A. Dorm Water Supply

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

The German University in Cairo (GUC) dorm houses are numbered from 1 to n. Underground water pipes connect these houses together. Each pipe has certain direction (water can flow only in this direction and not vice versa), and diameter (which characterizes the maximal amount of water it can handle).

For each house, there is at most one pipe going into it and at most one pipe going out of it. With the new semester starting, GUC student and dorm resident, Lulu, wants to install tanks and taps at the dorms. For every house with an outgoing water pipe and without an incoming water pipe, Lulu should install a water tank at that house. For every house with an incoming water pipe and without an outgoing water pipe, Lulu should install a water tap at that house. Each tank house will convey water to all houses that have a sequence of pipes from the tank to it. Accordingly, each tap house will receive water originating from some tank house.

In order to avoid pipes from bursting one week later (like what happened last semester), Lulu also has to consider the diameter of the pipes. The amount of water each tank conveys should not exceed the diameter of the pipes connecting a tank to its corresponding tap. Lulu wants to find the maximal amount of water that can be safely conveyed from each tank to its corresponding tap.

Input

The first line contains two space-separated integers n and p $(1 \le n \le 1000, 0 \le p \le n)$ — the number of houses and the number of pipes correspondingly.

Then p lines follow — the description of p pipes. The i-th line contains three integers a_i b_i d_i, indicating a pipe of diameter d_i going from house a_i to house b_i ($1 \le a_i, b_i \le n, a_i \ne b_i, 1 \le d_i \le 10^6$).

It is guaranteed that for each house there is at most one pipe going into it and at most one pipe going out of it.

Output

Print integer t in the first line — the number of tank-tap pairs of houses.

For the next t lines, print 3 integers per line, separated by spaces: $tank_i$, tap_i, and $diameter_i$, where $tank_i \ne tap_i$ ($1 \le i \le t$). Here $tank_i$ and tap_i are indexes of tank and tap houses respectively, and $diameter_i$ is the maximum amount of water that can be conveyed. All the t lines should be ordered (increasingly) by $tank_i$.

Examples

input

```
3 2
1 2 10
2 3 20
```

output

```
1
1 3 10
```

input

```
3 3
1 2 20
2 3 10
3 1 5
```

output

```
0
```

input

```
4 2
1 2 60
3 4 50
```

output

```
2
1 2 60
3 4 50
```

B. Basketball Team

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

As a German University in Cairo (GUC) student and a basketball player, Herr Wafa was delighted once he heard the news. GUC is finally participating in the Annual Basketball Competition (ABC).

A team is to be formed of n players, all of which are GUC students. However, the team might have players belonging to different departments. There are m departments in GUC, numbered from 1 to m. Herr Wafa's department has number h. For each department i, Herr Wafa knows number s_i — how many students who play basketball belong to this department.

Herr Wafa was also able to guarantee a spot on the team, using his special powers. But since he hates floating-point numbers, he needs your help at finding the probability that he will have at least one teammate belonging to his department.

Note that every possible team containing Herr Wafa is equally probable. Consider all the students different from each other.

Input

The first line contains three integers n, m and h ($1 \le n \le 100, 1 \le m \le 1000, 1 \le h \le m$) — the number of players on the team, the number of departments in GUC and Herr Wafa's department, correspondingly.

The second line contains a single-space-separated list of m integers s_i ($1 \le s_i \le 100$), denoting the number of students in the i-th department. Note that s_h includes Herr Wafa.

Output

Print the probability that Herr Wafa will have at least one teammate from his department. If there is not enough basketball players in GUC to participate in ABC, print -1. The answer will be accepted if it has absolute or relative error not exceeding 10^{-6}.

Examples

input

```
3 2 1
2 1
```

output

```
1
```

input

```
3 2 1
1 1
```

output

```
-1
```

input

```
3 2 1
2 2
```

output

```
0.666667
```

Note

In the first example all 3 players (2 from department 1 and 1 from department 2) must be chosen for the team. Both players from Wafa's departments will be chosen, so he's guaranteed to have a teammate from his department.

In the second example, there are not enough players.

In the third example, there are three possibilities to compose the team containing Herr Wafa. In two of them the other player from Herr Wafa's department is part of the team.

C. Arrangement

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

In the year 2500 the annual graduation ceremony in the German University in Cairo (GUC) has run smoothly for almost 500 years so far.

The most important part of the ceremony is related to the arrangement of the professors in the ceremonial hall.

Traditionally GUC has n professors. Each professor has his seniority level. All seniorities are different. Let's enumerate the professors from 1 to n, with 1 being the most senior professor and n being the most junior professor.

The ceremonial hall has n seats, one seat for each professor. Some places in this hall are meant for more senior professors than the others. More specifically, m pairs of seats are in "senior-junior" relation, and the tradition requires that for all m pairs of seats (a_i, b_i) the professor seated in "senior" position a_i should be more senior than the professor seated in "junior" position b_i.

GUC is very strict about its traditions, which have been carefully observed starting from year 2001. The tradition requires that:

- The seating of the professors changes every year.
- Year 2001 ceremony was using lexicographically first arrangement of professors in the ceremonial hall.
- Each consecutive year lexicographically next arrangement of the professors is used.

The arrangement of the professors is the list of n integers, where the first integer is the seniority of the professor seated in position number one, the second integer is the seniority of the professor seated in position number two, etc.

Given n, the number of professors, y, the current year and m pairs of restrictions, output the arrangement of the professors for this year.

Input

The first line contains three integers n, y and m ($1 \le n \le 16, 2001 \le y \le 10^{18}, 0 \le m \le 100$) — the number of professors, the year for which the arrangement should be computed, and the number of pairs of seats for which the seniority relation should be kept, respectively.

The next m lines contain one pair of integers each, "a_i b_i", indicating that professor on the a_i-th seat is more senior than professor on the b_i-th seat ($1 \le a_i, b_i \le n, a_i \ne b_i$). Some pair may be listed more than once.

Please, do not use the `%lld` specificator to read or write 64-bit integers in C++. It is preferred to use the `cin` stream (you may also use the `%I64d` specificator).

Output

Print the order in which the professors should be seated in the requested year.

If by this year the GUC would have ran out of arrangements, or the given "senior-junior" relation are contradictory, print "`The times have changed`" (without quotes).

Examples

input

```
3 2001 2
1 2
2 3
```

output

```
1 2 3
```

input

```
7 2020 6
1 2
1 3
2 4
2 5
3 6
3 7
```

output

```
1 2 3 7 4 6 5
```

input

```
10 3630801 0
```

output

```
The times have changed
```

input

```
3 2001 3
1 2
2 3
3 1
```

output

```
The times have changed
```

Note

In the first example the lexicographically first order of seating is 1 2 3.

In the third example the GUC will run out of arrangements after the year 3630800.

In the fourth example there are no valid arrangements for the seating.

The lexicographical comparison of arrangements is performed by the < operator in modern programming languages. The arrangement a is lexicographically less that the arrangement b, if there exists such i ($1 \le i \le n$), that $a_i < b_i$, and for any j ($1 \le j < i$) $a_j = b_j$.

D. Crime Management

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Zeyad wants to commit n crimes in Egypt and not be punished at the end. There are several types of crimes. For example, bribery is a crime but is not considered such when repeated twice. Therefore, bribery is not considered a crime when repeated an even number of times. Speeding is a crime, but is not considered such when repeated a number of times which is a multiple of five.

More specifically, c conditions on crime repetitions are known. Each condition describes the crime type t_i and its multiplicity m_i. If the number of times Zeyad committed the crime t_i is a multiple of m_i, Zeyad will not be punished for crime t_i. Some crimes may be listed more than once. In this case fulfilling at least one condition for this crime is enough to not be punished for it. Of course, if for certain crime the number of times Zeyad committed it is zero, he is innocent with respect to this crime.

Now Zeyad is interested in a number of ways he can commit exactly n crimes without any punishment.

The order of commiting the crimes matters. More formally, two ways, sequences $w1$ and $w2$, of committing n crimes are equal if $w1_i = w2_i$, for all $1 \le i \le n$.

Input

The first line contains two integers n and c ($0 \le n \le 10^{18}, 0 \le c \le 1000$) — the number of crimes Zeyad would like to commit and the number of conditions he is aware of.

Then the definitions for c conditions follow. There are 26 types of crimes. Each crime definition consists of crime type — a capital Latin letter — and its multiplicity.

The multiplicity of each crime is a positive integer number and the product of all multiplicities does not exceed 123. Some conditions may be repeated in the input more than once.

Crime of multiplicity 1 is not yielding any punishment regardless of the number of times it was committed. The strictness of the law is compensated by the fact that it's non-mandatory.

Obviously, if some crime is not listed in the set of conditions, then Zeyad will not consider it, as committing it would unavoidably lead to the punishment.

Please, do not use the `%lld` specificator to read or write 64-bit integers in C++. It is preferred to use the `cin` stream (you may also use the `%I64d` specificator).

Output

Output the number of different ways Zeyad can commit exactly n crimes with no punishment modulo 12345.

Examples

input
```
5 2
A 1
B 2
```
output
```
16
```

input
```
6 3
A 1
B 2
C 3
```
output
```
113
```

input
```
8 3
A 2
A 3
B 2
```
output
```
128
```

Note

In the first test case, the 16 ways are: AAAAA, AAABB, AABAB, AABBA, ABAAB, ABABA, ABBAA, BAAAB, BAABA, BABAA, BBAAA, ABBBB, BABBB, BBABB, BBBAB, BBBBA.

E. Darts

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The night after the graduation ceremony graduate students of German University in Cairo (GUC) are playing darts. As there's no real dart board available, the photographs of members of the GUC upper management are being used.

So, n rectangular photos are placed on the wall. They can overlap arbitrary and even coincide. The photos are not necessarily placed horizontally or vertically, they could also be rotated before being pinned to the wall.

The score of one dart throw is simply the number of photos the dart went through.

Fatma has made a throw but her score was not recorded. She only remembers that she did make it into at least one photo.

Assuming that the probability distribution of the throw is equal across the whole wall, what would be the expectation of Fatma's score?

Input

The first line of input contains integer n ($1 \le n \le 500$) — the number of photos on the wall. Then follow n lines describing the photos, each containing 8 single-space-separated integers (coordinates of 4 vertices): x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4. Each photo is a rectangle with a nonzero area. The coordinates are integers, not exceeding 10^4 by absolute value. The coordinates of the rectangle are given in either clockwise or counterclockwise order.

Output

Print the expected score of the throw. The answer will be accepted if it has absolute or relative error not exceeding 10^{-6}.

Examples

input

```
1
0 0 0 2 2 2 2 0
```

output

```
1.0000000000
```

input

```
1
-1 0 0 1 1 0 0 -1
```

output

```
1.0000000000
```

input

```
4
0 0 0 1 3 1 3 0
0 0 0 3 1 3 1 0
3 3 2 3 2 0 3 0
3 3 3 2 0 2 0 3
```

output

```
1.5000000000
```

input

```
2
-1 0 0 1 1 0 0 -1
0 0 1 1 2 0 1 -1
```

output

```
1.1428571429
```

Codeforces (c) Copyright 2010-2016 Mike Mirzayanov
The only programming contests Web 2.0 platform