Problem A. Admin's Quest

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

Dmitry is working as a system administrator at university. He is now facing at a big trouble: a number of computers under his management have been infected by a computer virus. Unfortunately, anti-virus system failed to detect this virus because it was very new.

Dmitry has identified the first computer infected by the virus and collected the records of all data packets sent within his network. He is now trying to identify which computers have been infected. A computer is infected when receiving any data packet from any infected computer. The computer is not infected, on the other hand, just by sending data packets to infected computers.

It seems almost impossible for him to list all infected computers by hand, because the size of the packet records is fairly large. So he asked you for help: write a program that can identify infected computers.

Input

Input consists of no more than 30 test cases.

First line of each test case contains two integers N and M ($0 < N \le 2 \cdot 10^4$, $0 \le M \le 2 \cdot 10^4$), where N is the number of computers and M is the number of data packets; then M descriptions of packets follow. Each description consists of three integers t_i, s_i and d_i — time then packet is sent, source computer id and destination computer id, respectively ($0 \le t_i \le 10^9$, $1 \le s_i, d_i \le N$, $s_i \ne d_i$, all t_i are different).

The first infected computer have id 1.

The input is terminated by a line containing two zeros. This line is not a part of any test case and should not be processed.

The size of the input does not exceed 5 MiB.

Output

For each test case, print the number of computers infected by the computer virus.

Example

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

Problem B. Bishops, Rooks and Queens

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

On the $n \times m$ chessboard some white queens, white rooks and white bishops are placed. Your task is to find maximum number of black pieces which can be put on the board so no black piece is under white's attack.

Input

First line of the input contains two integers: n — number of rows and m — number of columns ($1 \le n, m \le 10^9$). Next line consists of one integer k – number of pieces ($1 \le k \le 20$). Then k lines follow, each containing three integers t, a and b ($1 \le t \le 3$, $1 \le a \le n$, $1 \le b \le m$). Here $t = 1$ for a rook, $t = 2$ for a bishop and $t = 3$ for a queen. a and b are row and column, respectively, where piece is placed. It is guaranteed that no two pieces share the same field.

Output

Print one intger — maximum number of black pieces which can be put on the board so no black piece is under white's attack.

Example

standard input	standard output
8 8 5 3 7 6 1 4 7 1 5 2 2 2 3 2 2 4	13

Problem C. Cargo Transportation

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

On a Far Byteland island, there are two towns Northburg and Southburg, with steep mountains between them. The Byteland Transportation Company plans to run a ropeway cargo communication between Northburg and Southburg. They want maximize the revenue by loading as much package as possible on the ropeway car.

The ropeway car looks like the following. It's L length long, and the center of it is hung from the rope. Let's suppose the car to be a 1-dimensional line segment, and a cargo package to be a point lying on the line. You can put cargo at anywhere including the edge of the car, as long as the distance between the cargo and the center of the car is greater than or equal to R and the car doesn't fall down.

Let's there are N packages; m_i is the weight of i-th cargo package and x_i is coordinate of this package on a car (with center of the car used as origin). The car will fall down when

$$|\sum_{i=1}^{N} m_i x_i| > W.$$

Given list of cargo packages and order in which they are loaded to the car, check if it possible to choose a loading points for them such as while packages are loaded to those points in given order, car will never fall down at any time.

Input

The first line of the input contains four integers N ($1 \le N \le 100$), L ($1 \le L \le 10^5$), W ($1 \le W \le 10^4$), R ($1 \le R \le 4 \cdot 10^4$).

The second line contains N integers $m_0,\ m_1,\ \ldots\ m_{N-1}$ ($1 \le m_i \le 10^5$).

Output

If there is a way to load all cargo in the given order, output "`Yes`" in one line, without the quotes. Otherwise, output "`No`".

Examples

standard input	standard output
3 3 2 1 1 1 4	Yes
3 3 2 1 1 4 1	No

Problem D. Divisibility

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

Given the string consisting of no more than L characters. Each character is a decimal digit. Your task is to delete some characters so the resulting decimal number will be divisible by given integer p and maximal possible.

Note that leading zeroes in the answer are not tolerated.

Input

First line of the input contains non-empty string of length L, consisting of decimal digits. Second line contains integer p, $0 < L \cdot p \leq 10^6$.

Output

Print one integer — answer to the problem or -1, if it is impossible to obtain an integer divisible by p.

Examples

standard input	standard output
36956 6	3696
2017 8	0
333 6	-1

Problem E. Eat Or No?

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

On some planet in the galaxy far far away there're two kinds of living organism, let's say A and B. A may eat B, but if and only if size of predator is strictly bigger than its prey.

For example, let the population of A have sizes $\{8, 1, 7, 3, 1\}$ and population of B have sizes $\{3, 6, 1\}$, then there are 7 pairs of A, B where $A > B$: $(8, 3)$, $(8, 6)$, $(8, 1)$, $(7, 3)$, $(7, 6)$, $(7, 1)$, $(3, 1)$.

Given the size of each organism in populations of A and B, write a program to count how many pair of A, B are there such that A is strictly bigger than B.

Input

The first line of input contains an integer T ($1 \le T \le 30$), the number of test cases follow.

Each case will begin with two integers N ($1 \le N \le 2 \cdot 10^4$) and M ($1 \le M \le 2 \cdot 10^4$), denoting the number of organism A and B respectively. The next line will contain N positive integers represent the size of each A organism. The third line will contain M positive integers represent the size of each B organism. All sizes are not greater than 10^9.

Output

For each case, print on a single line the number of pair A, B such that A is strictly larger than B.

Example

standard input	standard output
2 5 3 8 1 7 3 1 3 6 1 3 4 2 13 7 103 11 290 215	7 1

Problem F. Fill The Board

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

Given a chessboard with N rows and N columns. Alice have infinite number of square blocks 2×2. She can put blocks on in the board aligned to the board cells; blocks cannot overlap.

Two configurations of the blocks are considered the same if one of then can be obtained from another by rotating the board. Otherwise they are considered different. Alice wants to know number of different configurations.

Input

First line of the input contains one integer T — number of the test cases ($1 \le T \le 20$).

Each of next T lines contains one integer N — dimension of the board ($1 \le N \le 20$).

Output

For each test case print one line containing the answer to this test case modulo $10^9 + 7$.

Example

standard input	standard output
3 1 2 4	1 2 12

Problem G. Go Travel!

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

John bought the new car — `Tesla Model S`. It is widely know that it is powered by electricity so instead refueling it must be charged at the special `<<SuperCharger>>` stations.

John plans go to travel from city A to city B on his new car. But because SuperCharger stations are not everywhere now, sometimes it become real problem to reach one city from another.

Given a map of U.S. with SuperCharger stations marked on it, John wants to find some route between A and B on Tesla. Note that he will be happy with any route — not especially optimal. At the start Tesla is fully charged.

Input

First line of the input contains four integers N, M, K, P — number of cities, number of roads between the cities, number of cities with SuperCharger and distance which Tesla may go with fully charged battery ($1 \le N \le 10^5, 1 \le M \le 3 \cdot 10^5, 1 \le K \le N, 1 \le P \le 10^9$).

Second line contains K integers — numbers of the cities with `SuperCharger`. Cities are numbered sequentially from one.

Next M lines describe the roads. Each road is described by three integers a_i, b_i, c_i — numbers of cities and length of the road ($1 \le a_i, b_i \le N, 1 \le c_i \le 10^9$).

John starts in city with id 1 and ends up his travel in city with id N. At the start Tesla is fully charged.

Output

In the first line print one integer T — number of cities in your route. Second line must contain numbers of those cities in the order they are visited by John. Remember that any path where distance between two charges is no more than P is acceptable, but this path cannot contain more than $3 \cdot 10^6$ cities

If the travel is impossible, print -1 instead.

Examples

standard input	standard output
4 4 1 10 2 1 4 11 1 2 9 2 3 5 3 4 5	4 1 2 3 4
6 7 3 5 1 2 3 1 2 1 2 3 1 3 1 1 3 4 4 4 5 1 5 6 1 4 6 2	-1
3 3 0 3 1 2 1 2 3 1 1 3 1	3 1 2 3

Note

In Sample 1 direct road between cities 1 and 4 is too long, so John must use not a direct road.

In Sample 2 John may reach cities to cities $1, 2, 3, 4, 5$, but he will be out of the charge while driving to last city.

Third example shows that not only shortest path is accepted.

Problem H. Hard Numbers

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

Hard number is a numbering sequence that is so troublesome (that's exactly where it gets its name). Hard number is similar to binary number (only consist of zeros and ones) except that the length of the number is important (thus leading zeros are preserved). Note that the "length" of hard numbers means the number of digits in the hard numbers.

To simplify the wording, hard numbers of length L will be written as H_L , which denotes all hard numbers with exactly L digits. Hard numbers (of any length) is an ordered list of numbers. The most basic (smallest) group of hard numbers is H_1 which are `0` and `1` in that order. H_L can be generated from H_{L-1} for any $L > 1$ in the next way:

- Two clones (c_1 and c_2) of H_{L-1} are created. Then operation with id $(L-2) \bmod 8 + 1$ from the list of operations is applied to produce C_1 and C_2.
- H_L is the list of numbers from C_1, followed by the list of numbers in C_2, preserving the order.

The numbered list of operations used to produce C_1 and C_2 from c_1 and c_2:

1. Append a digit zero to the end of all numbers in C_1 and append a digit one to the end of all numbers in C_2.
2. Append a digit zero to the beginning of all numbers in C_1 and append a digit one to the beginning of all numbers in C_2.
3. Append a digit one to the end of all numbers in C_1 and append a digit zero to the end of all numbers in C_2.
4. Append a digit one to the beginning of all numbers in C_1 and append a digit zero to the beginning of all numbers in C_2.
5. Reverse the order of the list of numbers in C_2 and do operation 1 above.
6. Reverse the order of the list of numbers in C_2 and do operation 2 above.
7. Reverse the order of the list of numbers in C_2 and do operation 3 above.
8. Reverse the order of the list of numbers in C_2 and do operation 4 above.

For example, H_2 is generated by applying the first operation on H_1, H_3 is generated by applying the second operation on H_2 and so on... and it will go back to the first operation again after the eighth operation. So, H_9 is generated by applying the eighth operation on H_8, H_{10} is generated by applying the **first** operation on H_9 and so on. The i-th number in H_L (i-th hard number of length L) is denoted as H_{Li}, i.e. $H_{11} = 0$, $H_{12} = 1$, $H_{21} = 00$, $H_{22} = 10$, $H_{23} = 01$, $H_{24} = 11$ etc.

To give you an example of "reverse the order of the list of numbers in C_2" for the fifth to eighth operations, we give the last 5 numbers of H_6: $H_{660} = 110011$, $H_{661} = 101111$, $H_{662} = 100111$, $H_{663} = 101011$, $H_{664} = 100011$.

Your job is to find the i-th hard number of length L or to tell the index i of given hard number (considering that length is obvious from the representation).

Input

First line of the input contains one integer T — number of the test cases ($1 \le T \le 55555$).

Each test case consists of one string. If the first token in the string is 'H', then two integers L and i follow and your job is to find i-th hard number of length L (numeration is 1-based), $L < 64$, $i \le 2^L$.

If the first token is 'I', then one binary string of length 63 or less follows — the hard number.

Output

For the 'H' query, print the binary string — i-th hard number of length L. For the I query print one integer — index i for the given binary string considered as hard number of its length.

Example

standard input	standard output
10	0
H 1 1	110
H 3 6	1000
H 4 13	14
I 1100	11100
H 5 14	16
I 1110	31
I 01010	100010
H 6 1	001110
H 6 20	011100
H 6 32	

Problem I. iGadget Z and Strange Model

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

Eve is the big fan of the Ellpa gadgets. Because the Elppa products are very popular, Eve needs to queue for new gadgets. Today three new gadgets are on sale: iGadget Y, iGadget Y+ and iGadget Z.

Each person in the queue is allowed to pick only one item. No need to ask, Eve wants only iGadget Z. Eve is quite sure that there is enough iGadget Y and iGadget Y+ in the shop and she knows number of remaining iGadget Z left in shop and number N of iGadget fans in queue in front of her.

Eve considers all possible outcomes (i.e. all possible sequences of N choices conforming the iGadget Z limits) **equiprobable**. Help her to calculate her chances to get iGadget Z in this model.

Input

Input consists of no more than 2500 test cases.

Each test case is placed in one line and contains two integers N ($1 \le N \le 50$) and Z ($0 \le Z \le 50$) — number of people in front of Eve and remaining number of iGadget Z, respectively.

Output

For each case, print in a single line the probability in percentage that Eve will get her iGadget Z, with absolute error 10^{-5} or better.

Example

standard input	standard output
2 1	50.0
3 2	76.92308
4 0	0
4 1	33.33333
17 3	33.85214
20 17	99.99974
20 21	100

Note

In sample 1 possible outcomes are: (Y,Y), (Y,Y+), (Y,Z), (Y+,Y), (Y+,Y+), (Y+,Z), (Z,Y), (Z,Y+). Only 4 of them gives Eve chance to obtain iGadget Z. Because she considers all outcomes equiprobable, chances are 4/8.

Note that in usual model with equal chances for each person to choose any available gadget with equal probability answer will be different.

Problem J. Japanese Year Names

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

As some of you know, in contemporary Japan two calendar systems are used: Gregorian calendar which is widely used across the world and so-called "traditional Japanese calendar" which associate dates with the Emperors.

In the traditional system, a year is represented with the Emperor name given at the time a new Emperor assumes the throne. If the era name is "Meiji", the first regnal year will be "Meiji 1", the second year will be "Meiji 2", and so forth.

Your task is to convert the year in Gregorian calendar to traditional system. You are given some associations between several Gregorian years and traditional years. For the simplicity, you can assume the following:

- A new era always starts on January 1st of the corresponding Gregorian year.
- The first year of an era is year 1.
- There is no year in which more than one era switch takes place.

Please note that, however, that information you got may be incomplete. In other words, some era that existed in the history may be missing in your data. So you will also have to detect the cases where you cannot determine exactly which era the given year belongs to.

Input

The input contains no more than 23456 test cases. Each test case has the following format:

The first line of the test case contains two positive integers N and Q ($1 \le N \le 1000$, $1 \le Q \le 1000$). N is the number of associations, and Q is the number of queries.

Each of the following N lines has three components: Emperor name, era-based year number e and the corresponding Gregorian year g ($1 \le e \le g \le 10^9$). Each of emperor names consist of at most 16 English alphabet characters. Then the last Q lines of the input specifies queries ($1 \le Q_i \le 10^9$), each of which is a Gregorian year to compute era-based representation.

The end of input is indicated by a line containing two zeros. This line is not part of any dataset and hence should not be processed.

You can assume that all the Gregorian year in the input is positive integers, and that there is no two entries that share the same Emperor name.

Output

For each query, output in a line the Emperor name and the era-based year number corresponding to the Gregorian year given, separated with a single whitespace. In case you cannot determine the era, output "`Unknown`" without quotes.

Example

standard input	standard output
4 3 iwata 10 1877 hosaka 6 1917 soejima 62 1987 takaya 22 2010 1868 1917 1988 1 1 emperobot 123 2168 2010 0 0	iwata 1 hosaka 6 Unknown Unknown

Problem K. Key

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

Consider the integer X and its decimal representation $d_1 \dots d_n$. Lets define a *split* as pair of numbers $d_1 \dots d_k$ and $d_{k+1} \dots d_n$ for any k between 1 and n inclusively (last split consists of X itself). For example, number 1234 can be split in next ways: 1 and 234, 12 and 34, 123 and 4 and 1234.

Lets define a *key* of the number as the sum of all $2n - 1$ integers, generated by splits. For example, key of number 1234 is 1+234+12+34+123+4+1234=1642.

Given the key, print any integer which generate it or -1 if no such integer exists.

Input

The input contains one integer K ($1 \le K \le 10^{15}$) — given key.

Output

Print any integer for which K is key or -1 if no such integer exists.

Example

standard input	standard output
1642	1234

Problem L. List of Strings

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

Given list of N strings. Your job is to answer Q questions of next form: how many different prefix strings exist for strings between L-th and R-th, inclusuvely.

Input

The input consists of multiple test cases.

For each test case, the first line contains one integer N ($1 \le N \le 10^5$). Then N lines follow, containing N strings composed of lowercase English letters; the total length of those N strings is between 1 and 10^5. The next line contains one integer Q ($1 \le Q \le 10^5$). For each query, you get two integers L', R' ($0 \le L', R' < N$). The query interval $[L, R]$ is defined as

$$[min((Z + L) \bmod N, (Z + R) \bmod N) + 1, max((Z + L) \bmod N, (Z + R) \bmod N) + 1],$$

where Z is equal to 0 for the first query or to answer to previous query for any other query.

Total size of the input does not exceed 10 mebibytes.

Output

For each query, output the answer on it.

Example

standard input	standard output
3 abc aba baa 3 0 2 0 1 1 1	7 6 3

Problem M. Missile

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

A missile has N jet engines. When the i-th engine is ignited, the missile's velocity changes to (vx_i, vy_i) immediately and this engine become useless in future.

Your task is to determine whether the missile can be on distance 10^{-9} or less to the given target point (X, Y). The missile can be considered as a mass point in a two-dimensional plane with the y-axis pointing up, affected by the gravity of 9.8 downward (i.e. the negative y-direction) and initially set at the origin $(0, 0)$.

The engines can be ignited at any time in any order. Still, note that at least one of them needs to be ignited at the beginning of the process to get the missile launched from the origin.

Input

The input consists of no more than 80 test cases.

First line of the each test case contains one integer N — number of engines ($1 \le N \le 1000$). Then each of the next N lines contains two integers vx_i and vy_i — speed after engine i is ignited ($0 < vx_i \le 1000$, $-1000 \le vy_i \le 1000$). Last line contains two intgegers — coordinates X and Y of the target ($0 < X \le 1000$, $-1000 \le Y \le 1000$).

The end of input is indicated by a line with a single zero.

Output

For each dataset, output whether the missile can reach the target point or any point in the sphere with center in this point and radius 10^{-9} in a line: "`Yes`" if it can, "`No`" otherwise.

Example

standard input	standard output
1 1 1 10 -480 2 6 7 5 5 4 1 3 10 5 4 4 8 4 2 0 0	Yes Yes No