Problem A. Nonoptimal Assignments

Input file:	`assign.in`
Output file:	`assign.out`
Time limit:	1 second
Memory limit:	64 megabytes

You might have heard about *assignment problem*, stated as follows. Given $n \times n$ matrix of integer numbers, one has to choose n elements of the matrix, so that for each row and each column exactly one element is selected from it, and the sum of the selected elements is minimal possible.

Little Mini has just heard about the problem and thinks that it can be solved using so called "greedy algorithm". That is, she thinks that it is possible to take the smallest element from the first row, after that take minimal element from the second row, that does not belong to the column that was already used, and so on. Each time if there are several possible elements, the one from the smallest column is used.

Her brother Maxi understands that it is not true. To prove it, he wants to construct a matrix where Mini's assignment would be non-optimal. Help him to do so.

Input

Input file contains n ($2 \le n \le 100$).

Output

Output $n \times n$ matrix of integer numbers where Mini's algorithm would not find the optimal assignment. If no such matrix exists, output "`Impossible`" instead. Matrix must contain only non-negative integer numbers not exceeding 100.

Example

assign.in	assign.out
2	0 1 1 10

Problem B. Cryptography

Input file:	`crypto.in`
Output file:	`crypto.out`
Time limit:	1 second
Memory limit:	64 megabytes

Young cryptoanalyst Georgie is planning to break the new cipher invented by his friend Andie. To do this, he must make some linear transformations over the ring $\mathbb{Z}_r = \mathbb{Z}/r\mathbb{Z}$.

Each linear transformation is defined by 2×2 matrix. Georgie has a sequence of matrices $A_1, A_2, \ldots, A_n$. As a step of his algorithm he must take some segment $A_i, A_{i+1}, \ldots, A_j$ of the sequence and multiply some vector by a product $P_{i,j} = A_i \times A_{i+1} \times \cdots \times A_j$ of the segment. He must do it for m various segments.

Help Georgie to determine the products he needs.

Input

The first line of the input file contains r ($1 \le r \le 10\,000$), n ($1 \le n \le 30\,000$) and m ($1 \le m \le 30\,000$). Next n blocks of two lines, containing two integer numbers ranging from 0 to $r-1$ each, describe matrices. Blocks are separated with blank lines. They are followed by m pairs of integer numbers ranging from 1 to n each that describe segments, products for which are to be calculated.

Output

Print m blocks containing two lines each. Each line should contain two integer numbers ranging from 0 to $r - 1$ and define the corresponding product matrix.

Separate blocks with an empty line.

Example

crypto.in	crypto.out
3 4 4 0 1 0 0 2 1 1 2 0 0 0 2 1 0 0 2 1 4 2 3 1 3 2 2	0 2 0 0 0 2 0 1 0 1 0 0 2 1 1 2

Problem C. Fibonacci Subsequence

Input file:	`fibsubseq.in`
Output file:	`fibsubseq.out`
Time limit:	1 second
Memory limit:	64 megabytes

A sequence of integer numbers $a_1, a_2, \ldots, a_n$ is called a *Fibonacci sequence* if $a_i = a_{i-2} + a_{i-1}$ for all $i = 3, 4, \ldots, n$.

Given a sequence of integer numbers $c_1, c_2, \ldots, c_m$ you have to find its longest Fibonacci subsequence.

Input

The first line of the input file contains m ($1 \le m \le 3\,000$). Next line contains m integer numbers not exceeding 10^9 by their absolute value.

Output

On the first line of the output file print the maximal length of the Fibonacci subsequence of the given sequence. On the second line print the subsequence itself.

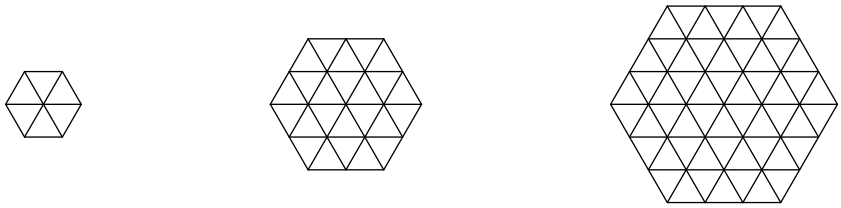
Example

fibsubseq.in	fibsubseq.out
10 1 1 3 -1 2 0 5 -1 -1 8	5 1 -1 0 -1 -1

Problem D. Hexagon and Rhombic Dominoes

Input file:	`hex.in`
Output file:	`hex.out`
Time limit:	1 second
Memory limit:	64 megabytes

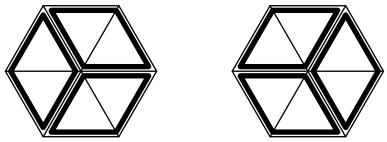
A regular hexagon with side length n is divided into $6n^2$ unit triangles.



Your task is to cover it with *rhombic dominoes* — pieces composed of two unit triangles sharing a side.

Each domino must be placed in such a way, that it covers exactly two unit triangles. No triangle must be covered with more than one domino.

Count the number of ways to do so. For example there are two ways to cover a hexagon with side length 1, they are shown on the picture.



Input

Input file contains n ($1 \le n \le 7$).

Output

Output the number of ways to cover hexagon with rombic dominoes.

Example

hex.in	hex.out
1	2
2	20

Problem E. Strange Limit

Input file:	`limit.in`
Output file:	`limit.out`
Time limit:	1 second
Memory limit:	64 megabytes

Consider sequence a_n defined with the following recurrence:

$$a_1 = p,$$
$$a_{n+1} = p^{a_n} \text{ for } n \geq 1,$$

where p is some prime number. Let

$$b_n = a_n \bmod m!,$$

where $m!$ denotes m factorial, that is $m! = 1 \cdot 2 \cdot \ldots \cdot m$.

It may seem strange, but for all p and all m the sequence b_n has limit as $n \to +\infty$. Your task is to find it. Given p and m, find

$$\lim_{n \to +\infty} b_n.$$

Input

Input file contains p and m ($2 \leq p, m \leq 12$, p is prime).

Output

Output the limit requested.

Example

limit.in	limit.out
2 2	0
2 3	4
3 8	30267

Problem F. Little Mammoth

Input file:	mammoth.in
Output file:	mammoth.out
Time limit:	1 second
Memory limit:	64 megabytes

It is well known that mammoths used to live in caves. This is a story of a little mammoth who lived in a cave with his mummy and daddy.

The mammoth was little and very cute. And he was very curious. He used to peep out of the cave and look around. And one day mummy said him:

"You are a big boy now, dear, so you may go out of the cave and take a little small walk around. But beware! There is a lot of danger outside. Horrible humans may try to catch you and make a dinner out of you! Do no walk further than r meters away from the cave entrance."

And the little mammoth made the first step out of the cave. He was a good boy, so he decided not to violate mummy's order. But suddenly he saw a nice field of grass around. Nice, green, juicy, tasty grass! How could he stand it!

But no, those dangerous humans. Little mammoth thought for a while and decided that he would only eat the grass that he can reach not breaking mummy's recommendation.

The field of grass is a rectangle. Find out how much grass can little mammoth eat.

Input

The first line of the input file contains x_c, y_c and r — coordinates of the entrance to the cave and the distance little mammoth is allowed to walk from it.

Next line contains x_1, y_1, x_2, and y_2 — coordinates of two opposite corners of the grass field. Coordinate system is set up in such a way that field's sides are parallel to coordinate axes.

All numbers in the input file are integer and do not exceed 1000 by their absolute values, $r > 0$, both field sides are non-zero.

Output

Output the area of the part of the field where little mammoth can eat grass. Your answer must be accurate up to 10^{-6}.

Example

mammoth.in	mammoth.out
0 0 5 3 3 7 7	0.547426365104

Problem G. Network Wars

Input file:	`network.in`
Output file:	`network.out`
Time limit:	1 second
Memory limit:	64 megabytes

Network of Byteland consists of n servers, connected by m optical cables. Each cable connects two servers and can transmit data in both directions. Two servers of the network are especially important — they are connected to global world network and president palace network respectively.

The server connected to the president palace network has number 1, and the server connected to the global world network has number n.

Recently the company *Max Traffic* has decided to take control over some cables so that it could see what data is transmitted by the president palace users. Of course they want to control such set of cables, that it is impossible to download any data from the global network to the president palace without transmitting it over at least one of the cables from the set.

To put its plans into practice the company needs to buy corresponding cables from their current owners. Each cable has some cost. Since the company's main business is not spying, but providing internet connection to home users, its management wants to make the operation a good investment. So it wants to buy such a set of cables, that cables *mean cost* is minimal possible.

That is, if the company buys k cables of the total cost c, it wants to minimize the value of c/k.

Input

The first line of the input file contains n and m ($2 \le n \le 100$, $1 \le m \le 400$). Next m lines describe cables — each cable is described with three integer numbers: servers it connects and the cost of the cable. Cost of each cable is positive and does not exceed 10^7.

Any two servers are connected by at most one cable. No cable connects a server to itself. The network is guaranteed to be connected, it is possible to transmit data from any server to any other one.

Output

First output k — the number of cables to buy. After that output the cables to buy themselves. Cables are numbered starting from one in order they are given in the input file.

Example

network.in	network.out
6 8 1 2 3 1 3 3 2 4 2 2 5 2 3 4 2 3 5 2 5 6 3 4 6 3	4 3 4 5 6
4 5 1 2 2 1 3 2 2 3 1 2 4 2 3 4 2	3 1 2 3

Problem H. Oil Deal

Input file:	`oil.in`
Output file:	`oil.out`
Time limit:	1 second
Memory limit:	64 megabytes

Oil is a very important strategic resource. Recently United States of Antarctica invaded the rich in oil country of Qari, and now try to keep the control of the oil transportation system. The system consists of pipelines that connect different nodes — oil sources and main country cities and ports. It is designed in such a way that it is possible to transport oil from any node to any other one.

However the resisting native forces of Qari are not satisfied with the situation. So they continuously perform terror acts, blowing up some oil pipelines. Recently terrorists have decided to perform a series of explosions and want to hurt the oil pipeline system as much as possible.

For each pipeline the terrorists know the cost of blowing it up. They have a fixed sum of money and want to explode as many pipes as possible for this sum. However, since they still need oil for themselves in different regions of the country, they want the system still be able to transport oil from any node to any other one. Help them to find out which pipes to blow up.

Input

The first line of the input file contains n — the number of nodes, m — the number of pipelines, and s — the amount of money terrorists have ($2 \le n \le 50\,000$, $1 \le m \le 100\,000$, $0 \le s \le 10^{18}$). The following m lines contain information about pipelines — for each pipeline the nodes it connects and the cost of blowing it up is specified (cost does not exceed 10^9).

Oil can be transported along each pipeline in both directions, each two nodes are connected by at most one pipeline.

Output

On the first line of the output file print the maximal number of pipelines terrorists can blow up. On the second line print the numbers of these pipelines (pipelines are numbered starting with 1 as they are listed in the input file).

Example

oil.in	oil.out
6 7 10 1 2 3 1 3 3 2 3 3 3 4 1 4 5 5 5 6 4 4 6 5	2 1 5

Problem I. Bishops on a Toral Board

Input file:	`toral.in`
Output file:	`toral.out`
Time limit:	1 second
Memory limit:	64 megabytes

A bishop is a chess piece that can move in any diagonal direction to any number of cells.

Imagine that we take a board of a size $m \times n$ and connect its top and bottom edges, and its left and right edges, so that it would have a form of a torus. For example, on a 7×10 board the neighbours of a cell $(4, 1)$ would be $(3, 10)$, $(3, 1)$, $(3, 2)$, $(4, 10)$, $(4, 2)$, $(5, 10)$, $(5, 1)$, $(5, 2)$; the neighbours of a cell $(1, 10)$ — $(7, 9)$, $(7, 10)$, $(7, 1)$, $(1, 9)$, $(1, 1)$, $(2, 9)$, $(2, 10)$, $(2, 1)$.

On a toral board chess pieces are no longer limited with the edges, and, for example, on a 7×10 board a bishop from any cell can move to any other cell of the board in one turn, say, from cell $(2, 1)$ to cell $(7, 9)$ by a path $(2, 1) \to (1, 10) \to (7, 9)$.

Let us say that a set of bishops *covers* the board if one can move some bishop to any free cell in one turn. In other words, each cell is either occupied or threatened by some bishop.

Your task is to find out what is the minimal number of bishops one needs to cover the $m \times n$ toral board.

Input

Input file contains m and n separated by a space ($1 \le n, m \le 10^{100}$).

Output

Output the minimal number of bishops needed to cover the $m \times n$ toral board.

Example

toral.in	toral.out
2 2	2
7 10	1