Problem A. Digits and Bases

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 секунда
Memory limit:	256 мебибайт

Calculate number of positive integers with next property: their representation in the system with base B_1 consists of D_1 digits, and their representation in system with base B_2 consists of D_2 digits. Leading zeroes are not allowed.

Input

First line of the input contains two integers B_1 and D_1 ($2 \le B_1 \le 100$ $1 \le D_1 \le 20$). Second line contains two integers B_1 and D_1 ($2 \le B_1 \le 100$, $1 \le D_1 \le 20$). Third line contains two integers B_2 and D_2 ($2 \le B_2 \le 100$, $1 \le D_2 \le 20$).

You may assume that maximal D_1-digit B_1-base integer and maximal D_2 digit B_2 base integer do not exceed 10^{18}.

Output

Print one integer — answer to the problem.

Examples

standard input	standard output
3 1 5 1	2
10 2 2 4	6

Note

In first sample the task is to calculate how much numbers have 1-digit representation for bases 3 and 5. There are 2 of those numbers: 1 and 2 (note that we need positive integers, so zero can't be a solution). In the second sample the task is to calculate two-digit decimal integers which are 4-digit in the binary system. There are 6 of them: between 10_{10} (1010_2) and 15_{10} (1111_2) inclusively.

Problem B. Balloons

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Bill defines an *balloon* of radius R and center O in N-dimensional space as set of all points with next property: Euclidean distance between point from balloon and O is equal to R.

Given two N-dimensional balloons with integer coordinates of O and integer radii. Check how much common points they have.

Input

First line of the input contains one integer N — number of dimensions of the space ($2 \le N \le 100$). Second line contains parameters of first balloon: radius R and coordinates of center $x_1, x_2, \ldots, x_N$ ($1 \le R \le 10^5$, $-10^5 \le x_i \le 10^5$). Third line describes the second balloon in the same form.

Output

Print one integer — number of common points for given balloons of -1, if number of common points is infinite.

Examples

standard input	standard output
3 2 0 0 0 1 3 0 0	1
3 2 0 0 0 1 1 1 1	-1
3 2 0 0 0 2 0 1 8	0

Problem C. ICPC History

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Till 1992 ICPC was a competition for the teams of **four** students. And at the 1991-1992 season the new rule was implemented and the number of contestants in team was reduced to current value of 3.

Giga, the coach of teams of some university in Georgia, US, is planning for the season 1991-1992. At the previous season K complete teams were attending the training sessions; L contestants are not eligible for ICPC for season 1991-1992 and M new students joined the ICPC community at the beginning of year.

Giga wants to know minimum number of additional students needed to join the community ti create full 3-member teams for all current members of ICPC community such as one contestant is enlisted in exactly one team.

Input

The input contains three integers K, L and M ($1 \le K \le 100$, $0 \le L \le 4 \cdot K$, $1 \le M \le 100$).

Output

Print one integer — minimum number of students to join the ICPC community, so Giga may form full teams for all current members of ICPC community.

Example

standard input	standard output
3 2 1	1
20 1 8	0

Problem D. Yellow Code

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

Inspired by Gray code, professor John Yellow has invented his own code. Unlike in Gray code, in Yellow code the adjacent words have many different bits.

More precisely, for $s = 2^n$ we call the permutation $a_1, a_2, \ldots, a_s$ of all n-bit binary words the *Yellow code* if for all $1 \le k < s$ words a_k and a_{k+1} have at least $\lfloor n/2 \rfloor$ different bits and a_1 and a_s also have at least $\lfloor n/2 \rfloor$ different bits.

Given n you have to find the n-bit Yellow code or detect that there is none.

Input

Input file contains the number n ($2 \le n \le 12$).

Output

Output 2^n n-bit binary vectors in the order they appear in some n-bit Yellow code, one on a line. If there is no n-bit Yellow code, output "`none`" on the first line of the output file.

Example

standart input	*standard output*
4	0000 1111 0001 1110 0010 1101 0011 1100 0101 1011 0100 1010 0110 1000 0111 1001

Problem E. Human Knot

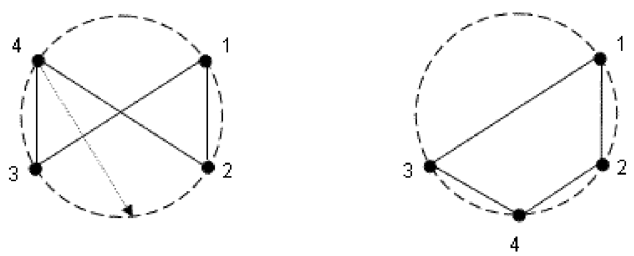
Input file:	*standart input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	64 megabytes

A classic ice-breaking exercise is for a group of n people to form a circle and then arbitrarily join hands with one another. This forms a "human knot" since the players' arms are most likely intertwined. The goal is then to unwind the knot to form a circle of players with no arms crossed.

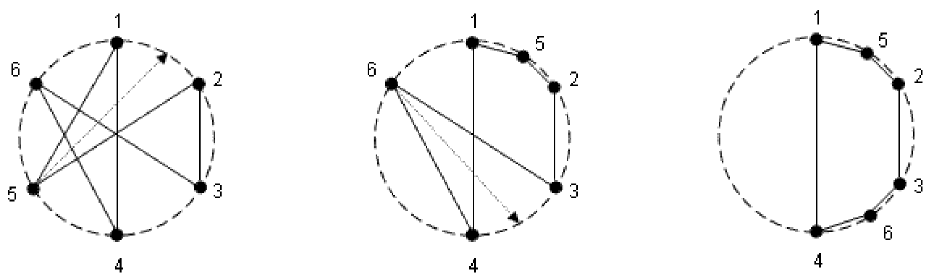
We now adapt this game to a more general and more abstract setting where the physical constraints of the problem are gone. Suppose we represent the initial knot with a 2-regular graph inscribed in a circle (i.e., we have a graph with n vertices with exactly two edges incident on each vertex). Initially, some edges may cross other edges and this is undesirable. This is the "knot" we wish to unwind.

A "move" involves moving any vertex to a new position on the circle, keeping its edges intact. Our goal is to make the fewest possible moves such that we obtain one n-sided polygon with no edge-crossings remaining.

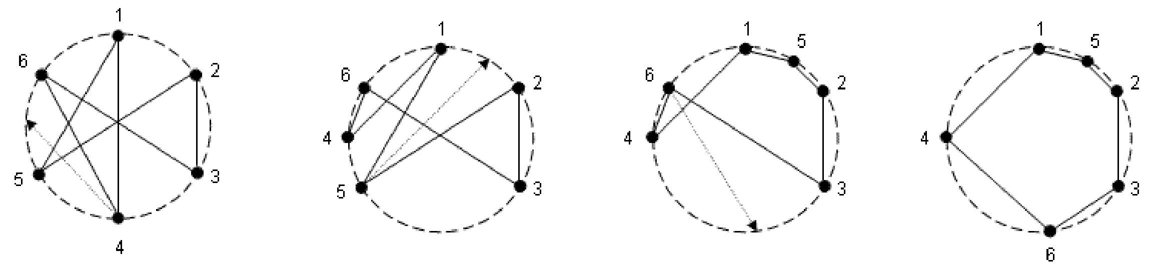
For example, here is a knot on 4 vertices inscribed in a circle, but two edges cross each other. By moving vertex 4 down to the position between 2 and 3, a graph without edge-crossings emerges. This was achieved in a single move, which is clearly optimal in this case.



When n is larger, things may not be quite as clear. Below we see a knot on 6 vertices. We might consider moving vertex 4 between 5 and 6, then vertex 5 between 1 and 2, and finally vertex 6 between 3 and 4; this unwinds the knot in 3 moves.



But clearly we can unwind the same knot in only two moves:



Input

The input consists of a number of cases. Each case starts with a line containing the integer n ($3 \le n \le 500$), giving the number of vertices of the graph. The vertices are labelled clockwise from 1 to n. Each of the next n lines gives a pair of neighbors, where line i ($1 \le i \le n$) specifies the two vertices adjacent to vertex i. The input is terminated by $n = 0$.

Output

For each case, if there is no solution, print "`Not solvable.`" on a line by itself. If there is a solution, print "`Knot solvable.`" on a line by itself, followed by the minimum number of moves required to solve the problem, on a line by itself.

Example

standart input	*standard output*
6	Knot solvable.
4 5	2
3 5	Knot solvable.
2 6	1
1 6	
1 2	
3 4	
6	
2 6	
1 4	
5 6	
2 5	
3 4	
1 3	
0	

Problem F. Margaritas on the River Walk

Input file: *standart input*
Output file: *standard output*

One of the more popular activities in San Antonio is to enjoy margaritas in the park along the river know as the *River Walk*. Margaritas may be purchased at many establishments along the River Walk from fancy hotels to *Joe's Taco and Margarita stand*. (The problem is not to find out how Joe got a liquor license. That involves Texas politics and thus is much too difficult for an ACM contest problem.) The prices of the margaritas vary depending on the amount and quality of the ingredients and the ambience of the establishment. You have allocated a certain amount of money to sampling different margaritas.

Given the price of a single margarita (including applicable taxes and gratuities) at each of the various establishments and the amount allocated to sampling the margaritas, find out how many different maximal combinations, choosing at most one margarita from each establishment, you can purchase. A valid combination must have a total price no more than the allocated amount and the unused amount (*allocated amount* − *total price*) must be less than the price of any establishment that was not selected. (Otherwise you could add that establishment to the combination.)

For example, suppose you have $25 to spend and the prices (whole dollar amounts) are:

Vendor	A	B	C	D	H	J
Price	8	9	8	7	16	5

Then possible combinations (with their prices) are: ABC(25), ABD(24), ABJ(22), ACD(23), ACJ(21), ADJ(20), AH(24), BCD(24), BCJ(22), BDJ(21), BH(25), CDJ(20), CH(24), DH(23) and HJ(21).

Thus the total number of combinations is 15.

Input

The input begins with a line containing an integer value specifying the number of datasets that follow, N, ($1 \le N \le 1\,000$). Each dataset starts with a line containing two integer values V and D representing the number of vendors ($1 \le V \le 30$) and the dollar amount to spend ($1 \le D \le 1\,000$) respectively. The two values will be separated by one or more spaces. The remainder of each dataset consists of one or more lines, each containing one or more integer values representing the cost of a margarita for each vendor. There will be a total of V cost values specified. The cost of a margarita is always at least one (1). Input values will be chosen so the result will fit in a 32 bit unsigned integer.

Output

For each problem instance, the output will be a single line containing the dataset number, followed by a single space and then the number of combinations for that problem instance.

Note: Some solution methods for this problem may be exponential in the number of vendors. For these methods, the time limit may be exceeded on problem instances with a large number of vendors such as the second example below.

Example

standart input	*standard output*
2 6 25 8 9 8 7 16 5 30 250 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30	1 15 2 16509438

Problem G. Angle Beats

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

You have a $n \times m$ field, consisting of '*', '+' and '.'.

You can draw connected polyominoes, consisting of three tiles. Each figure should have center at some '*' or '+' and all other cells on it should be '.'.

Also if some tile contain '*', it may be contained only in the "Angles", i.e. in the figures where one neighbour of the center is vertical and other is horizontal.

Your goal is to draw maximum number of non-intersecting polyominoes.

Input

First line contains two positive integers n and m — number of rows and columns in the table you are given ($2 \le n, m \le 100$). Then, n lines describing the table follow. Each line contains exactly m characters '*', '+', '.'.

Output

Output n lines, m characters each. If the cell belong to some figure, output some lowercase latin character. Of course, all cells of the figure should have the same character in output. Also two adjacent figures should have different characters.

Examples

standard input	standard output
2 2 *. ..	aa a.
3 3 ... .*. ...	.a. aa. ...
5 5 +*..+ ..++. .+.++ .**.+ .+*.+	+*baa .bb+a ccc++ .**.+ .+*.+
11 13 ....+++....+++.. ....+..+..... ....+..+..... ....+..+..... .+++....+++.. ....+......+. ....+......+. ....+......+. .+++....+++..	.abc....abc.. aabcc..aabcc. ..baaacccb... ...dddeee.... .abfffgggbc.. aab+a..aabcc. ..baa...abaaa ...ccc....ddd .abddd..abccc aabee..aabee. ..be.....be..

Problem H. Best Subsequence

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

You have an array $w_1, w_2, \ldots, w_n$ of length n.

You need to choose subsequence of k elements $1 \le i_1 < i_2 < \ldots < i_k \le n$.

Your goal is to find minimal $\max((w_{i_1} + w_{i_2}), (w_{i_2} + w_{i_3}), \ldots, (w_{i_{k-1}} + w_{i_k}), (w_{i_k} + w_{i_1}))$ among all such subsequences.

Input

The first line of input contains two integers n, k — the number of elements in the array w and the length of subsequence ($3 \le k \le n \le 200\,000$).

The second line contains n space separated integers $w_1, w_2, \ldots, w_n$ ($1 \le w_i \le 10^9$).

Output

Print one integer — minimal $\max((w_{i_1} + w_{i_2}), (w_{i_2} + w_{i_3}), \ldots, (w_{i_{k-1}} + w_{i_k}), (w_{i_k} + w_{i_1}))$ among all subsequences of length k.

Example

standard input	standard output
5 3 17 18 17 30 35	35

Problem I. Cool pairs

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

You have two permutations $p_1, p_2, \ldots, p_n$ and $q_1, q_2, \ldots, q_n$ and one integer k.

You need to find two integer arrays, a and b, such that $a_{p_1} \le a_{p_2} \le \ldots \le a_{p_n}$, $b_{q_1} \le b_{q_2} \le \ldots \le b_{q_n}$.

Let's call pair (i, j) **cool** if $i < j$ and $a_i + b_j < 0$.

Can you find a and b, such that the number of cool pairs will be equal to k?

Input

The first line of the input contains two integers n, k — length of the arrays and the required number of cool pairs $(1 \le n \le 300\,000)$, $(0 \le k \le \frac{n \cdot (n-1)}{2})$.

Next two lines contain the permutations, first of them contain the permutation $p_1, p_2, \ldots, p_n$ other one contain the permutation $q_1, q_2, \ldots, q_n$.

Output

If there are no pair of integer arrays, such that the number of cool pairs is equal to k, print «`No`».

Otherwise, print «`Yes`», and print a and b in next two lines, $(-n \le a_i, b_i \le n)$.

Example

standard input	standard output
5 3 3 5 1 2 4 1 2 3 4 5	Yes 2 3 -1 5 1 -5 -3 -2 -2 0

Problem J. Dates

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Ilya loves dating!

He is planning his dates on the next t days (these days are numbered from 1 to t).

He knows that at day x he can plan at most a_x dates.

Also, he has n known girls numbered from 1 to n, he can go on a date with girl i at any day in $[l_i, r_i]$, and he will get p_i pleasure for it. Of course, he can't date one girl more than once.

An additional constraint on girls is also guaranteed: if $i < n$, then $l_i \le l_{i+1}, r_i \le r_{i+1}$.

His goal is to plan dates with some girls to maximize total pleasure.

Help him — find maximum total pleasure he can get if he will properly choose the girls and the days of dates for them.

Input

The first line contains two integers n, t — the number of girls and the number of days ($1 \le n, t \le 300\,000$).

The next line contains t integers $a_1, a_2, \ldots a_t$,

a_i means the maximum number of dates Ilya can plan at day i ($0 \le a_i \le n$).

The next n lines contain the descriptions of the girls.

i-th of them contains three integers l_i, r_i, p_i — the borders of the i-th girl's days segment, and the pleasure Ilya will get for dating her ($1 \le l_i \le r_i \le t$, $1 \le p_i \le 10^9$).

It is guaranteed that if $i < n$, then $l_i \le l_{i+1}, r_i \le r_{i+1}$.

Output

Output one number — maximum total pleasure Ilya can get if he will properly choose the girls and the days of dates for them.

Example

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

Problem K. Expected Value

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

You are given the connected plane graph on n vertices. At first, you are standing at vertex 1. Then, each second you are going to some vertex, adjacent to the current vertex, this vertex is chosen randomly with equiprobability among all adjacent vertices. Your task is to calculate the expected value of the time when you will get to vertex n.

Input

The first line of input contains one integer n — the number of vertices in the given plane graph ($2 \le n \le 2000$).

Next n lines contain the description of the points, where vertices are located. i-th of them contains two integers x_i, y_i ($0 \le x_i, y_i \le 5000$), it is guaranteed that there are no two equal points.

The next line of input contains one integer m — the number of edges in the given plane graph ($n - 1 \le m \le 5000$).

The next m lines describe edges in the graph. The i-th of these lines contains two integers a_i and b_i ($1 \le a_i, b_i \le n$; $a_i \ne b_i$), denoting an edge between vertices a_i and b_i, and the segment between corresponding points. It is guaranteed that no two of the given segments are intersecting (they can intersect only at the common ends), and there are no multiple edges, and the graph is connected.

Output

It is guaranteed, that in the given tests required expected value can be represented as an irreducible fraction $\frac{P}{Q}$, where $P, Q > 0$. Then, you need to output $(P \cdot Q^{-1})$ modulo 998 244 353.

Examples

standard input	standard output
2 0 0 35 35 1 1 2	1
6 0 0 1 1 2 4 3 9 4 16 5 25 8 1 2 2 3 2 4 3 4 4 5 5 6 1 6 2 6	798595486