Problem A. Amphitheatre

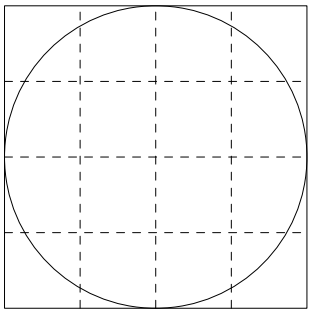
Input file:	amphitheatre.in
Output file:	amphitheatre.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Dig OK!

HoMM III

Andrew works for the RITArD (Research Institute of Toponymy, Archaeology and Digging). His group has recently discovered an ancient amphitheatre and now is preparing to start deeper research.

The digging area is a square with sides parallel to north-south and east-west directions. It was chosen to be as small as possible, however, to contain the whole amphitheatre (which has a form of a circle with radius r). The area was then split into regions with k straight lines going from north to south and l straight lines going from east to west. The distance between side of the square and the closest line parallel to it and the distances between neighboring lines are all equal.



The *excitement* of a research process in a region is equal to the area covered by the amphitheatre in that region. Andrew would like to maximize the excitement, so he asked you to write a program which can calculate the maximal possible excitement among all the regions.

Input

The only line of the input file contains three integer numbers k, l and r ($1 \le k, l, r \le 1000$).

Output

Write one real number: the maximal possible excitement among all given regions. Your answer will be considered correct if the relative or absolute error is within 10^{-6}.

Example

amphitheatre.in	amphitheatre.out
3 3 5	6.250000

Problem B. Brackets

Input file:	brackets.in
Output file:	brackets.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

> A bracket is an architectural member made of wood, stone, or metal that overhangs a wall to support or carry weight
>
> Wikipedia, the Free Encyclopedia

Vasya now works in RIBS — Research Institute of Bracket Sequences. In his current research project, he deals with regular bracket sequences of length $2N$.

One day Vasya noticed that his computer mixed up two of the bracket sequences he worked with. More precisely, the computer picked some positions and for each of them, it swapped the symbols in the first and second sequence in that position. For example, if the two regular bracket sequences were $A =$ `(())` and $B =$ `()()`, the computer could get four different pairs of sequences:

$A' =$ `(())`, $B' =$ `()()`
$A' =$ `(((`, $B' =$ `)))`
$A' =$ `)))`, $B' =$ `(((`
$A' =$ `()()`, $B' =$ `(())`

Here, in the first case, the computer made no swaps. In the second case, it swapped the brackets at position 3. In the third case, it swapped the brackets at position 2. Finally, in the last case, the computer swapped both the brackets at position 2 and the brackets at position 3, making a total of two swaps. Note that the resulting sequences A' and B' are not necessarily regular bracket sequences.

The problem is that Vasya does not remember what regular bracket sequences A and B looked like. So, for example, if he got $A' =$ `(((` and $B' =$ `)))`, the initial sequences could have been $A =$ `(())` and $B =$ `()()`, and the computer swapped the brackets at position 3; or they could have been $A =$ `()()` and $B =$ `(())`, and the computer swapped the brackets at position 2.

So, Vasya is willing to assume that the initial regular bracket sequences A and B were some sequences that required the minimal number of swaps to produce A' and B' from them. Help him find that number.

You are given two bracket sequences A' and B'. Find out the minimal number of swaps that computer could have made to produce these sequences from some regular bracket sequences A and B.

Input

The input contains two lines, each containing a string of symbols '`(`' and '`)`'. Length of each of these strings will be $2N$ ($1 \le N \le 1000$).

Output

Write only one integer — the minimal number of swaps. If computer could not get these sequences from any pair of requlars bracket sequences, write -1.

Examples

brackets.in	brackets.out
((( )))	1
(((( )))	-1

Problem C. Circle

Input file:	circle.in
Output file:	circle.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Round and round we go

A tutorial about avoiding non-integer arithmetics

RICTuS (Research Institute of Circular Transformations under Skies) has recently faced a very difficult problem. Of course, the problem is about a circle. On its border there are N cells numbered from 1 to N in clockwise order; two cells are adjacent if their numbers differ by one; cells 1 and N are also adjacent. Additionally there are N chips, also numbered from 1 to N. These chips are put on the circle (one chip on one cell) and the goal is to rearrange them in such a way that each chip would stand on a cell with its number. The main difficulty was the following: the RICTuS scientists didn't want to remove any chip from the circle during that rearrangement.

Recently scientists have understood that there was no solution for the initial problem. So they decided to add a central cell to the circle. This cell shouldn't be adjacent to all other, because it would make the problem too simple. After a long discussion this cell was considered to be adjacent only to 4 cells with numbers 1, $N/2$, $N/2+1$, N. Additionally, number N was considered to be even.

Now one can move any chip to an adjacent empty cell. This problem is quite new for RICTuS scientists, so they want you to find any solution of it.

Given the initial arrangement of chips, output the sequence of moves which will put every chip on a cell with its number.

Input

The first line in the input file contains one integer N ($4 \le N \le 100$, N is even). The second line contains N integers separated by spaces; i-th of them is the number of chip initially located on i-th cell.

Output

The output file should contain only one line. In it you should print the sequence of chip's numbers which you should move to achive the goal. There should not be more than 500 000 numbers in your output; otherwise, your solution will recieve "`Wrong Answer`" outcome.

Examples

circle.in	circle.out
4 1 2 4 3	3 4 3
4 4 3 2 1	4 1 4 3 2 3

Problem D. Computer

Input file:	computer.in
Output file:	computer.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Thing's positive

Specification of an unsigned type

RIPS (Research Institute of Probabilistic Systems) has recently invented and produced a new cutting edge computer. It is a fantastic machine — it has an infinite amount of memory and it processes all operations infinitely fast. It is clear that, even if it has some minor problems, its power is unparalleled. So, Vasya has to write programs for this computer.

However, the computer is in beta testing stage, so it still has several problems. More precisely, there are three of them.

1. The processor of this computer has just two operations: it can only add or multiply two integers stored in its memory, and write the result into memory. Adding or multiplying an integer stored in memory by itself is also allowed. No other operations and programming constructions such as cycles, conditions and even writing a constant to memory are possible.
2. Although operations are performed correctly, the process of writing integers to the memory has a bug: while writing, the number may change, but at most by 1. So, if you calculate $2 \cdot 3$, then the result will be 6, but the result stored in memory can be 5, 6 or 7.
3. The computer has no input or output devices. After boot, its memory contains just two positive integers x and y. These numbers are the same each time the computer boots.

The first problem that Vasya wants to solve is to store some integer z in the memory. Of course, Vasya understands that getting precisely z can be impossible. So, he wants to write a program that produces a result as close as possible to z in the worst case. The result of running a program is the last number stored by it in memory.

Input

Input file contains three integers x, y and z ($1 \le x, y, z \le 1000$, $z \ne x$, $z \ne y$).

Output

Write the distance to z that Vasya's program will get in the worst case.

Examples

computer.in	computer.out
2 3 6	1
2 3 7	2

Problem E. Guards in the Chess Kingdom (Division 1 Only!)

Input file:	guards.in
Output file:	guards.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

For great justice.

Zero Wing

The Kingdom of Chess consists of n cities, connected by m bidirectional roads. Cities are colored into 2 colors: black and white. Of course, there are no roads between cities of the same color.

Recently, crime rate increased in the Kingdom. Gentlemen of the road take away money from poor wayfarers. To avoid this, the King decided to ask for help of one of the most famous communities in the Kingdom — the Research Institute of Guards in the High Towers (RIGHT). They can, with the help of the King and Kingdom's treasury, place guard towers in some cities. A guard tower placed in a city reduces crime rate on all roads incident to this city. Of course, crime rate on *all* roads need to be reduced — so the King has to select cities for guard towers in such a way that for every road there is a guard tower in at least one of the two cities that are connected by this road. Placing guard tower is *very* costly, so the King has to place the minimal possible number of towers.

Additionaly, placing tower in a city reduces crime rate in this city as well. The King doesn't need to reduce crime rate in all cities, but loyal citizens are pleased when crime rate in their city is reduced. The King wants to maximize the total pleasure — that is, the total population of cities with towers should be maximal possible.

So, to help the King and all the Kingdom, you have to write a program that selects an optimal subset of cities to place guard towers in them. The number of cities in this subset should be minimal possible. In case of ambiguity, the total population of these cities should be maximal possible. In case of ambiguity, output any such subset.

Input

The first line of the input file contains two integers — n and m ($1 \le n \le 300$, $1 \le m \le 10\,000$). Here, n is the number of cities in the Kingdom and m is the number of roads. The second line contains n positive integers — population of cities 1, 2, ..., n, respectively. There is no city with population more than 100.

Next m lines contain description of roads. Line number $i + 2$ contains two integers — a_i and b_i. They mean that i-th road connect cities a_i and b_i. Cities are numbered from 1.

Output

The first line of the output file should contain two integers — P and K. Here, P is the total population of cities where towers should be placed and K is the number of such cities. Second and last line should contain K integers separated by spaces — the numbers of these cities in increasing order.

Example

guards.in	guards.out
4 4 1 2 3 4 1 2 1 4 2 3 3 4	6 2 2 4

Problem F. Integral (Division 1 Only!)

Input file:	integral.in
Output file:	integral.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

submit.exe?
head.think!

From "all you need to know about ACM ICPC"

RIPS (Research Institute of Probabilistic Systems) has recently invented and produced another cutting edge computer. It is a fantastic machine — it could compute any definite integral of a given polynomial infinitely fast. However, this computer works only with integer polynomial coefficients, and it gives the right answer only if the value of the definite integral is an integer.

$$I_i = \int_{u_i}^{v_i} \sum_{k=0}^{n} a_k x^k dx$$

For given T segments $[u_i, v_i]$, Vasya wants to know which of I_i will be computed correctly by computer.

Input

The first line of the input file contains two integes n and T ($1 \le n \le 30$, $1 \le T \le 100\,000$), where n is the degree of the polynomial. The second line contains $n + 1$ integer numbers a_0, a_1, ..., a_n ($|a_k| \le 10^9$) — the coefficients of the polynomial. Each of the next T lines contains two integers u_i and v_i ($|u_i| \le 10^9$, $|v_i| \le 10^9$) — the endpoints of i-th segment.

Output

For each segment, your program should output "Yes" if I_i is an integer and "No" otherwise.

Example

integral.in	integral.out
2 2 0 0 1 0 1 0 3	No Yes

Problem G. Minimal Path

Input file:	`minimal.in`
Output file:	`minimal.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

RIM (Research Institute of Minimality) needs you! They are investigating properties of minimal paths in a labyrinth. Recently, Vasya helped them in their research, but as he originates from RIGS, his work utilizes a string-based approach to minimality of paths.

More specifically, a *labyrinth* is presented as a two-dimensional map of $w \times h$ square cells. Each cell is either free or occupied by a wall; the whole labyrinth is considered to be surrounded by walls. One makes a *move* in the labyrinth from a given cell by choosing one of the four directions (down, left, right or up) and walking to the adjacent cell in that direction; both cells must be free to make a valid move. A *path* between two cells is a finite sequence of valid moves which starts in one cell and ends in the other; one is allowed to visit any cell more than once.

Now, Vasya has introduced some additional terms in this theory. A *string representation* of a path is a string of lowercase Latin letters '`d`', '`l`', '`r`' and '`u`' which describes the path. Each letter means a move from the current cell in the respective direction. Here, '`d`' stands for 'down' (increase row number by one), '`l`' for 'left' (decrease column number by one), '`r`' for 'right' (increase column number by one) and '`u`' for 'up' (decrease row number by one). A *minimal path* between two cells is the path which has the lexicographically smallest string representation.

Recall that a string $S = s_1 s_2 \dots s_p$ is said to be *lexicographically smaller* than a string $T = t_1 t_2 \dots t_q$ if either S is shorter than T and each symbol s_i is equal to t_i or for the smallest position i such that $s_i \neq t_i$, symbol s_i comes alphabetically earlier than t_i.

To fully understand Vasya's results, scientists from RIM need a program which will compute the string representation of a minimal path for any given labyrinth. Help them write that program.

Input

The first line of the input file contains two integers w and h separated by a single space ($1 \le w, h \le 10$). The second line of the input file contains four integers c_s, r_s, c_f and r_f separated by spaces. Here, (c_s, r_s) are the coordinates of the starting cell of the path and (c_f, r_f) are the coordinates of the final cell of the path ($1 \le c_s, c_f \le w$, $1 \le r_s, r_f \le h$). Finally, the next h lines contain w characters each; c-th character on r-th of them is '`.`' (dot) if the cell (c, r) is free or '`X`' (uppercase ex) if it is occupied by a wall. The starting cell and the final cell are different and are both free.

Output

If there is no minimal path between the two given cells, write "`None.`" on the first line of the output file. Otherwise, write the string representation of the minimal path.

Examples

minimal.in	minimal.out
2 2 2 1 1 2	dl
5 1 1 1 5 1 ..X..	None.
3 1 2 1 3 1 ...	None.

Problem H. Cutting a Pie

Input file:	pie.in
Output file:	pie.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

My pies! The toggles do nothing!

The cry of a cook who faced a non-working oven

Vasya worked hard for RIGS (Research Institute of Given Strings) for many years. But recently he understood that he should think about children. So he is going to get a job at RICH (Research Institute of Children's Happiness). He sent his curriculum vitae to the recruiting department, and they gave him a simple task to check his skills. Vasya isn't familiar with such problems yet, so he asked for your help.

The problem is about N children and only one pie. Each of these children has her (or his) own conception of a good piece of a pie. It can be formalized in the following way. Each girl (or boy) has two rational numbers a and b in her (or his) mind. She (or he) considers a piece of a pie of size s to be good if and only if $a \le s$ and $s \le b$.

The task is to divide a pie between children in such a way that each of them would think that her (or his) piece is good.

To cut a pie, the following sequence of actions should be performed:

1. Number all children by integers from 1 to N.
2. For all i, assign some integer k_i ($1 \le k_i \le K$) to a child with number i.
3. For all i, give to child with number i a piece of the pie with size $\frac{k_i}{\sum_{i=1}^{N} k_i}$.

Input

The first line of the input file contains two integers N ($1 \le N \le 300$) and K ($1 \le K \le 100$). Then follow N lines. The line $(i+1)$ contains two rational numbers a_i and b_i separated by exactly one space. Each of these numbers is given as a ratio of two integers. These integers lie between 0 and 10^9, inclusive. Second integer in each ratio isn't equal to 0.

Output

If there is no assignment which solves this problem, print "`No solution`". If there is a solution, print one line containing N integers — the numbers k_i which should be assigned to children. If there is more then one solution, you should choose the one with the smallest $\sum_{i=1}^{N} k_i$. If there is still ambiguity, compare sequences of numbers k_i lexicographically and choose the smallest one.

Recall that an array a_1, a_2, ..., a_p is said to be *lexicographically smaller* than an array b_1, b_2, ..., b_p if for the smallest i such that $a_i \ne b_i$, it is true that $a_i < b_i$.

Examples

pie.in	pie.out
1 2 0/1 1/2	No solution
3 2 0/1 1/1 0/1 1/1 0/1 1/4	1 2 1

Problem I. Pisces (Division 1 Only!)

Input file:	pisces.in
Output file:	pisces.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

It's not easy to find a black fish in a black room.
Especially if a black cat came there first.

man cat

Helge works for the RIEL (Research Institute of Extraterrestrial Life-forms). Her latest mission to the Pisces constellation was successful: n wonderful, active, colorful specimens of fish were obtained. There's a problem, however: the fishes of the same color are absolutely indistinguishable.

Now Helge wants to mark them with colored rings. She repeats the following steps until each pair of specimens becomes distinguishable:

- select some color that has not yet been used (not even as the natural color of some fish),
- get k random specimens from the aquarium (each subset of cardinality k is equally likely to be chosen),
- mark each of them with rings of selected color.

Each procedure (three steps) takes exactly one minute.

Two specimens are considered to be distinguishable if their colors differ or sets of colors of their rings differ.

The only thing Helge is unsure about now is the expected time before her task will be complete. So she asked you to write a program to compute this value.

Input

Input consists of two lines. First line contains two integers n and k ($1 \le k < n \le 30$). Second line contains n integers, i-th integer describes the initial color of i-th specimen. The integers which describe colors are positive and don't exceed 1000.

Output

Write one real number: the expected time (in minutes) before each pair of specimens becomes distinguishable. Your answer will be considered correct if the relative or absolute error is within 10^{-6}.

Examples

pisces.in	pisces.out
2 1 1 1	1.000000
2 1 30 239	0.000000

Problem J. Screening a String

Input file:	screening.in
Output file:	screening.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Islands on the screen!

The last words of the captain of the ship that was too fast...

As you know, Vasya works in RIGS. Today he studies a very strange process of screening a string. A screening of a string S is performed in the following way: somebody takes just only one of its first and second characters, one of third and fourth and so on, then concatenating them in the order they appeared in the original string. If a string has an odd length, the last character is always appended to the resulting string, otherwise the process finishes on choosing of exactly one of $(N-1)$-th and N-th character. For example, after screening the string `abacaba` one may obtain `aaaa` or `baba`, but it's impossible to obtain `baca` or `acb`.

The current goal is, given a string, to determine how many screenings are required to obtain a string formed of occurences of only one character. Your task is to write a program for that.

Input

The only one line of the input contains just one non-empty line of no more than 500 000 lowercase English letters — the given string.

Output

You have to write to the output the required number of screenings.

Examples

screening.in	screening.out
abacaba	1
littlekitten	3

Problem K. Tree

Input file:	tree.in
Output file:	tree.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

> You're thinking in Japanese aren't you? If you *must* think, do it in German!
>
> Neon Genesis Evangelion

After an epic fail of several NERV projects, Rei decided to join the RINGLET (Research Institute of Neon Genesis Lexicographically Enormous Trees). Now she is studying properties of nets consisting of genetically modified trees.

A *net* is a set of trees connected with several bio-cords. Each cord connects exactly two trees and no cord connects a tree to itself.

A *subnet* is a subset of cords. It is called stable if

- every tree can be reached from every other tree using one or more cords from it and
- it contains no proper subset with the property above.

From time to time one needs to compare two subnets lexicographically. It should be done simply writing down indices of cords used in each of them in ascending order and then comparing resulting arrays lexicographically.

Recall that an array $a_1, a_2, \ldots, a_p$ is said to be *lexicographically smaller* than an array $b_1, b_2, \ldots, b_p$ if for the smallest i such that $a_i \neq b_i$, it is true that $a_i < b_i$.

Researchers in RINGLET invented several methods of finding a subnet with the smallest total length of cords. However, most subnets that can be obtained with their methods are lexicographically large, so new Rei's assignment is to find the lexicographically smallest subnet among all the stable subnets which have the smallest total length of cords. She is sure she can do it in several minutes, but what about you?

Input

The first line of input file contains two integer numbers n and m ($2 \le n \le 100\,000$, $m \le 100\,000$) — the number of trees and cords between them. Then m lines follow, each containing three integer numbers — indices of trees connected by corresponding cord and the cord's length. The length of each cord is non-negative and does not exceed 100 000.

Output

Write $n - 1$ numbers to the output file — the numbers of edges of the lexicographically smallest stable subnet. The edges are numbered starting from 1 in the order they are given in the input file. It is guaranteed that the answer exists.

Example

tree.in	tree.out
4 4 1 2 1 2 3 1 3 4 1 4 1 1	1 2 3

Problem L. Funny Game (Division 2 Only!)

Input file:	`funnygame.in`
Output file:	`funnygame.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Two players are playing in the following game. Initially they have a play board of $m \times m$ square cells and a coin placed in the (x, y) board cell. Each player moves a coin on a neighbour cell in a move (two cells is neigbours if they have only one common side). No one player can move a coin out of the play board or into a cell, where it have already been. A player which cannot make a move loose.

Your task is to determine which player wins a game, if both of them play optimally.

Input

Input file contains a multitest. The first line contains a number of testcases, $N \le 400$.

Each of the following N lines contains 3 numbers: m_i, x_i, y_i ($1 \le m_i \le 500$, $1 \le x_i, y_i \le m_i$).

Output

Output file should contain N lines, each of them containing "`First`" if first player wins and "`Second`" if second player wins.

Example

funnygame.in	funnygame.out
2 1 1 1 2 2 2	Second First

Problem M. Sacrament Sum (Division 2 Only!)

Input file:	sum.in
Output file:	sum.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Lets define a_n as an integer number, nearest to the square root from n.

Your task is to find $S(m) = \sum_{i=1}^{m} \frac{1}{a_i}$ for given m.

Input

Input file contains a multitest. The first line contain a number of testcases, $N \leq 10^5$.

Each of the following N lines contains a number m_i ($1 \leq m_i \leq 10^{15}$).

Output

Output file should contain N lines, each of them containing value of $S(m_i)$ with 4 digits after decimal point.

Example

sum.in	sum.out
2 1 2	1.0000 2.0000

Problem N. Sophie Germain Primes (Division 2 Only!)

Input file:	sgprime.in
Output file:	sgprime.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

A *prime number* is a positive integer that can only be evenly divided by itself and by 1.

A *Sophie Germain prime* is a prime number p such that the number $2p + 1$ is also prime.

Write a program to list the Sophie Germain primes in a given range.

Input

Input file contains a multitest. The first line contain a number of testcases, $N \le 20$.

Each testcase consists of two integers L and H ($0 \le L \le H \le 10^5$) on one line — first and last number in range, respectively. End of the datasets is signaled by a negative value for L or H.

Output

For each dataset, there will be a single line of output. Print the values of L and H, separated by a single space. Then print a colon and a space. After that, print a list of all Sophie Germain primes p_i, $L \le p_i \le H$. Consecutive prime numbers in the output should be separated by a single space, with no space following the last prime.

Example

sgprime.in	sgprime.out
2 0 4 100 130	0 4: 2 3 100 130: 113