Problem A. Add and Reverse

Input file:	add-and-reverse.in
Output file:	add-and-reverse.out
Time limit:	2 seconds
Memory limit:	256 mebibytes

Consider a non-negative integer x stored in 32 bits of memory:

$$x = b_{31} \cdot 2^{31} + b_{30} \cdot 2^{30} + \ldots + b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 \cdot 2^0$$

where each bit b_i can take two values 0 and 1 independently of other bits.

We perform a sequence of operations with this integer, possibly an empty one. In one operation, we can either increase the number by one or reverse the bits constituting it: swap 31-st bit and 0-th bit, swap 30-th bit and first bit, ..., swap 16-th bit and 15-th bit. We can perform any number of any of these two operations in any order.

What is the minimum possible number of operations required to transform a zero to the given integer n?

The increasing by one is carried out modulo 2^{32}, which means that, if the current number is equal to $2^{32} - 1$, increasing it by one produces a zero.

Input

The only line contains an integer n ($0 \le n < 2^{32}$).

Output

Print one integer: the minimum possible number of operations required to transform a zero to the given integer n.

Examples

add-and-reverse.in	add-and-reverse.out
5	5
2147483648	2

Explanations

In the first example, the fastest way to get a 5 is to increase the number by one five times.

In the second example, we start by producing a one, and then reverse the bits, turning $1 = 2^0$ into $2\,147\,483\,648 = 2^{31}$.

Problem B. Analyze This

Input file:	analyze-this.in
Output file:	analyze-this.out
Time limit:	3.5 seconds
Memory limit:	256 mebibytes

I'm just gonna shake, shake, shake, shake, shake.
I shake it off, I shake it off.

Taylor Swift, "Shake It Off"

Paul Vitti owns one of the most famous New-York mafia cafe-restaurants "The Godfather". We know that its regular customers visit the cafe every day at the same time: i-th customer enters the restaurant at the moment t_i (measured in minutes from the opening).

Each morning, Paul decides which dish will become the soup of the day (sometimes it's not even a soup, but new yorkers are too busy to care about such little things). As a result, cooking one portion of j-th day meal takes D_j minutes.

Regular customers are so regular that they always order one portion of soup of the day as soon as they enter the cafe. Time is money, so when a customer gets his meal, he eats it momentarily and immediately runs on his business. There are always enough chefs and stoves in the kitchen, so every order starts to be cooked at once.

All customers know each other, so they shake hands when they meet. Additionally, everybody must shake Paul's hand upon entering. When an entering customer shakes hand of another customer who is already waiting for his meal, the latter one looks at his watch, and his anger becomes equal to the time he already waited in the cafe today. In the morning, everybody is in good mood, so their anger is equal to zero. If one customer is leaving and another one is entering, they still have time for a handshake.

Paul thoroughly analyzes his customers' behavior. For each day, help him to find such pair of customers that after their handshake, one of their anger values is maximum at that day. After that, he will decide to hire faster chefs, or to eliminate angry customers.

Input

The first line contains n ($1 \le n \le 400\,000$) and m ($1 \le m \le 400\,000$) which are the number of customers and the number of days, respectively. The next line contains n integers t_i ($0 \le x_i < 400\,000$), the entering times. Each of the next m lines contains one integer D_j ($0 \le D_j < 400\,000$) which is the time required to cook one portion of soup on j-th day.

Output

For each day, output two integers i and j ($0 \le i, j \le n$): the indices of people whose handshake can be the last one for one of them. The customers are numbered from 1 to n according to their order in the input. Paul has index 0, and his anger is constantly zero, it's better not to anger him. If there are several possible answers, output any one of them.

Example

analyze-this.in	analyze-this.out
3 4 1 2 5 0 5 1 3	0 1 1 3 1 2 2 3

Problem C. Bipartite Graph

Input file:	bigraph.in
Output file:	bigraph.out
Time limit:	2 seconds
Memory limit:	256 mebibytes

In this problem, you have to construct a bipartite graph which has the following properties:

1. The number of vertices in the first part is d.
2. The number of vertices in the second part is $d - 2$.
3. The number of edges is at most $3d$.
4. For all bipartite graphs constructed by removing two vertices from the first part, there is a perfect matching.

Recall that a bipartite graph is a graph where the vertices are divided into two parts so that each edge connects a vertex from the first part and a vertex from the second part. A perfect matching is a collection of edges such that each vertex of the graph is an end of exactly one edge from that collection.

Input

The first line contains an integer T, the number of test cases ($1 \le T \le 100$). Each of the next T lines contains an integer d, the number of vertices in the corresponding test case ($3 \le d \le 100$).

Output

For each test case, start by printing an integer m, the number of edges, on a separate line. On the next m lines, print the edge descriptions. Each edge description is a pair of integers u and v: the numbers of vertices of the first and the second part connected by that edge ($0 \le u < d$, $0 \le v < d - 2$).

The graph must not contain multiple edges.

Example

bigraph.in	bigraph.out
1 4	7 0 1 1 0 1 1 2 0 2 1 3 0 3 1

Note

You do not need to minimize the number of edges.

Problem D. Bridge Building

Input file:	bridge.in
Output file:	bridge.out
Time limit:	5.5 seconds *(8 seconds for Java)*
Memory limit:	512 mebibytes

A long time ago in 2009...

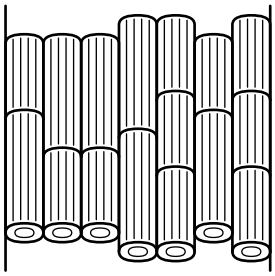
In the village Zaykino, heavy rain is common. After such rain, the river Dubrovka which can usually be just stepped over, overflows. To be able to get across the flooded river, the villagers want to build a floating bridge. Luckily, after construction of a bath-house which belongs to a businessman who settled nearby, there are some logs left.

All remaining logs have the same thickness. There are x logs of length a and y logs of length b.

The bridge will consist of l rows, each of which will be composed of one or more logs. Unfortunately, the last saw in Zaykino drowned in Dubrovka during the previous overflow and disappeared, so the logs can not be cut into pieces.

The chief engineer wants to build a bridge of maximum possible width. The width of a bridge is determined by the minimum width of a row of logs in it.

For example, if the villagers want to build a bridge of seven rows, and there are six logs of length 3 and ten logs of length 2, then they can build a bridge of width 5.



Input

Input contains one or more test cases. Each test case consists of five positive integers x, a, y, b and l. Each of these numbers does not exceed 500. The total number of logs in each test case is at least l.

Let $d = \max(x, a, y, b, l)$. It is guaranteed that the sum of d over all the tests is at most 5000.

Output

For each test case, print an integer on a separate line: the maximum possible width of the bridge.

Example

bridge.in	bridge.out
6 3 10 2 7 10 7 20 9 25 106 126 135 28 137	5 9 112

Problem E. Child's Game with Robot

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

This is an interactive problem.

Little Misha plays a child's game with a robot. In this game, the robot stands on a field which is divided into 3×3 squares and surrounded by borders. The game begins at the center square and consists in making ten actions. Each action is either declamation of a phrase or an attempt to move into one of the neighboring squares. Declamation is used to wait, or simply for amusement.

One of the squares is special, but Misha does not know in advance which one. The robot reports when it moves into the special square. The goal of the game is to move into the special square precisely during the tenth action.

To communicate with the robot, Misha uses text input and output. The following commands are recognized:

- `echo` *phrase* — declamation of *phrase*,
- `move north` — moving one square to the north,
- `move east` — moving one square to the east,
- `move south` — moving one square to the south,
- `move west` — moving one square to the west.

For each declamation command, the robot loudly reads the given phrase, and also prints it back in text. During a declamation, the robot does not move. The robot is guaranteed to correctly handle phrases which consist of characters with ASCII codes from 32 to 126 inclusive and are at most 256 characters long.

For each movement command, the robot prints back one of the four words:

- `bump` if instead of moving, the robot hit a barrier, in which case, it remains on the same square,
- `moved` if the robot successfully moved into a common square,
- `found` if the robot moved into the special square during one of the first nine actions,
- `win` if the robot moved into the special square during the tenth action.

Help Misha to play this game in such a way that, during the tenth action, the robot prints the desired word `win`.

Interaction Protocol

The solution must print commands to the standard output, one command per line. To prevent output buffering, flush the output buffer after each command you issue: this can be done by using, for example, `fflush (stdout)` in C or C++, `System.out.flush ()` in Java, `flush (output)` in Pascal or `sys.stdout.flush ()` in Python.

The answer for each command is printed back to the solution's standard input on a separate line.

After printing ten commands, the solution must terminate correctly.

Example

In the given example, the special square lies to the east from the center. It is guaranteed that this test will be the first when checking your solution.

The commands (solution's **output**) are given on the left, and the answers for them (solution's **input**) are on the right.

command	answer
move north	moved
move east	moved
move south	found
move west	moved
move east	found
move east	bump
move south	moved
echo Ready!	Ready!
echo Steady...	Steady...
move north	win

Note

In each test, the special square is selected in advance and does not change during the contest.

Problem F. Quadruples of Points

Input file:	four-points.in
Output file:	four-points.out
Time limit:	5 seconds *(8 seconds for Java)*
Memory limit:	256 mebibytes

> "I have a rule," — said the manager, — "always do only half of the work."
> "Why do you have such rule?" — asked Cheburashka.
> — It's simple, — said Ivan Ivanovich. — If I'll do all the work and agree with everything, then everybody will say that I'm too kind and that you can do anything under my command. Otherwise, if I'll do no work at all and decline everything, I will be considered lazy and basically worthless. So, doing only half of the work is ideal because nobody can say anything bad about me. Do you understand now?
>
> E. Uspenskiy, "Gena the Crocodile and his Friends"

Geologists of Flatland are planning an important research. For this purpose, they prepared n sets of points, and are going to make measurements at these points. Each set consists of four distinct points, defined by their two-dimensional coordinates.

The prepared sets must be approved by m managers. Each manager chooses a rectangle with sides parallel to coordinate axes and approves measurements at all points inside the rectangle or on its border.

A manager is called *fantastic* if for each given set of points, he approves exactly half of the points. Determine for each manager if he is fantastic or not.

Input

The first line contains two integers n and m: the number of point sets and the number of managers ($1 \le n \le 200\,000$, $1 \le m \le 200\,000$). Then follow n groups of lines, four lines in each group. Each line contains two integers: the coordinates of the next point. In each set, all points are pairwise distinct. The next m lines describe rectangles: four integers x_1, y_1, x_2, y_2 per line ($x_1 \le x_2$, $y_1 \le y_2$).

All coordinates do not exceed 10^9 by absolute value.

Output

For each manager, print "YES" if, for each given set of points, he approves exactly half of the points, and "NO" otherwise.

Example

four-points.in	four-points.out
2 3 0 0 0 1 1 0 1 2 2 0 2 -1 2 -2 2 -3 0 -1 2 0 0 -1 2 1 0 0 0 1	YES NO NO

Problem G. Mosaic Tracery

Input file:	`grid.in`
Output file:	`grid.out`
Time limit:	2 seconds
Memory limit:	256 mebibytes

In a town in far-away Oriental country, there is an art mosaic panel consisting of multi-colored tiles. From time to time, a set of tiles is extracted from the panel and replaced by tiles of another color. So, the mosaic tracery is changing constantly, which is great to attract tourists.

The surface of the mosaic panel is flat, and every tile is a square with side of unit length, so the panel looks like a rectangular grid. The set of tiles which will be extracted is selected by the following algorithm. First, one tile is selected arbitrarily. After that, each new tile which will be selected has to have a common side with at least one of already selected tiles.

The painter which will put a new picture lives in other town, so the set S of tiles to extract is carefully described in the following way. First, consider the set V of all points of the grid which are corners of at least one of the tiles in S. Assign numbers from 1 to N to points of V, where N is their total number. After that, consider the set E of all unit segments of the grid which are borders of at least one of the tiles in S. Note that endpoints of each segment in E belong to V. For each segment in E, write down the numbers assigned to its endpoints. The resulting list of pairs of numbers is the description of the set S.

However, before the description got to the painter, it has fallen into hands of mathematicians. As a result of their interest for various graph transformations the points became renumbered in some chaotic order. Luckily, their numbers are still different integers from 1 to N, and the pairs of numbers in the description still correspond to the set of unit segments E according to the new numbering of V.

The painter now needs a program that reconstructs the initial coordinates of the corners from this altered description. There may be several possible alternatives, but the program may output any one of them because rotation, reflection and parallel translation are not a problem for the painter.

Input

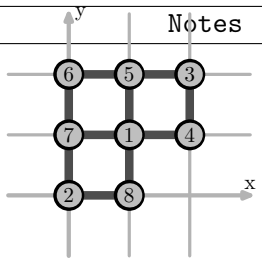
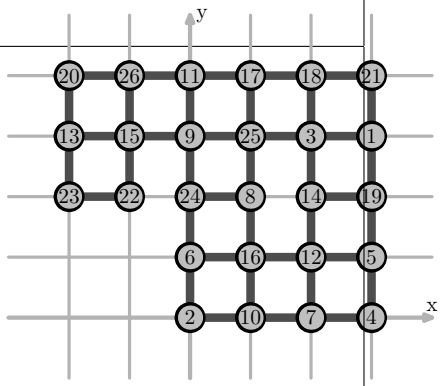
The first line of input contains two integers N and M: the number of points in V and the number of segments in E ($4 \le N \le 100\,000$). Each of the next M lines contains two integers U_i and V_i which is one record from the description of the set S ($1 \le U_i, V_i \le N$, $1 \le i \le M$). It is guaranteed that the input is consistent, that is, it describes some valid set S according to the statement.

Output

Output N lines. Each line must contain two integers X_j and Y_j: coordinates of point number j ($-10^5 \le X_j, Y_j \le 10^5$ for each $1 \le j \le N$).

The answer is correct if all the points in it have different coordinates, and for any pair (U_i, V_i) from the set E, the points U_i and V_i are neighbors on the plane (one of their coordinates is the same, and the other differs by one). If there is more than one possible solution, output any of them.

Examples

grid.in	grid.out	Notes
8 10 1 4 3 4 6 7 2 8 2 7 5 3 5 6 1 7 1 5 1 8	1 1 0 0 2 2 2 1 1 2 0 2 0 1 1 0	
26 39 7 10 25 17 12 14 10 2 9 15 4 7 1 3 23 13 19 14 11 26 15 13 26 20 18 17 16 6 7 12 9 11 21 18 3 18 13 20 22 23 6 24 1 21 16 8 5 12 22 15 19 1 25 9 24 9 3 25 15 26 17 11 12 16 2 6 10 16 4 5 8 25 14 3 5 19 8 24	3 3 0 0 2 3 3 0 3 1 0 1 2 0 1 2 0 3 1 0 0 4 2 1 -2 3 2 2 -1 3 1 1 1 4 2 4 3 2 -2 4 3 4 -1 2 -2 2 0 2 1 3 -1 4	

Problem H. List of Powers

Input file:	list-powers.in
Output file:	list-powers.out
Time limit:	4 seconds
Memory limit:	256 mebibytes

Let p be a prime number and a an integer such that $0 < a < p$. Consider all integers from l to r inclusive which can be expressed as $a^k \bmod p$ for some non-negative integer k. Suppose it is known that there are at most 100 such integers. Print all of them in ascending order.

Input

The only line contains four integers p, a, l, and r separated by spaces ($0 < a < p \le 10^9$, p is prime, $0 \le l \le r < p$).

Output

Print all integers from l to r inclusive which can be expressed as $a^k \bmod p$ for some non-negative integer k. The integers must be printed in ascending order. Separate consecutive integers by spaces. The input is guaranteed to be such that the correct answer contains at most 100 numbers.

Examples

list-powers.in	list-powers.out
5 3 0 3	1 2 3
5 4 2 3	

Explanations

In the first example, we must find all integers from $l = 0$ up to $r = 3$ inclusive which can be expressed as $3^k \bmod 5$ for some integer $k \ge 0$. These are numbers $3^0 \bmod 5 = 1$, $3^1 \bmod 5 = 3$ and $3^3 \bmod 5 = 27 \bmod 5 = 2$. The number 0 can not be expressed this way because 3^k does not divide evenly by 5 for any integer $k \ge 0$. So, we must print the numbers 1, 2, and 3 in ascending order.

In the second example, we must find all integers from $l = 2$ up to $r = 3$ inclusive which can be expressed as $4^k \bmod 5$ for some integer $k \ge 0$. Let us write down the first few such numbers:
$4^0 \bmod 5 = 1$,
$4^1 \bmod 5 = 4$,
$4^2 \bmod 5 = 16 \bmod 5 = 1$,
$4^3 \bmod 5 = 64 \bmod 5 = 4$,
$4^4 \bmod 5 = 256 \bmod 5 = 1$, ...
It can be proved that this sequence contains only numbers 1 and 4. So, the result is an empty list.

Problem I. Potential Well

Input file:	potentials.in
Output file:	potentials.out
Time limit:	3 seconds *(6 seconds for Java)*
Memory limit:	256 mebibytes

Billionaire Christian has learned today about Dijkstra's algorithm with Johnson's potentials. He liked it so much that he decided to add potentials to all of his favorite graphs. Christian's tastes are very peculiar — he keeps only directed weighted graphs in his secret room.

Let us recall how potentials work. Each vertex v is assigned a potential: a real number $\varphi(v)$. If there was an edge from u to v with initial weight w, then its new weight will be $w' = w - \varphi(v) + \varphi(u)$.

The strength of a graph is defined as the minimum of its edges' weights. Christian does not want weak graphs in his room, so he wants to choose potentials in such way that the strength of the graph is maximized. Help him do that.

Input

The first line contains two integers n ($2 \le n \le 1000$) and m ($1 \le m \le 100\,000$), the number of vertices and edges in Christian's favorite graph. Each of the next m lines contains three integers u_i, v_i and w_i describing an edge: its start, end and weight, respectively. ($1 \le u_i, v_i \le n$, $|w_i| \le 10^6$).

Output

The first line must contain the maximum strength of the graph. If it is possible to make the answer infinitely large, print the string "+inf" instead.

If the answer is finite, the second line must contain n real numbers: the potentials of vertices. Each of them must not exceed 10^{10} by absolute value. Your answer must differ from the correct one, and also from the answer computed based on your potentials, by no more than 10^{-5}.

Examples

potentials.in	potentials.out
3 3 1 2 1 2 3 1 3 1 1	1.0 0.0 0.0 0.0
2 1 1 2 1	+inf

Problem J. Steiner Tree in Random Graph

Input file:	`random-steiner.in`
Output file:	`random-steiner.out`
Time limit:	2 seconds *(3 seconds for Java)*
Memory limit:	256 mebibytes

You are given a weighted graph on n vertices. Your task is to find a connected subgraph of minimal total weight which contains all vertices with numbers from 1 to $n - k$ inclusive.

To make the problem easier to solve, and the tests easier to generate, all tests except the examples are generated using the following algorithm:

- Integers n, m, and k are chosen by the jury.
- All $\frac{n \cdot (n-1)}{2}$ possible edges are considered in random order.
- When being considered edge is added to the graph if, after it is added, it is still possible to add zero or more of the following edges and get a connected graph with at most m edges in total.
- For each edge, its weight is an integer chosen uniformly at random from the segment $[50, 100]$, independently from other edges.

Input

The first line contains three integers n ($5 \le n \le 100$), m ($2 \cdot n \le m \le 10 \cdot n$) and k ($0 \le k \le 30$). Each of the next m lines holds three integers a_i, b_i and w_i ($1 \le a_i < b_i \le n$, $50 \le w_i \le 100$) which mean that vertices a_i and b_i are connected by an edge of weight w_i.

It is guaranteed that the graph has no loops and no multiple edges. Additionally, it is guaranteed that in all tests except the examples, the graph is generated by the algorithm described above.

Output

On the first line, print one integer c, the number of edges in the chosen subgraph. On the next c lines, print the chosen edges: the numbers of vertices connected by each of them. If there are several possible answers, print any one of them.

Examples

random-steiner.in	random-steiner.out
5 10 3 1 2 50 1 3 51 1 4 52 1 5 53 2 3 54 2 4 55 2 5 56 3 4 57 3 5 58 4 5 59	1 1 2
5 10 2 1 2 100 1 3 100 1 4 57 1 5 56 2 3 100 2 4 54 2 5 53 3 4 52 3 5 51 4 5 50	3 5 2 5 3 1 5

Problem K. Rotation Transformation

Input file:	rotate-this.in
Output file:	rotate-this.out
Time limit:	2 seconds
Memory limit:	256 mebibytes

A simple motion over and over...

t.A.T.u., "A Simple Motion"

Julia loves simple motions very much, especially in three-dimensional space! Recently, she found a matrix that corresponds to some rotation around an axis that goes through the origin.

A rotation matrix A moves point $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ to $\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}$ like this:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} \times \begin{pmatrix} x \\ y \\ z \end{pmatrix}.$$

A rotation can also be described geometrically by a unit (that is, of length 1) vector $v = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix}$ around which the rotation takes place, and a rotation angle α. Suppose you look at the origin from the end of v. Then, if rotation is going counter-clockwise, angle α will be positive, otherwise, it will be negative.

Julia wants to express the rotation given by a matrix geometrically. Unfortunately, Julia does not know the values of v and α, therefore, she asked you to find them, so that she could continue her simple motion.

Input

You are given the matrix A: three lines with three **real** numbers on each. Each number is given with at most 20 digits after the decimal point. It is guaranteed that there exist such v and α that $1 \le |\alpha| \le 179$, and additionally, the elements of matrix A' which corresponds to a rotation around vector v by angle α differ from the corresponding elements of A by at most 10^{-13}.

Output

On the first line, print the angle α in degrees. On the second line, print coordinates of the unit vector v. Output all numbers as precisely as possible. Your answer will be considered correct if the difference between A_{ij} and the matrix constructed from your angle and unit vector will not exceed 10^{-6}.

Example

rotate-this.in	rotate-this.out
0 -1 0 1 0 0 0 0 1	90 0 0 1