the price of one stock unit, profit per second per stock unit, and the index of a stock type that supports the type i, respectively. If $s_i = 0$, then the type i is not supported by any other type. It is guaranteed that at least one of g_i is positive.

Output

Print one integer — minimum time (in seconds) Scrudge needs to reach profits of p cent per second. The time must be rounded up to the nearest integer (i.e. if Scrudge needs 3.14 seconds, you should print 4).

Example

standard input	standard output
1 1 1000000 200 100 0	30
2 1 1000000 200 100 0 2 1 1	29
2 1 1000000 200 100 2 2 1 0	14

Problem J. John And Template

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Vasya has an array of 2^n elements indexed by binary n-digit strings. Initially the array is filled with zeros. Then he did the following operation m times: he chose a template (which is a string of zeros, ones and question marks) and assigned the same value to all elements which indices satisfied the template. Now he wants to find the sum of all elements in the array. Help him.

Input

The first line of input contains two integers n and m ($1 \le n, m \le 32$).

Each of the next m lines contain a string s and an integer q — the template and the value for the current operation ($0 \le q \le 10^9$).

Output

Print the answer to the problem.

Examples

standard input	standard output
3 2 ?0? 2 1?? 3	16