Codeforces Beta Round 9 (Div. 2 Only)

A. Die Roll

1 second, 64 megabytes

Yakko, Wakko and Dot, world-famous animaniacs, decided to rest from acting in cartoons, and take a leave to travel a bit. Yakko dreamt to go to Pennsylvania, his Motherland and the Motherland of his ancestors. Wakko thought about Tasmania, its beaches, sun and sea. Dot chose Transylvania as the most mysterious and unpredictable place.

But to their great regret, the leave turned to be very short, so it will be enough to visit one of the three above named places. That's why Yakko, as the cleverest, came up with a truly genius idea: let each of the three roll an ordinary six-sided die, and the one with the highest amount of points will be the winner, and will take the other two to the place of his/her dreams.

Yakko thrown a die and got `Y` points, Wakko — `W` points. It was Dot's turn. But she didn't hurry. Dot wanted to know for sure what were her chances to visit Transylvania.

It is known that Yakko and Wakko are true gentlemen, that's why if they have the same amount of points with Dot, they will let Dot win.

Input
The only line of the input file contains two natural numbers `Y` and `W` — the results of Yakko's and Wakko's die rolls.

Output
Output the required probability in the form of irreducible fraction in format «`A/B`», where `A` — the numerator, and `B` — the denominator. If the required probability equals to zero, output «`0/1`». If the required probability equals to 1, output «`1/1`».

input
4 2
output
1/2

Dot will go to Transylvania, if she is lucky to roll 4, 5 or 6 points.

B. Running Student

1 second, 64 megabytes

And again a misfortune fell on Poor Student. He is being late for an exam.

Having rushed to a bus stop that is in point $(0, 0)$, he got on a minibus and they drove along a straight line, parallel to axis OX, in the direction of increasing x.

Poor Student knows the following:

- during one run the minibus makes n stops, the i-th stop is in point $(x_i, 0)$
- coordinates of all the stops are different
- the minibus drives at a constant speed, equal to v_b
- it can be assumed the passengers get on and off the minibus at a bus stop momentarily
- Student can get off the minibus only at a bus stop
- Student will have to get off the minibus at a terminal stop, if he does not get off earlier
- the University, where the exam will be held, is in point (x_u, y_u)
- Student can run from a bus stop to the University at a constant speed v_s as long as needed
- a distance between two points can be calculated according to the following formula: $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
- Student is already on the minibus, so, he cannot get off at the first bus stop

Poor Student wants to get to the University as soon as possible. Help him to choose the bus stop, where he should get off. If such bus stops are multiple, choose the bus stop closest to the University.

Input
The first line contains three integer numbers: $2 \le n \le 100$, $1 \le v_b, v_s \le 1000$. The second line contains n non-negative integers in ascending order: coordinates x_i of the bus stop with index i. It is guaranteed that x_1 equals to zero, and $x_n \le 10^5$. The third line contains the coordinates of the University, integers x_u and y_u, not exceeding 10^5 in absolute value.

Output
In the only line output the answer to the problem — index of the optimum bus stop.

input
4 5 2 0 2 4 6 4 1
output
3

input
2 1 1 0 100000 100000 100000
output
2

As you know, students are a special sort of people, and minibuses usually do not hurry. That's why you should not be surprised, if Student's speed is higher than the speed of the minibus.

C. Hexadecimal's Numbers

1 second, 64 megabytes

One beautiful July morning a terrible thing happened in Mainframe: a mean virus Megabyte somehow got access to the memory of his not less mean sister Hexadecimal. He loaded there a huge amount of n different natural numbers from 1 to n to obtain total control over her energy.

But his plan failed. The reason for this was very simple: Hexadecimal didn't perceive any information, apart from numbers written in binary format. This means that if a number in a decimal representation contained characters apart from 0 and 1, it was not stored in the memory. Now Megabyte wants to know, how many numbers were loaded successfully.

Input
Input data contains the only number n ($1 \le n \le 10^9$).

Output

Output the only number — answer to the problem.

input
10
output
2

For n = 10 the answer includes numbers 1 and 10.

D. How many trees?

1 second, 64 megabytes

In one very old text file there was written Great Wisdom. This Wisdom was so Great that nobody could decipher it, even Phong — the oldest among the inhabitants of Mainframe. But still he managed to get some information from there. For example, he managed to learn that User launches games for pleasure — and then terrible Game Cubes fall down on the city, bringing death to those modules, who cannot win the game...

For sure, as guard Bob appeared in Mainframe many modules stopped fearing Game Cubes. Because Bob (as he is alive yet) has never been defeated by User, and he always meddles with Game Cubes, because he is programmed to this.

However, unpleasant situations can happen, when a Game Cube falls down on Lost Angles. Because there lives a nasty virus — Hexadecimal, who is... mmm... very strange. And she likes to play very much. So, willy-nilly, Bob has to play with her first, and then with User.

This time Hexadecimal invented the following entertainment: Bob has to leap over binary search trees with n nodes. We should remind you that a binary search tree is a binary tree, each node has a distinct key, for each node the following is true: the left sub-tree of a node contains only nodes with keys less than the node's key, the right sub-tree of a node contains only nodes with keys greater than the node's key. All the keys are different positive integer numbers from 1 to n. Each node of such a tree can have up to two children, or have no children at all (in the case when a node is a leaf).

In Hexadecimal's game all the trees are different, but the height of each is not lower than h. In this problem «height» stands for the maximum amount of nodes on the way from the root to the remotest leaf, the root node and the leaf itself included. When Bob leaps over a tree, it disappears. Bob gets the access to a Cube, when there are no trees left. He knows how many trees he will have to leap over in the worst case. And you?

Input

The input data contains two space-separated positive integer numbers n and h ($n \le 35$, $h \le n$).

Output

Output one number — the answer to the problem. It is guaranteed that it does not exceed $9 \cdot 10^{18}$.

input
3 2
output
5

input
3 3
output
4

E. Interesting Graph and Apples

1 second, 64 megabytes

Hexadecimal likes drawing. She has drawn many graphs already, both directed and not. Recently she has started to work on a still-life «interesting graph and apples». An undirected graph is called interesting, if each of its vertices belongs to one cycle only — a funny ring — and does not belong to any other cycles. A funny ring is a cycle that goes through all the vertices just once. Moreover, loops are funny rings too.

She has already drawn the apples and some of the graph edges. But now it is not clear, how to connect the rest of the vertices to get an interesting graph as a result. The answer should contain the minimal amount of added edges. And furthermore, the answer should be the lexicographically smallest one. The set of edges $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$, where $x_i \le y_i$, is lexicographically smaller than the set $(u_1, v_1), (u_2, v_2), ..., (u_n, v_n)$, where $u_i \le v_i$, provided that the sequence of integers $x_1, y_1, x_2, y_2, ..., x_n, y_n$ is lexicographically smaller than the sequence $u_1, v_1, u_2, v_2, ..., u_n, v_n$. If you do not cope, Hexadecimal will eat you. ...eat you alive.

Input

The first line of the input data contains a pair of integers n and m ($1 \le n \le 50$, $0 \le m \le 2500$) — the amount of vertices and edges respectively. The following lines contain pairs of numbers x_i and y_i ($1 \le x_i, y_i \le n$) — the vertices that are already connected by edges. The initial graph may contain multiple edges and loops.

Output

In the first line output «YES» or «NO»: if it is possible or not to construct an interesting graph. If the answer is «YES», in the second line output k — the amount of edges that should be added to the initial graph. Finally, output k lines: pairs of vertices x_j and y_j, between which edges should be drawn. The result may contain multiple edges and loops. k can be equal to zero.

input
3 2 1 2 2 3
output
YES 1 1 3