Problem A. Detect a Mood

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Lets call *regular bracket sequence* a sequence from the opening (‘(’) and closing (‘)’) brackets, such as

- empty sequence is regular bracket sequence;
- if a is regular bracket sequence, then (a) is reguar bracket sequence;
- if a and b are regular bracket sequences, their concatenation is regular bracket sequence.

Giga found a regular bracket sequence and decided to add some mood in it. For this purpose he added some additional brackets, representing happy (‘)’) and sad (‘(’) smileys. Note that Giga’s brackets may not form the regular bracket sequence.

Given the resulting sequence, find out overall mood of the new sequence — difference between number of happy and sad smileys, added by Giga.

Input

Input consists of one line, containing non-empty string consisting of ‘(’ and ‘)’ brackets. String is not longer than 300 characters.

Output

Print one integer: difference between number of happy and sad smileys, added by Giga.

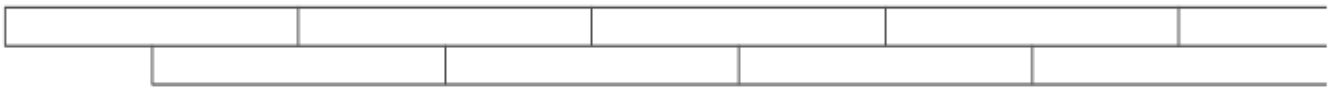
Example

standard input	standard output
(()(())	-1
)(	0

Problem B. Painting Tracks

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Identical uncolored tiles 2×1 laid out as the two tracks in next way. Upper row is composed of N tiles with upper left corners $(2, 1)$, $(4, 1)$, ..., $(2N, 1)$, while second is compised of M tiles with upper left corners $(3, 0)$, $(5, 0)$, ..., $(2M + 1, 0)$.



You are given 3 different colors and must paint the tracks in such a way that each tile must be completely painted in one of those colors and two adjacent tiles need to be painted in a different color.

Determine the number of different possible paintings.

Input

Input contains two integers N and M ($1 \le N, M \le 10^{17}$, $|N - M| \le 55$).

Output

Print one integer — number of different paintings.

Examples

standard input	standard output
2 2	6
3 1	12

Problem C. The Robot on the Plane

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

The robot moves along the plane step-by-step by the following rules:

1. The length of the first move is 1, the length of each subsequent move is exactly three times longer than the previous one.
2. At each step robot may either to rest ('S'), i.e. do not move, or select one of the four directions — up ('U), down ('textttD'), left ('textttL') or right ('textttR') and then step in the selected direction by a step length equal to the length of the move.

Given two points (x_0, y_0) and (x_1, y_1) on the plane, determine, whether the robot can start his travel in the first point and reach second point. If second point can be reached, print the sequence of letters, correcponding to robot's moves. If there is more than one answer, print any of them.

Input

First line of the input contains four integers — coordinates of starting point x_0 and y_0 and coordinates of ending point x_1 and y_1. All coordinates does not exceed 10^{17} by absolute value.

Output

If it is impossible for robot to reach the ending point, printf 'NO' in the first line. Otherwise print 'YES', and in second line print the sequence of steps — not more than 10^5 characters 'U', 'D', 'R', 'L', 'S'.

Note that 'S' may be last move only in case, when no more steps exist in the route (i.e. only when points coincide).

Examples

standard input	standard output
0 0 1 1	NO
1 1 4 2	YES UR
-732 -732 -732 -732	YES S

Problem D. Match of the Giants

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

As you know, some universities have a wonderful tradition to hold the so-called "Battle of the Giants": coach of each university selects N teams from each university, and then summary results of selected teams for those universities at same programming contest are compared.

Very reputable universities GTU and TSU use their own way to calculate results of match: each coach selects ordered list of teams and then teams play one vs. one: if i'team in list of GTU is placed higher in standings than i-th team in list of TSU, then GTU scores for i-th team, else TSU scores (consider that places cannot be shared, scoring team adds 1 to score, other team's score not changed). So final score for each university is sum of scores for all N pairs. In this way order of teams in list is very important, so why coaches keep their lists in secret till the contest starts.

Today the Battle between GTU ad TSU was postponed due to the coincidence in time with an important matsh of Georgian rugby team. So bookmakers decided to use this pause to calculate odds for the upcoming Battle.

Bookmakers somehow got unordered list of teams from each side with information about team's strength; it is guaranteed that more strong team will always score against weaker one. It happened that no two teams from different universities have same strength.

But bookmakers does not know ordering, which was selected by coaches, so they want you to calculate maximal possible score for each university,

Input

The first line of the input consists of one integer N — number of teams from the each university ($1 \le N \le 2 \cdot 10^5$). Second line contains N integers G_i ($1 \le G_i \le 10^5$) — strengths of GTU teams. Third line contains N integers T_i ($1 \le T_i \le 10^5$) — strengths of TSU teams. It is guaranteed that $T_i \ne G_j$ for any $1 \le i, j \le N$.

Output

Print two integers — maximal possible score for GTU and maximal possible score for TSU.

Examples

standard input	standard output
4 7 1 5 3 2 4 6 4	3 3
1 2 3	0 1
5 2 3 3 2 3 4 1 1 1 4	3 2

Example explanation

In first example the GTU teams are having the strengths of $\{7, 1, 5, 3\}$, and TSU teams — strengths $\{2, 4, 6, 4\}$, respectively. Then, GTU can score a maximum of 3 points with ordering $\{7, 1, 5, 3\}$ vs TSU $\{6, 4, 4, 2\}$ (GTU scores in pairs 1,3,4 and TSU scores in pair 2). At other side, in orderings $\{1, 3, 5, 7\}$ for GTU vs $\{2, 4, 6, 4\}$ for TSU TSU can score 3 points as well (in pairs 1,2 and 3). So both universities may score maximum of 3 points.

Problem E. Billiards

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

The ball is standing in point A on the rectangular pool table. Ball is pushed with constant speed and direction in such a way that it reaches point B before first contact with the billiard's border.

Considering ball as a point, reflections as ideal and neglecting effects of frictional forces, check if ball will, may be after some reflections, reach the point C.

Input

First line of the input contains two positive integers x_1 and y_1 — coordinates of right upper angle of the billiard table, while lower left angle of the table is placed at the origin.

Second line contains two integers x_a and y_a ($0 \le x_a \le x_1$, $0 \le y_a \le y_1$) — starting coordinates of the ball (point A). Third line contains two integers x_b and y_b ($0 \le x_b \le x_1$, $0 \le y_b \le y_1$) — coordinates of point B.

Fourth line contains two integers x_c and y_c ($0 \le x_c \le x_1$, $0 \le y_c \le y_1$) — coordinates of point C.

Neither two of three points A, B and C can coincide.

Output

Print "YES" if after some (possibly zero) number of reflections ball will reach point C and "NO" otherwise.

Example

standard input	standard output
10 10 2 1 5 8 6 3	YES
2 2 1 0 0 1 1 1	NO

Problem F. Passage

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 megabytes

> *Once upon a time a farmer went to a market and purchased a fox, a goose, and a bag of beans. On his way home, the farmer came to the bank of a river and rented a boat. But in crossing the river by boat, the farmer could carry only himself and a single one of his purchases - the fox, the goose, or the bag of beans. If left together, the fox would eat the goose, or the goose would eat the beans. The farmer's challenge was to carry himself and his purchases to the far bank of the river, leaving each purchase intact. How did he do it?*
>
> (An old puzzle)

Afanasiy and Anatoliy work as the zoo caretakers at the famous Novosibirsk Zoo, and they have to solve problems much worse than the rescue of beans and a goose. For example, once two of them had to carry across the Ob river all zoo animals in two boats! You are invited to feel yourself in their skins.

There are N animals in the zoo. If you leave some of the animals together unattended, it is very likely that one will eat another. It is known for each animal what other animals it can eat. Each zoo caretaker is a great expert in his work and can deal even with a tiger! When there is at least one caretaker near the animal, it definitely cannot eat anyone.

Zoo caretakers have two boats. According to the terms of insurance, it is forbidden to put two or more animals into the same boat at the same time. In any boat at any given time, there may be no more than one caretaker and no more than one animal. Initially, all the animals, caretakers, and boats are located on the left bank of the river. All of them must be transfered to the right bank of the river. Of course, there are no other banks of the river, except for the left and right ones.

Caretakers can float on boats between the two banks as they please. Both boats are rowing, so they cannot move without a caretaker. If at any point, there is such a pair of the animals on the same bank that one of them can eat the other one, and there is no caretaker on this bank, then something irretrievable happens. Of course, this must be avoided: all the animals should get to the right bank of the river safe and sound.

You need to determine whether it is possible to carry out the crossing.

Input

In the first line of the input file there is one integer N, which is the number of animals in the zoo ($2 \le N \le 200$). Each of the animals has an inventory number ranging from 1 to N inclusively.

The following N lines contain information about gastronomic passions of the animals. i-th of them describes which animals can eat the animal with the inventory number i ($1 \le i \le N$). The line begins with a non-negative integer k_i, being the number of animals which can eat the animal i, followed by k_i positive integers, which are the inventory numbers of those ravenous animals. These

numbers are different and lie in the range from 1 to N. None of these numbers can be equal to i because no animal can eat itself.

The total number of "predator-prey" pairs, which is equal to the sum of all numbers k_i, does not exceed 1500.

Output

If it is possible to carry across all animals safe and sound, print "`Hurrah!`". If not, print "`Fired.`".

Examples

standard input	standard output
5 1 2 1 3 1 4 0 0	Hurrah!
5 4 2 3 4 5 4 3 4 5 1 4 4 5 1 2 4 5 1 2 3 4 1 2 3 4	Fired.

Problem G. Files list

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 megabytes

You are given a set of files, and you are to output the number of files with each extension. The file extension is a sequence of characters in the name of the file after a dot character.

The file system is case-sensitive: even if the file names differ only by a characters case, they are still considered to be different.

Input

In the first line of the input file there is an integer N, which is number of file names ($1 \le N \le 10^3$). In each of the following N lines there is a file name that is no more than 200 characters in length. The file name consists of only lower and upper Latin letters, numbers and a dot character '.'. It is guaranteed that a dot character can be found in the file name exactly once. Also, there is at least one symbol before and after the dot. It is guaranteed that each file name is present only once in the input file.

Output

For each of the extensions that are present in the input file, output the number of files with this extension in the form of `<extension>:␣<number>`. Output extensions in order of the first mention in the input file.

Example

standard input	standard output
6 218052.pdf Movie00.mkv Invoice.xls book.pdf book.epub Movie01.mkv	pdf: 2 mkv: 2 xls: 1 epub: 1

Example explanation

Two files with extension `pdf`: `218052.pdf`, `book.pdf`.

Two files with extension `mkv`: `Movie00.mkv`, `Movie01.mkv`.

One file with extension `xls`: `Invoice.xls`.

One file with extension `epub`: `book.epub`.

Problem H. Housing payments

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 megabytes

Being a decent citizen, Ippolit Matveevich pays his monthly housing payments the same day he receives his receipt. When he got to the bank recently, he found out that a new tariff system was implemented.

For now on he has to pay extra X_i rubles, if he pays during the i-th month, as commission to the bank, regardless of the actual sum being payed. At first, it seems like if he pays rarely, then he will save money on commission. But the system works smarter: after each month the total amount of debt increases by p_i percent. Therefore, it is the choice: either to pay rarely with smaller commissions but bigger fine, or to pay more regularly with smaller fines but bigger commission payments.

Help Ippolit Matveevich to find the way to minimize the total sum of his payments.

All payments must be completed in first N month; its impossible to make a payment without commission; its impossible to pay "in advance" for future month.

Input

In the first line of the input file there is one integer N, being number of months for which he has to make payments ($1 \le N \le 10^5$).

In each of the following N lines there are three integers S_i, x_i, p_i, where S_i is the principal amount of the payment received in a receipt on the i-th month, x_i is the commission for payments on i-th month, and p_i is the percentage by which the sum of debt will increase at the end of the month ($10^3 \le S_i \le 10^4, 10 \le x_i \le 500, 1 \le p_i \le 10$).

Output

The output file should contain one real number, being the minimum possible amount of money Ippolit Matveevich has to pay for the entire period. The answer must be printed with an absolute or relative error not greater than 10^{-8}.

Examples

standard input	standard output
3 5000 20 1 5000 20 1 5000 20 1	15060
3 5000 500 1 5000 200 1 5000 100 3	15250.5

Example explanation

In the first case, it is more favorable to pay each month. In the second one paying once at the

end of three months is cheaper.

Problem I. Arithmetic expressions

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	256 megabytes

Pupil Peter composes arithmetic expressions from the symbols '0'-'9', parentheses '(' and ')', and symbols '+' and '-' (plus and minus signs). Each composed expression can be either a string containing the decimal representation of an integer (without leading zeros) lying in the range from 0 to $M-1$ inclusive, or a string of the form (E), where E is also an expression, or a string of the form $E_1 \sigma E_2$, where E_1 and E_2 are also expressions and σ is a symbol '+' or '-'.

Peter wants to count the number of different expressions of the fixed length N, for which the remainder after dividing the value by M is equal to P. The value of an arithmetic expression is calculated by the school rules of arithmetic. Please note: the remainder of the division is always within the range from 0 to $M-1$, even if the value is negative.

Since the answer can be large, you should print its remainder modulo 10^9+7.

Input

The first line of the input file consists of three integers N, M and P, where N is length of expressions, M is modulus used for division, P is required remainder of the division ($1 \le N \le 50$, $0 \le P < M \le 200$).

Output

The output file should contain one integer — the number of all expressions that have given length and given value of the remainder of division. You should output the answer modulo 10^9+7.

Examples

standard input	standard output
3 100 2	12
3 100 98	8
2 100 98	1

Example explanation

All sought-for expressions for the case $N = 3$, $M = 100$, $P = 2$:

(2) 0+2 1+1 2+0 2-0 3-1 4-2 5-3 6-4 7-5 8-6 9-7

All sought-for expressions for the case $N = 3$, $M = 100$, $P = 98$:

0-2 1-3 2-4 3-5 4-6 5-7 6-8 7-9

All sought-for expressions for the case $N = 2$, $M = 100$, $P = 98$:

98

Problem J. Novice urbanist

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 megabytes

Lately New State University (NSU) is being constructed rapidly. New buildings, dormitories and housings keep appearing here and there along the legendary Student Avenue. Vasya enjoys walking along his favorite avenue, however, the problem is that different buildings of NSU are located on opposite sides of the street. In order to get from one building to another it is sometimes necessary to have a long walk to the nearest crosswalk. So he decided to write a computer program to understand how to move the crosswalks along the Student Avenue in the most beneficial way for pedestrians. He wants as many crosswalks in front of NSU buildings as possible, at the same time the move of crosswalks should be minimal. Help Vasya to write a program, as he is also preparing an appeal to the mayor of the city and needs trustworthy calculations.

Here's how Vasya drew the plan of the avenue and NSU buildings before writing the program. The Student Avenue is represented by a straight line. Crosswalks are regarded as points on this line. All the buildings of NSU stand parallel to the avenue, so you can consider them to be segments on the line. Each of the segments has its left and right boundaries. A crosswalk is located in front of a building if the corresponding point on the Avenue is located between the left and the right borders of the building (including the boundary points). Since the crosswalks have been established in compliance with certain standards, Vasya decided to keep the distances between them, so he wants to move all the crosswalks by the same distance.

Input

In the first line of the input file there are two integers n, m, where n is the number of crossings on the Student Avenue, m is the number of NSU buildings ($1 \le n \le 10^4$, $1 \le m \le 10^3$).

In the second line there is a list of integers a_i ($1 \le i \le n$), specifying the coordinates of crosswalks on the avenue ($0 \le a_i \le 10^6$).

The third line contains m pairs of integers l_i, r_i ($1 \le i \le m$), each pair describes the coordinates of the boundaries of a building ($0 \le l_i < r_i \le 10^6$).

Output

In the output file print two nonnegative integers: the distance by which you should shift the crosswalks, and the number of crosswalks that will be in front of any building as a result of the shift. The second number (number of crosswalks) must be the maximum possible. If the answer is not uniquely determined, you have to additionally minimize the first number (the shift). Take into account that it is possible to move the crosswalks in any of the two directions.

Examples

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

Problem K. Hive

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 megabytes

There is a plane, which is tiled with regular hexagons. There are honeycombs on this plane. Worker bees at first misunderstood the project documentation, and now they have to turn beehive honeycombs around queen bee by 60 degree clockwise.

Beehive consists of hexagonal honeycombs, which are oriented in such manner that there are hexagon nodes below and above, and there are edges to the left and right which the honeycomb shares with its adjacent honeycombs in the row. Every consequent row is shifted relative to the previous row by half a honeycomb. The Ox axis goes from left to right along the horizontal row of honeycombs. The Oy axis is inclined 60 degrees relative to the Ox axis. The axes intersect at the honeycomb with coordinates $(0, 0)$. Example explanation contains illustrations showing the tiles numeration.

Input

In the first line of the input file there are three integers N, X and Y, where N is number of honeycombs in the hive (excluding the honeycomb of a queen bee), X and Y are coordinates of the honeycomb of a queen bee ($1 \le N \le 10^4; |X|, |Y| \le 10^4$). In each of the following N lines there is a pair of integers x and y, being the coordinates of a honeycomb of the hive ($|x|, |y| \le 10^4$). The coordinates of all honeycombs in the input file are different.

Output

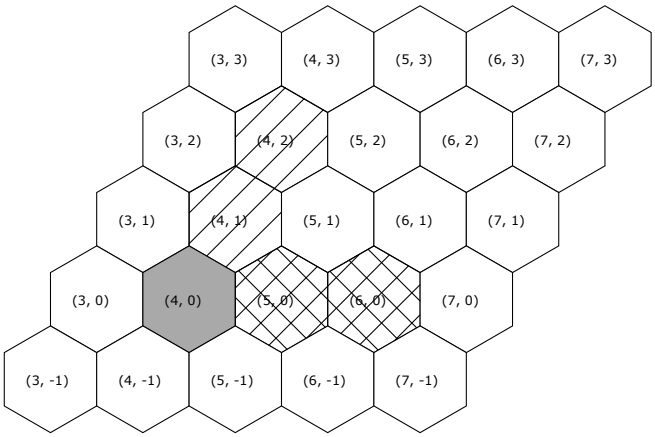
For each of N honeycombs, you should output two integers on the separate line: its coordinates after the turn. The coordinates must be ordered as they are in the input file.

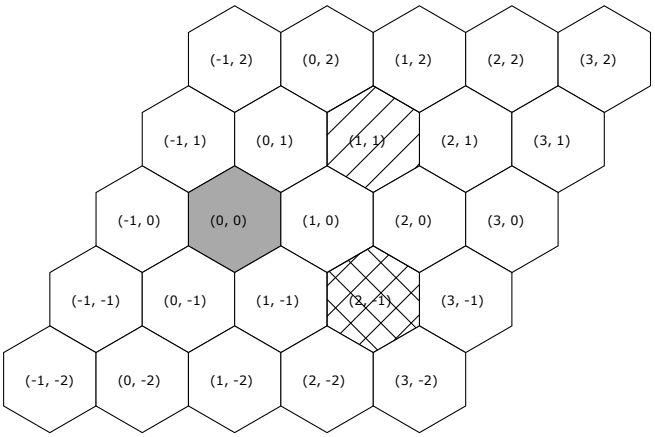
Example

standard input	standard output
2 4 0 4 1 4 2	5 0 6 0
1 0 0 1 1	2 -1

Example explanation

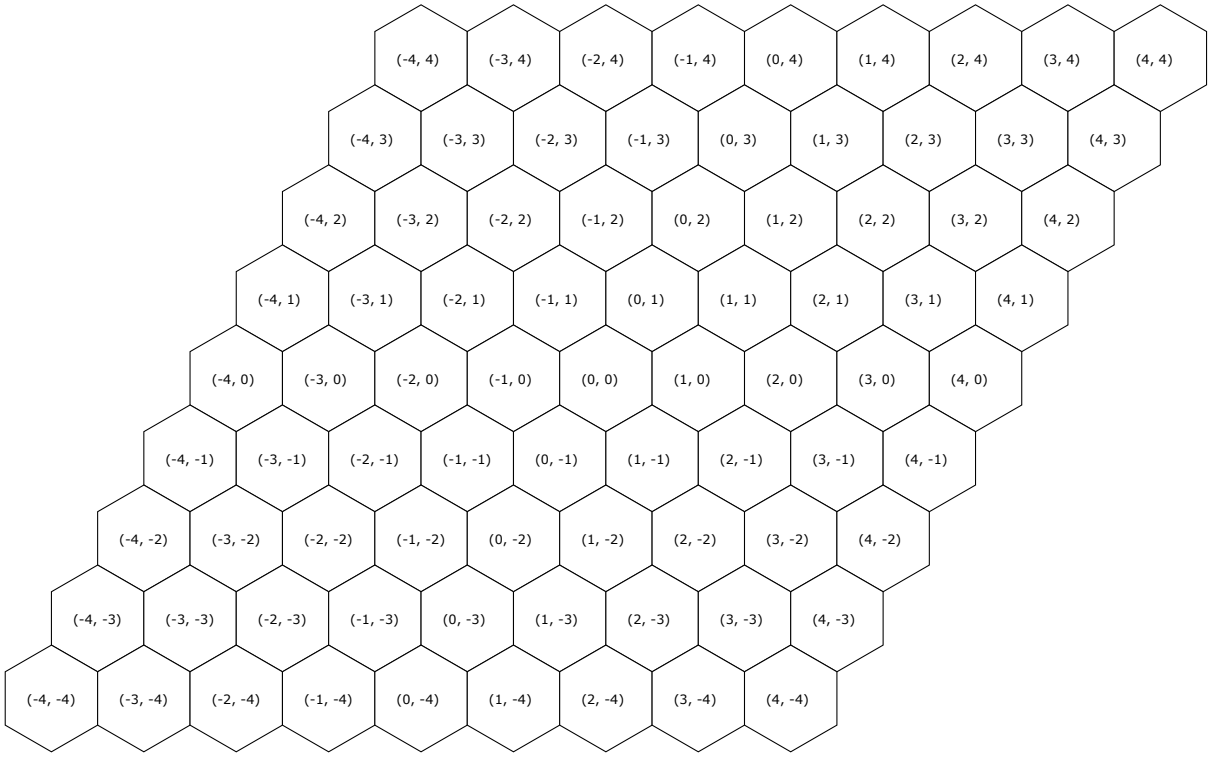
Hives from the examples are presented in the figures below. The honeycomb of the queen bee is marked in gray, honeycombs in their initial position are shaded with parallel lines, honeycombs after the turn are shaded with intersecting lines.





Illustration

One may draw on this hive. ;-)



Problem L. Side effects

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 megabytes

The programmer is developing an application. Being a part of a club "Young Friends of Functional Programming", he wants to know how many of its functions have side effects. Initially, for each function of the program it is known, whether it has a side effect or it has not; also, it is known that no function calls any other. Yet the programmer decides to change the logic of the application and starts to put some calls of the functions to other functions. The programmer does not write new functions. If a function calls a function with side effects, then the caller function also starts to have side effects (and so on in the chain of calls). Recursion in calls is allowed. Also, one function can call another several times. You need to determine how many functions with side effects are present in the program after each addition of a function call by the programmer.

Input

The first line of the input file contains three integers N, K and M, where N is total number of functions, K is initial number of functions with side effects, M is number of function calls added by the programmer ($1 \le N, K, M \le 10^5; K \le N$). Functions are numbered in order from 1 to N.

Next, there are K different numbers ranging from 1 to N in one line, being indices of functions which have side effects initially. In each of the following M lines there is a pair of integers a and b, which means that the programmer has added the call of function with the number b from the function number a ($1 \le a, b \le N$).

Output

For each of M additions of function calls, print the number of functions with side effects at the time after the addition of this call.

Example

standard input	standard output
3 1 5 3 1 1 2 3 3 2 2 1 3 1	1 2 2 2 2

Example explanation

The picture a) shows functions before adding calls. Pictures b) - f) show consecutive additions of function calls. The white circle with black outline is a function with no side effects, black circle is a function with side effects, the arrow shows a function call.

Firstly, the recursive function call is added to the function 1 from itself, from this action, obviously, the answer will not change. After adding a call to the function 3 from the function 2, the program

starts to have two functions with side effects: 2 and 3 because the function 3 initially had side effects. Subsequent additions of calls do not increase the number of functions with side effects.