Problem A. The Most Beautiful

Input file:	`beautiful.in`
Output file:	`beautiful.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

"You are fair, I can't deny,
But the Princess is the fairest
And her beauty is the rarest!"

A. S. Pushkin, "Tale of the Dead Princess and the Seven Knights"

It's time for the annual beauty contest! Imagine lines of awesome girls, showing their best looks... Unfortunately, the page margins are too narrow to fit their photos.

Being a judge of the contest is as hard as hell. The debates might take several days and a number of torn jacket buttons. To make things simpler, the Head of judges invented the following procedure.

First, every judge writes out a list of his preferences, i. e. some ordering of contestants. In this ordering, the first girl is the one the judge likes the least, the second is the least liked by him in absence of the first one, etc. After that, judges consequently place contestants: the first judge chooses the girl to be the last, the second one chooses the girl to be the pre-last and so on. After the n-th judge chooses someone, the first judge takes turn. The process continues until all girls have been assigned places. Each judge chooses the first (i. e., least liked) girl in his list that has not yet been chosen.

Vasya is the judge number k. He already knows the lists for every judge (telepathy, you know). And he does care of the only one contestant: Katya. He wants to arrange girls in his list in a way that Katya's final place would be as high as possible. Help him with this complicated task.

Input

Input consists of one or more test cases.

Each case starts with a single line containing integers n ($1 \le n \le 1\,000$), that denotes the number of judges, and k ($1 \le k \le 1\,000$), denoting the number of contestants. The second line contains two integers: Vasya's index among judges v ($1 \le v \le n$) and Katya's index among contestants x ($1 \le x \le k$). The following n lines contain judges' lists: k integers each. Each list starts with the girl that should be the last according to that judge. Vasya's list contains zeroes instead.

Input will be terminated with a test case with $n = k = 0$ which should not be processed.

The sums of n and k in all cases will not exceed 1 000 each.

Output

For each test case write the only integer: the highest rank Katya can achieve.

Adhere to the sample output format below as close as possible.

Example

beautiful.in	beautiful.out
2 3 1 2 0 0 0 2 3 1 0 0	Case #1: Katya's place can be 2.

Problem B. Chess (Division 1 Only!)

Input file:	checkmate.in
Output file:	checkmate.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Given is a toroidal three-dimensional chessboard $n \times n \times n$ ($2 \le n \le 4$).

The cells have coordinates (x, y, z) ($0 \le x, y, z < n$). Cell (x, y, z) is adjacent to cells $((x \pm 1) \bmod n,\ y,\ z)$, $(x,\ (y \pm 1) \bmod n,\ z)$ and $(x,\ y,\ (z \pm 1) \bmod n)$. Two distinct cells (x_1, y_1, z_1) and (x_2, y_2, z_2) are said to be *corner-neighbours* iff the minimum and maximum of "mod-distances" along x, y and z axes are both 1. Here, the "mod-distance" between two integers a and b is the value $\min(|a - b|,\ n - |a - b|)$.

There are k ($0 \le k \le 4$) aggressive black kings living on that chessboard, and they want to take the non-aggressive white king. Black and white move in turn. During a single move, player should move a single king of his colour to some corner-neighbouring cell. Exactly one king moves during a single move. Kings can take each other. This is done by moving to the cell where the other king is placed. One is only allowed to take king of the opposite colour. The king which is taken does not exist: it can't move or take other kings, and it doesn't occupy any cell. No two pieces can occupy the same cell.

You are given the initial coordinates of the pieces. The white king is the first to move. The question is: can the black kings manage to take him? If the answer is positive, you also have to find out the minimal number of moves to achieve that goal. The white king will resist this, so he'll try to get taken as late as possible.

Input

Input consists of one or more test cases.

Each test case starts with two integers n ($2 \le n \le 4$, the size of the chessboard) and k ($0 \le k \le 4$, the number of aggressive black kings). After that, $(k + 1) \cdot 3$ integers 0 through $n - 1$ follow: the coordinates of $k + 1$ kings (first 1 white king, then k black ones). Initially, all pieces occupy distinct cells.

Input will be terminated with a test case with $n = k = 0$ which should not be processed. There are no more than 10^4 tests.

Each integer in the input is followed by a nonempty sequence of spaces and/or newline characters.

Output

For each test case, write "YES" if the black kings manage to take the white king, and "NO" otherwise. In the first case, you should also write the minimal number of moves required if both players move optimally. You only have to count the black player's moves.

Adhere to the sample output format below as close as possible.

Example

checkmate.in	checkmate.out
2 4 0 0 0 0 0 1 0 1 0 1 0 0 1 1 1 4 4 2 2 2 0 0 0 0 0 1 0 1 0 0 1 1 0 0	Case #1: YES 1 Case #2: YES 3

Problem C. Codeforces

Input file:	`codeforces.in`
Output file:	`codeforces.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Vasya likes to compete in Codeforces competitions. In each competition, participants are given a set of at most five problems. For each correct solution, contestants get some score depending on the time spent to solve the problem. Also there are other ways to earn or lose score. Needless to say that sometimes these ways become critical to the competition outcome.

For example, when all the problems are very tricky, the winner can have no problems solved. And when they are easy, even the last one on the scoreboard may have all the problems solved. Vasya thinks such problemsets are bad.

To formalize his thoughts, he invented a formula to calculate the badness of a contest. He assumes all contestants get distinct score, so there are no coders tied for the same place.

Let n be the number of competitors. Then Vasya defines the badness as

$$B = \frac{2\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} f(i,j)^2}{n(n-1)}.$$

Here, $f(i,j)$ is equal to zero if the i-th ranked coder solved more problems than the j-th ranked coder. Otherwise, $f(i,j)$ is the difference between the number of problems solved by i-th ranked coder and the number of problems solved by j-th ranked coder.

Input

Input consists of one or more test cases.

Each case starts with a line containing a single integer n ($2 \le n \le 100\,000$). Second line contains n integers — the number of problems solved by coders (from the first ranked coder to the last one; remember that there are no more than five problems in a competition). Input will be terminated with a test case with $n = 0$ which should not be processed. Sum of all n in the input doesn't exceed 100 000.

Output

For each test case write a single line with the answer. The answer can be integer or rational. In case of rational answer, numerator and denominator should be coprime, and denominator should be positive. Adhere to the sample output format below as close as possible.

Example

codeforces.in
3 3 2 1 2 0 5 4 0 1 0 0 0
codeforces.out
Case #1: The contest badness is 0. Case #2: The contest badness is 25. Case #3: The contest badness is 1/6.

Problem D. Concatenation (Division 1 Only!)

Input file:	concat.in
Output file:	concat.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

You are given the string S which consists of lowercase Latin letters. Consider string $T(S)$ which is the concatenation of all substrings of S in lexicographical order.

For example, if $S =$ aba, the substrings are {a, b, a, ab, ba, aba}, the substrings in sorted order are {a, a, ab, aba, b, ba}, and thus $T(S) =$ aaababababba.

You have to find i-th character of the string $T(S)$.

Input

Input consists of one or more test cases.

Each case starts with a single line containing a positive integer m which denotes the number of queries. The next line contains string S ($1 \le |S| \le 5\,000$). The next line contains m integers a_i ($1 \le a_i \le |T(S)|$) denoting the queries themselves.

Input will be terminated with a test case with $m = 0$ which should not be processed.

The sum of m in all cases will not exceed 5 000.

The sum of lengths of all strings S will not exceed 5 000.

Output

For each test case, write a single line with m characters: answers to the queries. Adhere to the sample output format below as close as possible.

Example

concat.in	concat.out
10 aba 1 2 3 4 5 6 7 8 9 10 1 x 1 0	Case #1: aaababababba Case #2: x

Problem E. Dictionary

Input file:	dictionary.in
Output file:	dictionary.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

. . . Table of contents for a dictionary. . .

The List of Most Useless Books

You've found a dictionary. There are several words listed lexicographically with a hieroglyphic translation. You've already scanned the dicionary. Unfortunately, the recognition software ignored all the hieroglyphs. Moreover, it ignored all the whitespace, leaving you with a long line of Latin letters.

As there's no linguistic interest left in the line, you can solve the following combinatory problem: count the number of ways to split the line into one or more distinct words such that the words will be listed lexicographically.

Input

Input consists of one or more test cases.

Each case consists of a single line of n lowercase Latin letters ($1 \le n \le 3\,000$).

Input will be terminated with a line containing a sinlge dash, which should not be processed.

The sum of n in all cases will not exceed $3\,000$.

Output

For each test case, write the number of ways to split the line. As this number might be huge, output it modulo $10^9 + 9$.

Adhere to the sample output format below as close as possible.

Example

dictionary.in	dictionary.out
a aa ab ba abacaba abracadabra -	Case #1: There are 1 ways. Case #2: There are 1 ways. Case #3: There are 2 ways. Case #4: There are 1 ways. Case #5: There are 7 ways. Case #6: There are 34 ways.

Problem F. Friends' Friends

Input file:	`ffriends.in`
Output file:	`ffriends.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Social networks have recently changed the meaning of the word "friend". Once that meant a person who is open for you, who shares your interests, ideas, whatever. Now that's just an entry in a database, several bytes in your privacy settings.

But this is not the point. You are testing a new social network engine. To reproduce a certain bug, you need to find a pair of persons who don't have common friends.

Input

Input consists of one or more test cases.

Each case starts with a single line containing integer n ($1 \le n \le 1\,000$) that denotes the number of members of the social network.

The following n lines contain encoded graph of "friendship" relationships. Line number i describes i bits: j-th one is set to 1 iff people i and j are "friends".

The graph is encoded using a kind of *Base64* scheme.See the following algorithm for details.

Bits are grouped per six. Each group of six bits is considered as a binary number 0 through 63. Numbers 0 through 25 are encoded with letters "`A`" through "`Z`", numbers 26 through 51 are encoded with letters "`a`" through "`z`", numbers 52 through 61 are encoded with digits "`0`" through "`9`", numbers 62 and 63 are encoded with characters "`+`" and "`/`", respectively. Each line containing number of bits that is not a multiple of six is padded with trailing zeroes. Each line is encoded independently. The first character on a line corresponds to the first six bits, the second one to the second group of six bits, and so on.

Input will be terminated with a test case where $n = 0$ which should not be processed.

The sum of n in all cases will not exceed 1 000.

Output

For each test case, write a pair of indices of members that have no common "friends", or an error message if there is no such pair. If there are multiple solutions, choose the one with the smallest possible first number. If there's still a tie, choose the smallest possible second number.

Adhere to the sample output format below as close as possible.

Example

ffriends.in
3 A g Q 3 A g w 0
ffriends.out
Case #1: Members 1 and 2 have no common friends. Case #2: Social graph is too dense.

Problem G. Game Initialization (Division 1 Only!)

Input file:	game.in
Output file:	game.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

> "Three players take four tokens, and the fifth player keeps throwing out. Once the fox is eaten, it takes four moves back."
>
> "Foundling" movie

Well, it's time to set up the game field. There are n positions. Some of them are connected. Connections are made in a way that there's a single path from every position to every other one. There are also several tokens. Tokens may be placed onto the positions. There is only one restriction: two tokens can't be placed closer than c moves from each other.

To choose a fairly random placement, first find the number of ways to place the tokens. You may place any number of tokens, or even not place them at all.

And what about the goal of the game? I don't know. Maybe it's time for you to invent one.

Input

Input consists of one or more test cases.

Each test case starts with a single line containing integers n ($1 \le n \le 10\,000$) that denotes the number of positions, and c ($1 \le c \le \min(n, 500)$) that denotes the minimal distance between any pair of tokens.

The following $n - 1$ lines contain pairs of integers a_i, b_i denoting pairs of connected positions.

Input will be terminated with a test case with $n = c = 0$ which should not be processed.

The sum of n in all cases will not exceed $10\,000$.

Output

For each test case, write the number of ways to place the tokens modulo 10^6.

Adhere to the sample output format below as close as possible.

Example

game.in	game.out
5 2 1 2 1 3 1 4 4 5 2 2 1 2 2 1 1 2 0 0	Case #1: 14 Case #2: 3 Case #3: 4

Problem H. Lists Intersection

Input file:	`lists.in`
Output file:	`lists.out`
Time limit:	2 seconds *(3 seconds for Java)*
Memory limit:	256 Mebibytes

RIGS has another task for Vasya. Given are n lists of strictly increasing integers. His task is to write a program that will find the size of intersection of these lists, that is, the number of integers that are present in each of the lists.

Vasya knows that his program will be tested on generated lists. Each list contains exactly n integers (y_0, y_1, ..., y_{n-1}) and depends on five given parameters (x_0, y_0, a, b, m). The elements y_1, y_2, ... along with auxiliary values x_1, x_2, ... are generated as follows:

$$\begin{aligned} x_i &= (a \cdot x_{i-1} + b) \bmod 4\,294\,967\,296, \\ y_i &= y_{i-1} + 1 + (x_i \gg m). \end{aligned}$$

Here, $u \gg v$ means shifting the binary representation of u by v bits to the right. That is effectively the same as performing an integer division of u by 2^v. If m is greater than 31, the value of $x_i \gg m$ is equal to zero.

Input

Input consists of one or more test cases. Each case starts with a single line containing one integer n ($1 \le 8\,000$). Each of the next n lines contains five integers x_0, y_0, a, b, m ($0 \le x_0, y_0, a, b, m < 4\,294\,967\,296$) which are the parameters of the list. Input will be terminated with a test case with $n = 0$ which should not be processed.

The sum of n in all cases will not exceed 8 000.

Output

For each test case, write a single line with the answer to the problem. Adhere to the sample output format below as close as possible.

Example

lists.in
3 0 0 1 1 0 1 0 30 239 1000000 0 0 0 2 1 2 0 0 2 5 1 1 1 5 2 1 0

lists.out
Case #1: Intersection contains 2 integer(s). Case #2: Intersection contains 0 integer(s).

Problem I. Rectangles (Division 1 Only!)

Input file:	`rectangles.in`
Output file:	`rectangles.out`
Time limit:	6 seconds
Memory limit:	256 Mebibytes

You are given a square matrix of integer numbers. Every cell of the matrix also has width and length, both equal to one.

You have to find such submatrix (subrectangle) that $\frac{S}{P}$ is maximal possible, where S is sum of all numbers in the submatrix and P is the length of the perimeter of the subrectangle corresponding to the submatrix.

Input

Input consists of one or more test cases.

Each case starts with a single line containing a single integer n ($1 \le n \le 300$) denoting the size of the matrix. The following n lines contain n integers each: the matrix itself. All elements of the matrix do not exceed 10^9 by absolute value.

Input will be terminated with a test case with $n = 0$ which should not be processed.

The sum of n in all cases will not exceed 300.

Output

For each test case, write the value of the function $\frac{S}{P}$ as precisely as possible. The value you output should differ from the exact answer by no more than 10^{-5}. After that, write x_1, y_1 and x_2, y_2: coordinates of the corners of the rectangle. The first corner should be the upper left corner, and the second one should be the lower right corner.

Adhere to the sample output format below as close as possible.

Example

rectangles.in
2 100 100 1 1 0
rectangles.out
Case #1: The maximal value is 33.333333333, rectangle corners are (1, 1) and (2, 1).

Problem J. Wilson Sequence

Input file:	wilson.in
Output file:	wilson.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

For given n, Vasya should evaluate n-th element of Wilson sequence:

$$w_n = (n-1)! \bmod n.$$

Input

Input consists of one or more test cases. There are at most 10 000 cases in the input.

Each case consist of single line with an integer n ($1 \le n \le 1\,000\,000$). Input will be terminated with a test case where $n = 0$ which should not be processed.

Output

For each test case, write a single line with the answer. Adhere to the sample output format below as close as possible.

Example

wilson.in
1 2 3 5 6 21 0

wilson.out
Case #1: 1st Wilson number is equal to 0. Case #2: 2nd Wilson number is equal to 1. Case #3: 3rd Wilson number is equal to 2. Case #4: 5th Wilson number is equal to 4. Case #5: 6th Wilson number is equal to 0. Case #6: 21st Wilson number is equal to 0.

Problem K. Division (Division 2 Only!)

Input file:	`division.in`
Output file:	`division.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

For given base b, numerator x, and denominator y (last two are integers to the base b) calculate the length of the period in the base b expansion of the fraction $q = x/y$.

Digits of the base b expansion of the the rational number q are the coefficients of the powers of the base:

$$q = q_n q_{n-1} \dots q_0 . q_{-1} q_{-2} \dots = \sum_{i=-\infty}^{n} q_i b^i$$

For example, $1_{10}/4_{10} = 0.25_{10}$, $1_4/10_4 = 0.1_4$, $3_{10}/11_{10} = 0.272727\ldots_{10} = 0.(27)_{10}$.

In the first two examples, there is a finite b-based expansion of the fraction: all digits that follow are zero. In such a case, we say that the length of the period is zero (in case of ambiguity like $0.4(9)_{10}$ remember that we need minimal possible length of period). In the third example the period length is equal to 2.

Input

The first line of input file contains the number of fractions T ($1 \le T \le 250$). For each fraction you are first given the base b as a decimal integer ($2 \le b \le 36$), and then the numerator x and the denominator y ($0 < y \le 10^5_{10}$, $0 \le x \le y$). Digits greater than 9 are represented by letters from '`a`' to '`z`', where case does not matter.

Output

For each fraction print a single line containing the length of the period in the base b expansion of x/y. Print the length as a decimal number.

Example

division.in	division.out
4 10 1 4 4 1 10 10 4 11 36 Ic Pc	0 0 2 9

Problem L. Insects (Division 2 Only!)

Input file:	`insects.in`
Output file:	`insects.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

You have a long wooden stick and a group of insects is walking on top of it. Their behavior is simple: each insect walks forward with a constant speed $1 cm/sec$. Whenever it meets another insect, both insects immediately turn around and walk the opposite direction. If an insect comes to the end of the stick wood, it falls down and does not interact with other insects anymore.

Your task is to simulate the movement of insects. For simplicity, suppose that the insects have zero size.

Input

The input file starts with a line containing two integers L and A, separated by a space. L is the length of the stick in centimeters ($1 \le L < 10^5$), A is the number of insects at the beginning of the simulation ($1 \le A \le L + 1$).

Each of next A lines contain a positive integer X_i , one space, and an uppercase letter. The number ($0 \le X_i \le L$) specifies the position of the i-th insect and the letter its initial direction: either "`L`" for left (towards the zero) or "`R`" for right. No two insects will start at the same position.

Output

Print the exact time when the last insect (or two) will reach the end of the stick. If possible, print the time as an integer, otherwise use exactly one digit after the decimal point.

Examples

insects.in	insects.out
98765 1 0 R	98765
12 2 0 L 12 R	0
19 6 8 L 7 L 12 L 18 R 3 R 9 L	16

Problem M. Sentences (Division 2 Only!)

Input file:	sentences.in
Output file:	sentences.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Psychologists have claimed that understanding works in general when using the following rule: The first and last letters of word remain unmodified and all the characters in the middle can be reordered freely. But in some cases some misunderstanding may be caused, for example, "colud" can be read as "cloud" or "could".

Given a sentence, modified in such a way, and a dictionary of words, how many different sentences can you find that could potentially be mapped to the given sentence?

Input

First line of the input file contains the number of words in the dictionary n ($0 \le n \le 10^4$). Following n lines contain those words (only upper- and lowercase English letters, no longer than 100 characters each). Next line contains number of sentences m ($0 \le m \le 10^4$). Next m lines contain those sentences. The sentences consist of upper- and lowercase English letters and spaces abd have a maximal length of 10^4 characters.

Output

For each sentence in the input file print in the separate line the number of sentences that can be mapped to a given sentence.

Example

sentences.in	sentences.out
5 xyxyx xxyyx xyzxx Xxzyx xx 4 xyxyx xyzzx xxyzx xyxzx xyyxx xyxyx	2 0 1 4

Problem N. Transportation (Division 2 Only!)

Input file:	`transport.in`
Output file:	`transport.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

You are given the plan of the city, described by the streets (with weight limits) between the crossings. Crossings are are numbered from 1 (railway station) to n (port). You need to find the maximal weight that can be transported from railway station to sea port. It is guaranteed, that there is at least one path. All streets can be travelled in both directions.

Input

On the first line of input file are given two integers — n ($1 \le n \le 1000$) — number of street crossings and m — number of streets. The following m lines contain triples of integers: 2 distinct numbers between 1 and n, specifying start and end crossing of the street and the maximum allowed weight, which is positive and not larger than 10^6. There will be at most one street between each pair of crossings.

Output

Print maximum allowed weight that can be transported from railway station to sea port.

Example

transport.in	transport.out
3 3 2 1 2 3 1 4 2 3 6	4

Problem O. Travel (Division 2 Only!)

Input file:	travel.in
Output file:	travel.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

While traveling, Vasya never wrote down his routes completely, but he keeps all tickets used by him. On each ticket are written two consecutive steps of the route.

At the end of his travel Vasya would like to have his route written down as one long sequence of all the steps in the correct order. Please help him in reconstructing the route.

Input

First line of input file contains one integer T ($3 \le T \le 340$) — number of tickets. Each of the next $T - 1$ lines describe one ticket — names of station of departure and station of arrival respectively, separated by a single space. The name of each station is always a single string of upper- and lowercase English letters. Maximal length of name is 30 characters.

Output

Output T lines containing the steps of the Vasya's route in correct order.

Example

travel.in	travel.out
4 Petushki NizhnyNovgorod Moscow Petushki Ikstlan Moscow	Ikstlan Moscow Petushki NizhnyNovgorod