Problem A. BibTEX

Input file:	`bibtex.in`
Output file:	`bibtex.out`
Time limit:	1 second
Memory limit:	256 megabytes

BibTeX is a tool for formatting lists of references. The BibTeX tool is typically used together with the LaTeX document preparation system. BibTeX uses a style-independent text-based file format for lists of bibliography items, such as articles, books, theses.

In this problem you will have to emulate the part of BibTeX to create bibliography based on book and article references description.

Reference description starts with "`@reference type`" followed by '{' followed by a list of fields formatted as "`name = "value"`", separated by commas, followed by '}'.

Your program must support the following reference types:

- **article** — An article from a journal or magazine.

  Required fields: author, title, journal, year

  Optional fields: volume, number, pages
- **book** — A book with an explicit publisher.

  Required fields: author, title, publisher, year

  Optional fields: volume

The fields are formatted as follows:

Field	Description
author	Each author is formatted as "`Name1 Name2 ... Surname`". There are at least 1 and at most 10 names. If there are several authors, they are separated by "`and`". No author has name or surname equal to "`and`". The total length of the field doesn't exceed 200 characters. Names and surnames contain only letters of the English alphabet and are separated by a space.
title	Title of the source. A string of up to 200 characters containing letters of the English alphabet, spaces, digits and punctuation.
journal, publisher	Journal name or publisher name. A string of up to 200 characters containing letters of the English alphabet, spaces, digits and punctuation.
year	Integer number from 1500 to 2008.
volume	Integer number from 1 to 10^6.
number	Integer number from 1 to 10^6.
pages	Formatted as "`from--to`" or as "`page`". "From", "to" and "page" are integers from 1 to 10^6, "from" < "to".

Authors of each source are sorted by surname, then by first name, second name, etc. After that all references are sorted by first author surname, then first author first name, then first author second name, etc, then by second author surname, etc, (if corresponding name doesn't exist, empty string is used instead) then by title. No two references have the same set of authors and the same title (except books that can have several volumes, such references are sorted by the volume).

Each article reference is formatted as:

"`Authors Title // Journal[, Volume][ (Number)] -- year[ -- pages]`".

Here "`[...]`" means optional part. Authors are separated by comma. Each author is formatted as "`Surname I1. I2. ...`" where I1, I2, etc are author's initials (the first letter of the author's corresponding name). Pages are formatted either as "`p. page`" if there is only one page, or "`pp. from--to`" if there are many.

Each book reference is formatted as:

"`Authors Title[, Vol. Volume] -- Publisher, Year`".

All references are numbered starting from 1 and preceded by their number in square brackets.

See example for further reference.

Input

Input file contains BibTeX reference list containing up to 100 references. All entries in the input file are case sensitive. All elements of input file are separated by at least one space and/or line feed.

Output

Output the bibliography in the required format.

Example

bibtex.in

```
@book
{
    author = "Donald Ervin Knuth",
    title = "The Art of Computer Programming",
    volume = "1",
    publisher = "Addison-Wesley Professional",
    year = "1997"
}

@book
{
    author = "Donald Ervin Knuth",
    title = "The Art of Computer Programming",
    volume = "2",
    publisher = "Addison-Wesley Professional",
    year = "1997"
}

@article
{
    author = "Robert Endre Tarjan and Andrew Goldberg",
    title = "A new approach to the maximum flow problem",
    journal = "Journal ACM",
    volume = "35",
    year = "1988",
    pages = "921--940"
}
```

bibtex.out

```
[1] Goldberg A., Tarjan R. E. A new approach to the maximum flow problem // Journal ACM, 35 -- 1988 -- pp. 921--940
[2] Knuth D. E. The Art of Computer Programming, Vol. 1 -- Addison-Wesley Professional, 1997
[3] Knuth D. E. The Art of Computer Programming, Vol. 2 -- Addison-Wesley Professional, 1997
```

Problem B. Signed Derangements

Input file:	`derangements.in`
Output file:	`derangements.out`
Time limit:	1 second
Memory limit:	256 megabytes

Signed permutation of size n is an ordered set of n numbers ranging from $-n$ to n except 0, where absolute values of any two numbers are different. An example of a signed permutation is $\langle 4, -2, 3, -5, -1 \rangle$. Clearly, there are $2^n n!$ signed permutations of size n.

A signed permutation $\langle a_1, a_2, \ldots, a_n \rangle$ is called a *signed derangmenent* if $a_i \neq i$ for all i. For example, $\langle 4, -2, 3, -5, -1 \rangle$ is not a signed derangement, but $\langle 4, -2, -3, -5, -1 \rangle$ is.

Given n, find the number of signed derangements of size n.

Input

Input file contains one integer number n ($1 \le n \le 200$).

Output

Output one integer number — the number of signed derangements of size n.

Example

derangements.in	derangements.out
2	5

The following signed 2-permutations are signed derangements: $\langle 2, 1 \rangle$, $\langle 2, -1 \rangle$, $\langle -2, 1 \rangle$, $\langle -1, -2 \rangle$, and $\langle -2, -1 \rangle$.

Problem C. Electricity 2

Input file:	`electricity.in`
Output file:	`electricity.out`
Time limit:	1 second
Memory limit:	256 megabytes

The Power Antarctica company owns most electricity infrastructure in the young Antarctica Federation. The electricity network in Antarctica consists of n nodes — generating and consuming stations, and m power lines connecting nodes. Each power line is directed, that is, power can be transmitted along it only in one direction.

When the system was first planned there was a power line connecting every two nodes in one of the directions, so there were $n(n-1)/2$ power lines. However, later some of the power lines were dismantled to prevent shortcut circuits. A total of k lines were dismantled (so $m = n(n-1)/2 - k$), and now the system satisfies the condition that there are no three nodes u, v and w such that power lines between them form a directed triangle (so, power lines uv, vw and wu cannot exist simultaneously).

Now the power use of consumers has increased, and even longer circuits are dangerous, so Power Antarctica would like to dismantle some other power lines as well. It would like to dismantle up to k power lines, so that there were no directed cycle in the power network. You are hired to plan the procedure. You must not minimize the number of lines to dismantle, but this number must not exceed k.

Input

The first line of the input file contains two integer numbers: n and m ($1 \le n \le 100$, $0 \le m \le n(n-1)/2$). The following m lines describe power lines, each power line is described by two integer numbers — the number of node it starts at and the number of node it ends at. There is no directed triangle in the power network.

Output

The first line of the output file must contain l — the number of power lines to dismantle. This number must not exceed $n(n-1)/2 - m$. The second line must contain l integer numbers — the numbers of power lines to dismantle. After dismantling the listed lines there must be no directed cycle in the power network. Power lines are numbered starting from 1 in the order they are given in the input file.

If there is no solution, output -1 at the first line of the output file.

Example

electricity.in	electricity.out

4 4
1 2
2 3
3 4
4 1
	1
	1

Problem D. Currency Exchange

Input file:	`exchange.in`
Output file:	`exchange.out`
Time limit:	1 second
Memory limit:	256 megabytes

Josh is the owner of the currency exchange office in Flatland. Recently Josh has ordered a new set of metal digits to put the rate of exchange at the tableau in front of his office. Unfortunately, the set of digits contains exactly two copies of each digit. Now Josh wonders whether he will be able to show the exchange rate at his office. Josh exchanges Flatland dallars to Edgeland tuhriks. Josh knows that the exchange rate is a positive integer number ranging from l to r, inclusive. Help Josh to find out how many different rates he can actually display.

Input

Input file contains two numbers l and r ($1 \le l \le r \le 10^{18}$).

Output

Output one number — the number of exchange rates Josh can display using his set.

Example

exchange.in	exchange.out
`1 1000`	`990`

Problem E. New Mayors

Input file:	`mayors.in`
Output file:	`mayors.out`
Time limit:	1 second
Memory limit:	256 megabytes

The country of Freeland has n cities, some of them are connected by bidirectional roads. For national security reasons, the following condition is satisfied for each city u. Let us denote by $N(u)$ the set of cities that u is connected by roads with. Then for any two cities v in w in $N(u)$ there is a way to get from v to w by roads, visiting only cities from $N(u)$ (but not u). Also it is possible to get from any city to any other city by roads.

Now the president of the country want to assign new mayors to the cities in Freeland. The mayors assigned will belong to three main political parties of Freeland: Right Freeland, Great Freeland and Brave Freeland. Since the president doesn't want any party to get too strong, he would like no two cities such that their mayors belong to the same party be connected by a road. You, the Minister of Internal Affairs of Freeland, was assigned the task to find which city would have a mayor from which party.

Input

The first line of the input file contains two integer numbers: n and m — the number of cities and roads in Freeland, respectively ($1 \le n \le 500$, $0 \le m \le 10\,000$). The following m lines describe roads, each road is described by the cities it connects. No road connects a city to itself, any two cities are directly connected by at most one road.

Output

If it is possible to assign mayors to cities in such a way that no two mayors from the same party govern the cities connected by a road, output "`Plan OK`" at the first line of the output file. The second line must contain n characters, for each city output '`R`' if it must be governed by a mayor from Right Freeland, '`G`' for Great Freeland, or '`B`' for Brave Freeland.

If the task is impossible to complete, output "`Plan failed`" at the first line of the output file.

Example

mayors.in	mayors.out

5 8
1 2
1 3
1 4
1 5
2 3
3 4
4 5
2 5
	Plan OK
	RGBGB

4 6
1 2
1 3
1 4
2 3
3 4
4 2
	Plan failed

Problem F. $\sqrt{Nim}$

Input file:	`nim.in`
Output file:	`nim.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Square Root Nim game is played using the following rules. Two players have a pile of n stones. They alternatively take stones from the pile. Each turn if the pile contains k stones the player can take from 1 to $\lfloor\sqrt{k}\rfloor$ stones, inclusive, from it. For example, if there are 10 stones, the player can take 1, 2 or 3 stones. If there are no stones left, the player who must make a turn loses.

Given n, find out whether the first player to make a turn would win if both players play optimally.

Input

The input file contains one integer number n ($1 \le n \le 10^{12}$).

Output

Output "`WIN`" if the first player will win, or "`LOSE`" if the first player will lose.

Example

nim.in	nim.out
3	WIN
5	LOSE

Problem G. Pulp Fiction

Input file:	`pulp.in`
Output file:	`pulp.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

John Thamesoff is writing detective stories. His stories are not very clever, neither they have suspense, or so, but readers like his stories because he has an easy-reading language, and they help to relax after a hard working day.

The main secret of John is that he doesn't actually invent the plots of his stories by himself. His brother Phil gives him ideas, and John writes the stories. Of course, John shares part of his income with Phil. Unfortunately Phil's activity in creating plots depends on his biorhythms. So, for instance, he cannot invent new stories every day. John and Phil calculated that Phil will be able to create n new plots, the i-th of the plots will be ready on r_i-th day. There can be several plots ready on the same day.

For every plot Phil will create John estimated the time he will need to write the story. It will take p_i days for him to complete the i-th one. John has good memory, so he can temporarily leave the story, switching to another one, should this be useful.

Of course, brothers want to have stories ready as soon as possible. Actually they would like to minimize the average time from now till the story will be ready. Let the i-th story be ready by day c_i, then John would like to minimize $\sum c_i$.

Given Phil's biorhythms and John's productivity, help brothers to develop the optimal schedule for writing.

Input

The first line of the input file contains n — the number of stories brothers are planning to write ($1 \le n \le 100\,000$). The following n lines contain two integer numbers each: r_i and p_i ($1 \le r_i, p_i \le 10^9$).

Output

Output one number — the minimal possible value of $\sum c_i$ brothers can achieve.

Example

pulp.in	pulp.out

2
1 5
2 1
	10

Phil gets an idea on day 1. John starts writing the story. On day 2 Phil gets the new idea. John switches to the new story and completes it in one day (on day 3, so $c_2 = 3$). Now Johns returns to the first story he was writing and completes it in another four days (on day 7, so $c_1 = 7$ and $\sum c_i = 10$).

Problem H. Settling the Universe Up

Input file:	`settling.in`
Output file:	`settling.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

The colonists from Earth are settling the Universe up. More and more young families choose to leave the Earth for good in order to bring the human race to the new heights. Well, actually they just have no way back because hypertunnels that are used for the colonization are one-way.

You work in the coordination center of the colonization. There are n planets that are used in the process of colonization, numbered from 1 to n. Earth has number 1. Brave pathfinders sometimes put a hypertunnel from one planet to another. The planets are numbered in order of decreasing of their torsion power, so a hypertunnel can only exist from a planet with smaller number to a planet with greater number. Sometimes, due to torsion fields distortion, hypertunnels disappear. Your task is to trace all such events, quickly answer questions whether there exists a path from one planet to another, and provide some other statistics.

After some period of working with pen and paper you decide to write a program to do the work for you.

Input

The first line of the input file contains n — the number of planets ($1 \le n \le 200$), and m — the number of hypertunnels that exist at the moment you start writing the program. The following m lines describe hypertunnels, each hypertunnel is described by two integer numbers: u and v — the planet where the tunnel starts and the planet where the tunnel goes ($1 \le u < v \le n$).

The next line contains k — the number of events and queries ($1 \le k \le 100\,000$, the number of events doesn't exceed 1000). The following k lines contain information about events and queries. Each event is either "`+` i j" if the hypertunnel between i and j is added, or "`-` i j" if the hypertunnel between i and j has disappeared. No tunnel is created if there is a tunnel already, a tunnel can disappear only if it exists. Each query is "`?` i j" and asks whether there is a path from planet i to planet j. In all events and queries $i < j$.

Output

Output $k+1$ lines. The first line must contain the number of pairs of planets (u, v) such that it is possible to get from planet u to planet v by hypertunnels that initially exist. After each event output the updated number of pairs. After each query output "`YES`" or "`NO`".

Example

settling.in	settling.out
4 4	5
1 2	NO
2 4	6
1 3	6
3 4	3
6	4
? 2 3	YES
+ 2 3	
- 1 3	
- 1 2	
+ 1 4	
? 2 3	

Problem I. Segment Transformations

Input file:	`transform.in`
Output file:	`transform.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Fred is working in a laboratory called *Bioinformatics and Electrical Engineering Research* (BEER). His recent work is related to investigation of strings transformation. Now he is working with segment transformations.

Segment transformation $T(i, j, k)$ of a string s that consists of DNA characters ($\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$) is performed in the following way. Characters of s at positions between i and j, inclusive, are replaced by characters that are k positions later in the `A`, `C`, `G`, `T` order ($1 \le k \le 3$). The order is considered circular, so `T` is followed by `A`. For example, after applying $T(2, 5, 2)$ to a string "`AGGTCAT`" we get a string "`AAACTAT`".

Given two strings s and t of equal length, Fred wonders, what minimal number of segment transformations is needed to convert s to t. Help him to find that out.

Input

The first line of the input file contains s. The second line contains t. Both strings have equal length, ranging from 1 to 100, inclusive, and contain letters from the set $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$.

Output

The first line of the output file must contain k — the minimal number of segment transformations needed to convert s to t. The following k lines must contain three integer numbers each and describe transformations.

Example

transform.in	transform.out

AGGTCAT
AAACTAA
	2
	2 5 2
	7 7 1

Problem J. Zen Garden

Input file:	`zen.in`
Output file:	`zen.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Zen garden is a small enclosed area containing a number of rocks. The visitor of a Zen garden sits down to the grass, relaxes and contemplates the rocks. The secret of some Zen gardens is that there is only one place from which all rocks are completely visible. The person who sits in the secret place gets to the next stage of Zen.

Young Niyoka Nasisimoto recently discovered a new Zen garden and wonders if there is a place from which all rocks are completely visible. The garden Niyoka has found is quite unusual — the rocks in it have a shape of tall circular columns of various radii. The garden has a rectangular form of $x \times y$ meters. The boy has painted the plan of the garden on a sheet of paper and started looking for the secret place. Help him find whether there exists a point in the garden, such that all rocks are completely visible from it, that is, any ray starting at the point intersects the interior of at most one rock (but may touch several ones). The point must not be inside any rock (but may be on its border).

Input

The first line of the input file contains three integer numbers: n, x and y — the number of rocks in the garden and the size of the garden ($1 \le n \le 10$, $4 \le x, y \le 10^4$).

Let us introduce the coordinate system in such a way that the corners of the garden are at points $(0, 0)$, $(x, 0)$, $(0, y)$, and (x, y). The following n lines contain three integer numbers each — coordinates of centers of rocks and their radii. All rocks are inside the garden, no two rocks intersect (although they may touch each other).

Output

If there is a point inside the garden, from which all rocks are completely visible, output its coordinates at the first line of the output file. In the other case, print "`No Zen`". Print as many digits after the decimal point as you can. Verifying program will use precision of 10^{-5} in its floating point operations.

Example

zen.in	zen.out

4 10 10
2 2 2
8 8 2
2 8 2
8 2 2
	5.0 5.0

5 10 10
2 2 2
8 8 2
2 8 2
8 2 2
5 5 1
	No Zen