Problem A. Annihilation

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

There are n particles and m anti-patricles in the reactor. Initially, they all are at rest in some distinct points of three-dimensional space.

Scientists can turn on a force field inside the reactor. The characteristic of the field is a unit vector v which can be chosen arbitrarily.

Under the effect of a force field, each particle moves with unit speed in direction of vector v, and each anti-particle moves with unit speed in the direction of vector $-v$.

When a particle and an anti-particle collide, they annihilate and disappear. One unit of energy is released.

Scientists want to know how many ways are there to choose the vector v such that exactly k units of energy are released after all collisions take place. Help them find that number.

Input

The first line of input contains an integer n: the number of particles ($1 \le n \le 500$).

Then follow n lines; i-th of them consists of three integers x_i, y_i and z_i: the coordinates of i-th particle.

The next line of input contains an integer m: the number of anti-particles ($1 \le m \le 500$).

Then follow m lines; i-th of them consists of three integers x'_i, y'_i and z'_i: the coordinates of i-th anti-particle.

All coordinates are integers from 1 to 10^9, inclusive. It is guaranteed that no two points in the input are the same.

Output

For each k from 1 to $\min(n, m)$, print two integers on a separate line. The first of them must be this number k, and the second one must be the number of ways to choose a unit vector v so that the total amount of energy released will be k. If for some k the number of ways is zero, you must not print the respective line. The lines must be printed in the order of increasing k. See examples for clarification.

Examples

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

Problem B. Life on Mars

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

Curiosity rover recently sent a very interesting photo from Mars to Earth. This sensational photo indicates that life is finally found on Mars.

The photo contains alien bacteria. Each bacterium can be viewed as a segment on a plane. With the help of the photo, scientists have studied structure of the bacteria and got some interesting conclusions.

For example, the scientists learned that the bacteria continuously grow in length and do not change their orientation in space. Each bacterium grows in only one side: one of its two ends moves with unit speed, effectively increasing its length with that speed, and the other end remains still.

Scientists are eager to know what happens when some two bacteria bump into each other. Unfortunately, they have only one photo, so they do not know which end of each bacterium grows and which one remains still.

The next photo will not come any time soon, but scientists want to estimate the time of the first collision as soon as possible. Help them by finding the minimal and the maximal possible time to pass before the first collision takes place. Remember that each bacterium on the photo has exactly two growth scenarios (by increasing its length with unit speed along a straight line from one of their two ends), and growth scenarios of different bacteria are not found to be dependent in any way.

Input

The input contains one or more test cases.

Each test case starts with an integer n: the number of bacteria on the photo ($1 \le n \le 300$). Then follow n lines; i-th of these lines contains four integers x_i, y_i, x'_i and y'_i: the coordinates of endpoints of i-th bacterium. All coordinates are integers not exceeding 1000 by absolute value. It is guaranteed that initially, each bacterium has length from 1 to 50, inclusive, and no two bacteria have common points.

The input is terminated by a single line containing a single zero. This is not a test case and should not be processed. It is guaranteed that the total sum of n in the input does not exceed 300.

Output

For each test case, print two space-separated real numbers on a separate line. The first of them must be the minimal time to pass between the moment when the photo was taken and the first moment when some two bacteria collide. The second one must be the maximal such time. The numbers must be printed with absolute or relative error not exceeding 10^{-6}. If some of the numbers can be arbitrarily large, print "`infinity`" instead of that number.

Examples

standard input	*standard output*
3 -3 -2 0 -4 2 3 -5 1 4 -2 5 2 5 -1 -1 1 1 -1 2 1 2 -1 -2 1 -2 2 -1 2 1 -2 -1 -2 1 0	3.6400549446 infinity 1.0000000000 1.4142135623

Problem C. Binary Hyperparallelepiped

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 megabytes

The Thue-Morse sequence is a binary sequence obtained in the following way. Start with a string $S_0 =$ "0" and consider the following procedure f: given a string S, produce $f(S)$ by replacing each "0" by "01" and each "1" by "10". Apply this procedure to S_0 to get $S_1 = f(S_0)$, then proceed by setting $S_2 = f(S_1)$, and so on. The infinite sequence S_∞ obtained by applying this procedure an infinite number of times is the Thue-Morse sequence.

The first few S_i are:

$$0 \to 01 \to 0110 \to 01101001 \to 0110100110010110$$

Note that the definition is correct since it turns out that each subsequent string contains the previous one as a prefix.

This sequence can easily be generalized to higher dimensions. For example, for two dimensions, the procedure should replace each 0 with $\begin{smallmatrix}0&1\\1&0\end{smallmatrix}$ and each 1 with $\begin{smallmatrix}1&0\\0&1\end{smallmatrix}$. The first few results in this case are:

$$[0] \to \begin{bmatrix}0&1\\1&0\end{bmatrix} \to \begin{bmatrix}0&1&1&0\\1&0&0&1\\1&0&0&1\\0&1&1&0\end{bmatrix} \to \begin{bmatrix}0&1&1&0&1&0&0&1\\1&0&0&1&0&1&1&0\\1&0&0&1&0&1&1&0\\0&1&1&0&1&0&0&1\\1&0&0&1&0&1&1&0\\0&1&1&0&1&0&0&1\\0&1&1&0&1&0&0&1\\1&0&0&1&0&1&1&0\end{bmatrix}$$

For n-dimensional case, the procedure replaces 0 and 1 by n-dimensional hypercubes $2 \times 2 \times \ldots \times 2$ with sides parallel to coordinate axes. The 2^n values inside them are arranged in "chessboard order": each two n-dimensional unit hypercubes sharing a $(n-1)$-dimensional face contain different values. There are exactly two such hypercubes. Thus, each 0 is replaced by the hypercube with element 0 closest to the origin, while each 1 is replaced by the other such hypercube.

You are given an n-dimensional space where unit hypercubes with positive coordinates are filled with 0s and 1s as described above. At the beginning of the filling process, the unit hypercube with coordinates $(1, 1, \ldots, 1)$ was filled with 0. After the first step, the unit hypercubes with all coordinates from 1 to 2 were filled, after the second step, we filled all unit hypercubes with coordinates from 1 to 4, and so on. Consider a hyperparallelepiped consisting of all unit hypercubes with coordinates $(x_1, x_2, \ldots, x_n)$ such that $a_i \le x_i \le b_i$ for $1 \le i \le n$. Your task is to count the number of 1s in this hyperparallelepiped.

Input

The first line of input contains an integer n: the number of dimensions ($1 \le n \le 100$).

The second line contains n integers a_1, a_2, ..., a_n.

The third line contains n integers b_1, b_2, ..., b_n.

It is guaranteed that $1 \le a_i \le b_i \le 10^{18}$ for each i such that $1 \le i \le n$.

Output

Print one integer: the number of 1s in the given hyperparallelepiped.

Examples

standard input	*standard output*
2 1 1 8 8	32
4 5 6 7 8 10 12 14 16	1512

Problem D. Counting Rectangles

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 megabytes

The floor of the King's palace is a rectangle of size $n \times m$. It is covered by $n \cdot m$ square tiles of unit size. On each tile, there is a *diamond* drawn on it: a square with sides joining centers of the tile's sides.

The King decided to check what his advisor is capable of, and so the King ordered him to count the rectangles in this picture. The sides of these rectangles should be parallel either to tiles' sides or to diamonds' sides. Two rectangles are considered different if their sets of vertices differ. As the requested number can be fairly large, the King will be satisfied with an answer modulo an integer p.

If by the next morning the advisor does not find the correct answer, he will be executed. Help the advisor answer the King's question.

Input

The first line of input contains three integers n, m and p ($1 \le n, m \le 10^{18}$, $1 \le p \le 10^9 + 7$). Note that p is not necessarily prime.

Output

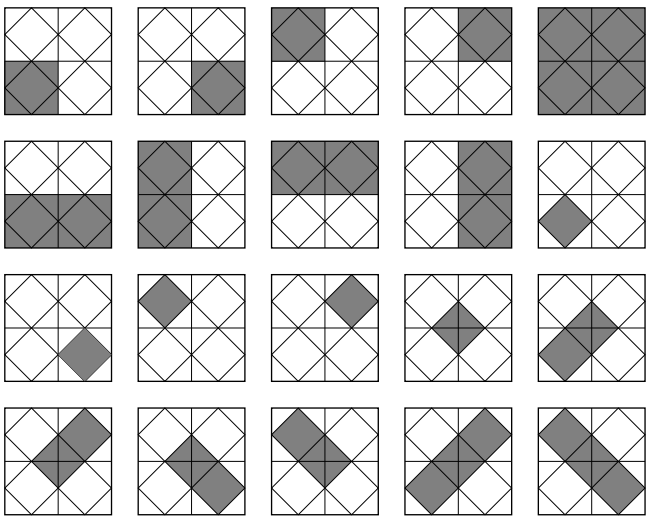
You must print one integer: the total number of rectangles modulo p.

Examples

standard input	*standard output*
2 2 100500	20
777777 888888 999999	222222

Note

For the first example, all possible rectangles are shown on the picture below.



Problem E. Hacker

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Adrian is trying to crack a password. He knows that the password is a number of N digits written in some base K (there are K^N total passwords). He also knows that the password he is trying to crack has an interesting property: every prefix of length L ($1 \leq L < N$) differs from the suffix of length L. For example, `0111` is a possible password while `01001` is not (the prefix of length 2 and the suffix of length 2 are both `01`).

Adrian managed to get a glimpse of the password so he can tell you for each position in the password either that it is a certain digit ($0, 1, \ldots, K-1$) or that he does not know anything about it. Your task is to help Adrian find out the number of possible passwords respecting the information he knows so far.

Input

The first line of input contains two integers N and K ($1 \leq N \leq 200$, $1 \leq K \leq 10$). The second line contains a string of length N representing information about the password Adrian already knows. Each character is either a digit if he managed to see it or a question mark (‘`?`’) otherwise.

Output

A single line containing the total number of possible passwords, modulo 666 013.

Example

standard input	standard output
4 2 1??0	3

Note

The three possible passwords are `1000`, `1100` and `1110`.

Observe that `1010` is not a valid password because its prefix of length 2 is equal to its suffix of length 2.

Problem F. Network Instability

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

Vasya is a network administrator in a large company named Glitchware. There are a total of n computers in the network, and some pairs of computers are directly connected by network cables. There are m such connections. No cable connects a computer with itself. There is no more than one cable between any pair of computers.

Each computer in Glitchware has a special piece of software installed and maintained by Glitchware developers, named NetBug. New versions are released almost daily, and update system behavior is totally incomprehensible. For instance, an update can occur on only some computers and does not always install the latest version available. An update can even install the same version of NetBug or an older version as well. Computers exchange data, and an exchange can happen only between two computers directly connected by a cable.

After several months of work, Vasya discovered that the most common reason for network instability is incompatibility of NetBug versions installed on Glitchware computers. More precisely, when two of computers connected by a cable have different versions of NetBug installed, this connection is *unstable*: trying to exchange data between them can result in a network failure. The more such unstable connections, the more the probability of network failure. On the other side, exchanging data with the same versions of NetBug does not usually lead to network errors.

Vasya's goal is to prevent and fix network errors. But he does not have to be at his working place each day. Vasya wants to be there only when the probability of a network error is high enough. To plan ahead, he got himself the NetBug update schedule for the next year. However, finding that probability proved to be a difficult task.

Help Vasya find the number of unstable connections after each software update according to the schedule.

Input

The first line of input contains two integers n and m ($1 \le n, m \le 10^5$).

The second line contains n integers v_1, v_2, ..., v_n: versions of NetBug initially installed on Glitchware computers ($1 \le v_i \le 10^5$).

Then follow m lines; i-th of them contains a_i and b_i: the numbers of computers connected by i-th cable ($1 \le a_i, b_i \le n$, $a_i \ne b_i$). It is guaranteed that no two computers are connected by more than one cable.

The next line contains an integer q: the number of planned software updates ($1 \le q \le 10^5$).

Then follow q lines; i-th of them contains two integers c_i and v'_i: the number of computer which gets a NetBug update and the NetBug version on that computer after the update. ($1 \le c_i \le n$, $1 \le v'_i \le 10^5$). Updates are listed in chronological order, and no two updates happen simultaneously.

Output

Print q lines; i-th of them must contain a single integer: the number of unstable connections immediately after i-th software update.

Examples

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

Problem G. Interplanetary Durak

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Interplanetary Durak is a card game played with a deck of 2^n cards, q of which are trump cards.

Before the game, the deck is shuffled. One shuffle step is made according to the following procedure. First, the top card of the deck is moved to the bottom. Next, the second top card is moved to the bottom. In the resulting deck, the third card from the top is moved to the bottom, and so on. This process continues until $(2^n - 1)$ cards have been moved to the bottom of the deck.

Shuffling the deck consists of s shuffle steps done in sequence. Shuffling is performed by specially designed shuffling machines, so the whole process is really fast.

Cheater Kvo does not like to lose the game. In order to win, it suffices for him to know where exactly each trump card is located in the deck. Kvo is attentive and able to notice and memorize the position of each trump card before the shuffle. However, he is not fast and intelligent enough to calculate their positions after the shuffle.

The Interplanetary Durak Intergalactic Championship will be held soon. Kvo offered you a share of the grand prize if you help him win. He will be secretly sending you SMS (short messages) with locations of trump cards. Time penalties are strict on the Championship, so you will have to answer his queries really fast.

Input

The input consists of one or more SMS. The first line of each SMS contains three integers n, s and q ($1 \le n \le 50$, $0 \le s \le 10^9$, $1 \le q \le \min(2^n, 10^5)$). The next line contains q distinct integers p_1, p_2, ..., p_q ($1 \le p_i \le 2^n$).

Input is terminated by a line containing three zeroes. It is not an SMS and should not be processed. The total number of integers in the input does not exceed $10^5 + 6$.

Output

You must print the answer for each SMS on a separate line. The line must start with "`SMS` X`:`" where X is the number of SMS in the input (starting from 1). After that, you must print a space and q integers separated by spaces: the positions of trump cards after shuffling. The order of trump cards in the output must correspond to their order in the input.

Examples

standard input	*standard output*
3 1 8 1 2 3 4 5 6 7 8 3 2 8 1 2 3 4 5 6 7 8 3 3 8 1 2 3 4 5 6 7 8 0 0 0	SMS 1: 8 1 5 2 7 3 6 4 SMS 2: 4 8 7 1 6 5 3 2 SMS 3: 2 4 6 8 3 7 5 1

Problem H. K-graph

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

You are given a directed acyclic graph with N vertices and M edges. You are also given a positive integer K. Each of the graph's edges has an integer nonnegative value assigned to it.

In this problem, we will consider paths of length greater or equal to K. Let S_{Max} be the sum of the K greatest edges on such a path and S_{Min} be the sum of the K smallest edges. The cost of a path is defined as $S_{Max} - S_{Min}$. Your task is to find a path of maximum cost. You are not required to output the path, you should only compute its cost.

Input

On the first line of input, you can find the integers N, M and K ($1 \le N \le 300$, $0 \le K \le 300$, $0 \le M \le 900$). Each of the next M lines contains three integers a, b, c representing the fact that there is an edge in the graph from a to b with value c ($0 \le c \le 10^6$).

Output

Print the required value. If there is no chain with at least K edges, print -1 instead.

Example

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

Problem I. k-th Edge in a Subtree

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

Consider a tree of n vertices numbered by consecutive integers from 1 to n. There is an integer value from 1 to 10^9 written on each edge.

You have to answer the following queries: for some sub-tree, write down the values on its edges in non-decreasing order and find k-th of these values.

Each subtree is defined by some subset of z vertices. The subtree is union of all pairwise shortest pathes between these vertices. In other words, an edge belongs to the subtree if and only if it belongs to at least one of these shortest paths.

Input

The first line of input contains an integer n: the number of vertices in the tree ($1 \le n \le 10^5$).

The next $(n-1)$ lines contain three integers each; i-th of them contains integers a_i, b_i and c_i: the numbers of vertices connected by i-th edge and the number written on it ($1 \le a_i, b_i \le n$, $a_i \ne b_i$, $1 \le c_i \le 10^9$).

The next line contains an integer q: the number of queries ($1 \le q \le 10^5$).

Each of the next q lines describes one query. Each query description consists of a single line starting with a description of a set. A set of z vertices is denoted by z distinct integers v_1, v_2, ..., v_z followed by a single zero ($1 \le z \le n$, $1 \le v_j \le n$). Then follows an integer k ($1 \le k \le n-1$). See examples for further clarification.

It is guaranteed that the sum of all z in the queries does not exceed 10^5.

Output

For each query, print one integer on a separate line: value of the k-th edge of the given subtree. If there are less than k edges in the respective subtree, print -1 instead.

Examples

standard input	*standard output*
6 1 2 4 2 3 2 3 4 2 3 5 1 6 5 3 5 1 2 5 6 0 2 2 4 6 0 4 6 4 0 1 1 3 0 3 3 0 5	2 3 1 -1 -1

Problem J. Reverse Sort

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

You are given an integer sequence A of length N. Your goal is to sort this sequence by using the operation `Reverse(left, right)`. As expected, this operation takes the subsequence $(A_{left}, A_{left+1}, \ldots, A_{right})$ and reverses it. The cost of such an operation is equal to the length of the subsequence, namely $(right-left+1)$. You are not required to sort the sequence with minimum total cost, however, the total cost should be less or equal than $4 \cdot 10^6$ for each input file.

Input

Your program should read the integer N ($1 \le N \le 32 \cdot 10^3$) on the first line of input and the N integers that form sequence A on the second line ($1 \le A_i \le 32 \cdot 10^3$).

Output

Your program must print K lines where K is the number of operations you will use. Each line must consist of two integers, $left$ and $right$, meaning that you choose to reverse the subsequence ($left$, $left+1$, ..., $right$).

Example

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

Note

After the first `Reverse`, the sequence becomes `1 5 4 3 2 2`. After the second `Reverse`, the sequence becomes `1 2 2 3 4 5` and is indeed sorted.

Problem K. Sequence

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Consider the sequence 1, 2, 3, 3, 4, 5, 5, 6, 6 .... You have probably spotted the pattern already, but in case you have not, it is an increasing sequence in which each positive integer X has frequency equal to the number of bits set to 1 in its binary representation. We are interested in finding the K-th term of this sequence for some large integer K.

Input

The only line of input contains the integer K written in hexadecimal representation ($1 \le K \le 16^{200\,000}$). We use capital letters '`A`–`F`' for hexadecimal digits greater than 9.

Output

You must print the required number, also written in hexadecimal representation.

Examples

standard input	*standard output*
4	3
17	D
3D961	8447
96D271	F41F4