Codeforces Beta Round 83 (Div. 2 Only)

A. Palindromic Times

2 seconds, 256 megabytes

Tattah is asleep if and only if Tattah is attending a lecture. This is a well-known formula among Tattah's colleagues.

On a Wednesday afternoon, Tattah was attending Professor HH's lecture. At `12:21`, right before falling asleep, he was staring at the digital watch around Saher's wrist. He noticed that the digits on the clock were the same when read from both directions i.e. a palindrome.

In his sleep, he started dreaming about such rare moments of the day when the time displayed on a digital clock is a palindrome. As soon as he woke up, he felt destined to write a program that finds the next such moment.

However, he still hasn't mastered the skill of programming while sleeping, so your task is to help him.

Input

The first and only line of the input starts with a string with the format "`HH:MM`" where "`HH`" is from "`00`" to "`23`" and "`MM`" is from "`00`" to "`59`". Both "`HH`" and "`MM`" have exactly two digits.

Output

Print the palindromic time of day that comes soonest after the time given in the input. If the input time is palindromic, output the soonest palindromic time after the input time.

input
12:21
output
13:31

input
23:59
output
00:00

B. Datatypes

2 seconds, 256 megabytes

Tattah's youngest brother, Tuftuf, is new to programming.

Since his older brother is such a good programmer, his biggest dream is to outshine him. Tuftuf is a student at the German University in Cairo (GUC) where he learns to write programs in Gava.

Today, Tuftuf was introduced to Gava's unsigned integer datatypes. Gava has n unsigned integer datatypes of sizes (in bits) $a_1, a_2, \ldots a_n$. The i-th datatype have size a_i bits, so it can represent every integer between 0 and $2^{a_i} - 1$ inclusive.

Tuftuf is thinking of learning a better programming language. If there exists an integer x, such that x fits in some type i (in a_i bits) and $x \cdot x$ does not fit in some other type j (in a_j bits) where $a_i < a_j$, then Tuftuf will stop using Gava.

Your task is to determine Tuftuf's destiny.

Input

The first line contains integer n ($2 \le n \le 10^5$) — the number of Gava's unsigned integer datatypes' sizes. The second line contains a single-space-separated list of n integers ($1 \le a_i \le 10^9$) — sizes of datatypes in bits. Some datatypes may have equal sizes.

Output

Print "`YES`" if Tuftuf will stop using Gava, and "`NO`" otherwise.

input
3 64 16 32
output
NO

input
4 4 2 1 3
output
YES

In the second example, $x = 7$ (111_2) fits in 3 bits, but $x^2 = 49$ (110001_2) does not fit in 4 bits.

C. Dorm Water Supply

1 second, 256 megabytes

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 $tank_i \neq tap_i$ $(1 \leq i \leq t)$. 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 $tank_i$.

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

D. Basketball Team

1 second, 256 megabytes

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 $(1 \leq n \leq 100, 1 \leq m \leq 1000, 1 \leq h \leq m)$ — 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 $(1 \leq s_i \leq 100)$, 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}.

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

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.

E. Arrangement

2 seconds, 256 megabytes

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).

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

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 ($1 \le j < i$) $a_j = b_j$.