Sky Code

Stancu likes space travels but he is a poor software developer and will never be able to buy his own spacecraft. That is why he is preparing to steal the spacecraft of Petru. There is only one problem - Petru has locked the spacecraft with a sophisticated cryptosystem based on the ID numbers of the stars from the Milky Way Galaxy. For breaking the system Stancu has to check each subset of four stars such that the only common divisor of their numbers is 1. Nasty, isn't it? Fortunately, Stancu has succeeded to limit the number of the interesting stars to N but, any way, the possible subsets of four stars can be too many. Help him to find their number and to decide if there is a chance to break the system.

Sample Input

In the input file several test cases are given. For each test case on the first line the number N of interesting stars is given ($1 \leq N \leq 10000$) . The second line of the test case contains the list of ID numbers of the interesting stars, separated by spaces. Each ID is a positive integer which is no greater than 10000. The input data terminate with the end of file.

Sample Output

For each test case the program should print one line with the number of subsets with the asked property.

Sample Input

```
4
2 3 4 5
4
2 4 6 8
7
2 3 4 5 7 6 8
```

Sample Output

```
1
0
34
```

Internet Service Providers

A group of N Internet Service Provider companies (ISPs) use a private communication channel that has a maximum capacity of C traffic units per second. Each company transfers T traffic units per second through the channel and gets a profit that is directly proportional to the factor T (C - T N) . The problem is to compute T_optim , the smallest value of T that maximizes the total profit the N ISPs can get from using the channel.

Notice that N , C , T , and T_optim are integer numbers.

Input

Write a program that reads sets of data from an input text file. Each data set corresponds to an instance of the problem above and contains two integral numbers - N and C - with values in the range from 0 to 10^9 . The input data are separated by white spaces, are correct, and terminate with an end of file.

Output

For each data set the program computes the value of T_optim according to the problem instance that

corresponds to the data set. The result is printed on the standard output from the beginning of a line. There must be no empty lines on the output. An example of input/output is shown below.

Sample Input

```
1 0
0 1
4 3
2 8
3 27
25 1000000000
```

Sample Output

```
0
0
0
2
4
20000000
```

GCD Determinant

We say that a set $S = \{x_1, x_2,..., x_n\}$ is factor closed if for any $x_i \in S$ and any divisor d of x_i we have $d \in S$.

Let's build a GCD matrix $(S) = (s_{ij})$, where $s_{ij} = GCD(x_i, x_j)$ - the greatest common divisor of x_i and x_j . Given the factor closed set S , find the value of the determinant:

$$D_n = \begin{vmatrix} \gcd(x_1,x_1) & \gcd(x_1,x_2) & \gcd(x_1,x_3) & \dots & \gcd(x_1,x_n) \\ \gcd(x_2,x_1) & \gcd(x_2,x_2) & \gcd(x_2,x_3) & \dots & \gcd(x_2,x_n) \\ \gcd(x_3,x_1) & \gcd(x_3,x_2) & \gcd(x_3,x_3) & \dots & \gcd(x_3,x_n) \\ \dots & \dots & \dots & \dots & \dots \\ \gcd(x_n,x_1) & \gcd(x_n,x_2) & \gcd(x_n,x_3) & \dots & \gcd(x_n,x_n) \end{vmatrix}$$

Input

The input file contains several test cases. Each test case starts with an integer n ($0 < n < 1000$) , that stands for the cardinality of S . The next line contains the numbers of S : $x_1, x_2,..., x_n$. It is known that each x_i is an integer, $0 < x_i < 2 * 10^9$. The input data set is correct and ends with an end of file.

Output

For each test case find and print the value D_n mod 1000000007 .

Sample Input

```
2
1 2
3
1 3 9
4
1 2 3 6
```

Sample Output

```
1
12
4
```

Stock Exchange

The world financial crisis is quite a subject. Some people are more relaxed while others are quite anxious. John is one of them. He is very concerned about the evolution of the stock exchange. He follows stock prices every day looking for rising trends. Given a sequence of numbers p_1, p_2,..., p_n representing stock prices, a rising trend is a subsequence $p_{i1} < p_{i2} < \cdots < p_{ik}$, with $i_1 < i_2 < \cdots < i_k$. John's problem is to find very quickly the longest rising trend.

Input

The program input is from a text file. Each data set in the file stands for a particular set of stock prices. A data set starts with the length L ($L \leq 100000$) of the sequence of numbers, followed by the numbers (a number fits a long integer). The program prints the length of the longest rising trend. White spaces can occur freely in the input. The input data are correct and terminate with an end of file.

Output

For each set of data the program prints the result to the standard output from the beginning of a line.

Note for the Sample:

An input/output sample is in the table bellow. There are three data sets. In the first case, the length L of the sequence is 6. The sequence is 5, 2, 1, 4, 5, 3. The result for the data set is the length of the longest rising trend: 3.

Sample Input

```
6
5 2 1 4 5 3
3
1 1 1
4
4 3 2 1
```

Sample Output

```
3
1
1
```

Perfect Election

In a country (my memory fails to say which), the candidates { 1, 2...N } are running in the parliamentary election. An opinion poll asks the question ``For any two candidates of your own choice, which election result would make you happy?". The accepted answers are shown in the table below, where the candidates i and j are not necessarily different, i.e. it may happen that $i = j$. There are M poll answers, some of which may be similar or identical. The problem is to decide whether there can be an election outcome (It may happen that all candidates fail to be elected, or all are elected, or only a part of them are elected. All these are acceptable election outcomes.) that conforms to all M answers. We say that such an election outcome is perfect. The result of the problem is `1' if a perfect election outcome does exist and 0 otherwise.

Input

Write a program that reads sets of data from an input text file. Each data set corresponds to an instance of the problem and starts with two integral numbers: $1 \leq N \leq 1000$ and $1 \leq M \leq 1000000$. The data set continues with M pairs $\pm i$ $\pm j$ of signed numbers, $1 \leq i$,$j \leq N$. Each pair encodes a poll answer as follows:

Accepted answers to the poll question	Encoding
I would be happy if at least one from i and j is elected.	$+ i$ $+ j$
I would be happy if at least one from i and j is not elected.	$- i$ $- j$
I would be happy if i is elected or j is not elected or both events happen.	$+ i$ $- j$
I would be happy if i is not elected or j is elected or both events happen.	$- i$ $+ j$

The input data are separated by white spaces, terminate with an end of file, and are correct.

Output

For each data set the program prints the result of the encoded election problem. The result, 1 or 0, is printed on the standard output from the beginning of a line. There must be no empty lines on output. An example of input/output is shown below.

Note for the Sample:

For the first data set the result of the problem is 1; there are several perfect election outcomes, e.g. 1 is not elected, 2 is elected, 3 is not elected. The result for the second data set is justified by the perfect election outcome: 1 is not elected, 2 is not elected. The result for the third data set is 0. According to the answers -1 +2 and -1 -2 the candidate 1 must not be elected, whereas the answers +1 -2 and +1 +2 say that candidate 1 must be elected. There is no perfect election outcome. For the fourth data set notice that there are similar or identical poll answers and that some answers mention a single candidate. The result is 1.

Sample Input

```
3 3  +1 +2  -1 +2  -1 -3
2 3  -1 +2  -1 -2  +1 -2
2 4  -1 +2  -1 -2  +1 -2  +1 +2
2 8  +1 +2  +2 +1  +1 -2  +1 -2  -2 +1  -1 +1  -2 -2  +1 -1
```

Sample Output

```
1
1
0
1
```

Lucky Cities

John has recently arrived in Romania for the South Eastern European Regional competitions. John has never been to Romania before so Romanians decided to organize sightseeing tour for him. This tour will include several Romanian cities and none of them will be visited more than once. John will start in one city and will visit some other cities according to a guide tour. At the end of the tour John will return to the starting point.

There are N cities numbered from 1 to N and M two-way roads in the country. Each road connects two different cities. Consider a sightseeing tour for John c_1, c_2,..., c_n , where each c_i denotes a city in Romania. Then all c_i must be distinct, c_i and c_{i+1} must be connected by a road, where i = 1, 2,..., n - 1 , c_n and c_1 must be connected by a road as well.

Being a odd person John would like to visit an odd number of cities. The organizers have drawn the plan of all possible tours with an odd number of cities.

Residents of the cities would like John to visit them. So if there is at least one tour passing through some city than this city is called lucky. Your task is to calculate the number of lucky cities in Romania.

Input

The first line of input contains a single integer T - a number of test cases. Every test case starts with a line containing two integers separated by a single space - N and M . Each of the next M lines will contain two integers a_i and b_i separated by a single space - the labels of cities that i -th road connects.

Output

Output should contain T lines - answers for each of the test cases.

Constrains:

$1 \leq T \leq 77$,

$0 \leq N, M \leq 100000 (10^5)$,

$1 \leq a_i < b_i \leq N$.

Sample Input

```
1
7 7
1 5
3 5
3 7
1 7
6 7
4 7
4 6
```

Sample Output

```
3
```

Quick answer

Joe is fond of computer games. Now, he must solve a puzzling situation. In front of his eyes lies a huge map with fortified towns. His enemy is a very powerful and tricky character who can connect and disconnect the towns by giving some commands. Two towns are connected if they have been directly connected or interconnected through some other connected towns at some moment in time. When a town is disconnected it gets isolated and clears its own connection history, not the connection history of the other towns. Each connection is bi-directional. Initially the towns are isolated. Joe is asked to answer quickly if two towns are connected, according to the history of the character's commands.

Write a program which based on information input from a text file counts the number of yes answers and the number of no answers to questions of the kind: is $town_i$ connected with $town_j$?

Input

The program reads data from a text file. Each data set in the file stands for a particular map and the associated character's commands, as follows:

1. The number of towns on the map N ($N \leq 10000$) ;
2. A list of commands of the form:
   a)
   `c` $town_i$ $town_j$, where $town_i$ and $town_j$ are integers from 1 to *no_of_towns* . The command means that $town_i$ and $town_j$ get connected.
   b)
   `d` $town_i$, where $town_i$ is an integer from 1 to *no_of_towns* . The command means that $town_i$ gets disconnected.
   c)
   `q` $town_i$ $town_j$, where $town_i$ and $town_j$ are integers from 1 to *no_of_towns* . The command stands for the question: is $town_i$ connected with $town_j$?
   d)
   `e`, that ends the list of commands

Each command is on a separate line. Commands (a), (b), (c) can appear in any order. The towns' connectivity is updated after each command of type (a) or (b). Each command of type (c) is processed according to the current configuration.

For example, the input file illustrated in the figure bellow corresponds to only one data set which stands for a map with `4` fortified towns. The character gives 9 commands. There are N_1 yes answered questions and N_2 no answered questions.

Output

The program prints these two numbers to the standard output on the same line, in the order: N_1 , N_2 as shown in the sample below.

Sample Input

```
4
c 1 2
c 3 4
q 1 3
c 2 3
q 1 4
d 2
q 4 1
q 2 4
e
```

Sample Output

```
2 , 2
```

Build Your Home

Mr. Tenant is going to buy a new house. In fact, he is going to buy a piece of land and build his new house on it. In order to decide which piece of land to buy, Mr. Tenant needs a program which will give a score to each piece. Each candidate piece of land is a polygonal shape (not necessarily convex), and Mr. Tenant wonders what the best score is. Among possible scores, he considered the number of vertices, sum of angles, minimum number of required guards, and so forth. Finally, Mr. Tenant decided that the best score for a piece of land would simply be its area. Your task is to write the desired scoring program.

Input

The input file consists of multiple pieces of land. Each piece is a simple polygon (that is, a polygon which does not intersect itself). A polygon description starts with a positive integer number k , followed by k vertices, where each vertex consists of two coordinates (floating-point numbers): x and y . Naturally, the last vertex is connected by an edge to the first vertex. Note that each polygon can be ordered either clockwise or counterclockwise. The input ends with a ``0" (the number zero).

Output

For each piece of land, the output should consist of exactly one line containing the score of that piece, rounded to the nearest integer number. (Halves should be rounded up, but Mr. Tenant never faced such cases.) Hint: The scoring program has to handle well degenerate cases, such as, polygons with only one or two vertices.

Sample Input

```
1 123.45 67.890
3 0.001 0   1.999 0   0 2
5 10 10   10 12   11 11   12 12   12.0 10.0
0
```

Sample Output

```
0
2
3
```

Lucky numbers

John has recently arrived in Bucharest for the South Eastern European Regional Contest. John is famous for his theory of lucky numbers. That's why all the contestants and spectators are very happy.

According to that theory 4 and 7 are lucky digits, and all the other digits are not lucky. A lucky number is a number that contains only lucky digits in decimal notation. A very lucky number is a number that can be expressed as a product of several lucky numbers. A lucky number by itself is considered to be very lucky. For example, numbers 47, 49, 112 are very lucky.

Your task is to calculate the number of very lucky numbers that are not less than A and not greater than B . Of course, numbers A and B are given by John.

Input

The first line of the input contains a single integer T - a number of test cases. Each of the next T lines contains two integers separated by a single space - A and B .

Output

Output must contain T lines - answers for the test cases.

Constrains:

$1 \leq T \leq 7777$,

$1 \leq A \leq B \leq 1000000000000 (10^{12})$.

Hint: Very lucky numbers for the last case of the sample input are: 4, 7, 16, 28, 44, 47, 49, 64, 74 and 77.

Sample Input

```
4
1 2
88 99
112 112
1 100
```

Sample Output

```
0
0
1
10
```