Problem A. Area

Input file:	`area.ln`
Output file:	`area.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Calculate the area of a polygon.

Input

First line contains integer number n ($3 \le n \le 50000$) — number of vertices. n lines follow, containing two integer numbers x and y each ($-10000 \le x, y \le 10000$) — coordinates of vertices.

The vertices indeed form a polygon, i.e. the edges neither cross, nor touch each other.

Vertices are given in counterclockwise order.

Output

Output the area of the polygon with six hundred digits after decimal point.

Example

area.ln	area.out

3
0 0
1 0
0 1
	0.5000000000 ... 000

Output file contains six hundred zeroes.

Problem B. Bipartite Matching

Input file:	`pairs.in`
Output file:	`pairs.out`
Time limit:	1 second
Memory limit:	64 megabytes

A *bipartite graph* is a graph $(V, E), E \subset V \times V$ such that a set of vertices V can be partitioned into two sets A and B in such a way that $\forall (e_1, e_2) \in E \ e_1 \in A, e_2 \in B$ and $A, B \subset E, A \cap B = \varnothing$.

A *matching* of a bipartite graph is a set of its non-adjacent edges, that is, a set $S \subset E$ such that for any two edges $e_1 = (u_1, v_1), e_2 = (u_2, v_2)$ in S $u_1 \neq u_2$ and $v_1 \neq v_2$.

Your task is to find a maximal matching of a bipartite graph, that is the matching with the maximal number of edges.

Input

First line contains two integer numbers n and m ($1 \leq n, m \leq 250$). n and m are numbers of vertices in A and B, respectively.

n lines follow with description of edges. i-th vertex of A is described in $i+1$-th line. Each line contains numbers of vertices of B connected with i-th vertex of A. The numbering of vertices in A and B is independent. Each list is terminated by a single zero.

Output

First line of the output file should contain one integer number l — the number of elements in a maximal matching. l lines should follow, each containing two integer numbers u_j, v_j — the edges forming a matching.

Example

pairs.in	pairs.out

2 2
1 2 0
2 0
	2
	1 1
	2 2

Problem C. Coincidence

Input file:	`maxcon.im`
Output file:	`maxcom.out`
Time limit:	1 second
Memory limit:	64 megabytes

Common subsequence of two strings s_1 and s_2 is a pair of sequences of indices $(\{a_i\}, \{b_i\})$ such that $a_1 < a_2 < \ldots < a_k$, $b_1 < b_2 < \ldots < b_k$, and $s_1[a_i] = s_2[b_i]$ for all $1 \le i \le k$.

Find a longest common subsequence of two strings.

Input

First and second line of an input file contain two strings of French lowercase characters `a` … `z`. There are no spaces before, inside or after the strings. Lengths of strings do not exceed 100.

Output

In the first line of output file output k – the length of a longest common subsequence. On the second line output k numbers – indices of a common subsequence in the first input string. On the third line output the same for the second input string. Index of the first character in the string is 1. Indices should be output in ascending order.

Example

maxcon.im	maxcom.out

abcd
cxbydz
	2
	3 4
	1 5

Problem D. Day of Week

Input file:	`dayofw.in`
Output file:	`dayofw.out`
Time limit:	1 second
Memory limit:	64 megabytes

We now use the Gregorian style of dating in Russia. The leap years are years with number divisible by 4 but not divisible by 100, or divisible by 400.

For example, years 2004, 2180 and 2400 are leap. Years 2001, 2181 and 2300 are not leap.

You task is to write a program which will compute the day of week corresponding to a given date in the nearest past or in the future using today's agreement about dating.

Input

The input file consists of one or more test cases. Each test case is located in a single line. This line contains the day number d, month name M and year number y ($1980 \le y \le 10^{300}$). The month name is the corresponding English name starting from the capital letter. Some extra spaces and/or line feeds may follow the last case.

Output

For each test case, output a single line with the English name of the day of week corresponding to the date, statring from the capital letter. All other letters must be in lower case.

Example

dayofw.in	dayofw.out
9 October 2001 14 October 2001	Tuesday Sunday

Problem E. Number Partitions

Input file:	`part.in`
Output file:	`part.out`
Time limit:	1 second
Memory limit:	64 megabytes

A *partition* of a number n is a set of integer positive numbers $d_1 \geq d_2 \geq \ldots \geq d_k$ such that

$$\sum_{i=1}^{k} d_i = n$$

Generate all partitions of a given number in anti-lexicographical order.

Input

Input number contains one integer positive number n, $1 \leq n \leq 30$.

Output

Output file should contain all partitions of a number n, one partition at a line, in anti-lexicographical order.

Example

part.in	part.out
3	3 2+1 1+1+1

Problem F. Fibonacci Strings

Input file:	`fib.in`
Output file:	`fib.out`
Time limit:	1 second
Memory limit:	64 megabytes

A *Fibonacci string* is a string of `0`s and `1`s that does not contain consecutive `1`s.

Output a Fibonacci string by its position in the lexicographically ordered set of all Fibonacci strings of the same length.

Input

The input file contains two integers n and k; n is the length of the Fibonacci string ($1 \leq n \leq 44$), and k is the postion of the string to be displayed (valid $k \geq 1$).

Output

Output n-th Fibonacci string in the only line of output.

Example

`fib.in`	`fib.out`
`3 3`	`010`

Problem G. Generation of a Permutation

Input file:	`gen.in`
Output file:	`gen.out`
Time limit:	1 second
Memory limit:	64 megabytes

Generate a permutation of $1 \le N \le 100$ elements by its lexicographical number $1 \le K \le N!$.

Input

The only line of input contains N and K separated by an arbitrary number of whitespace characters.

Output

In the only line of output file print N different integers $x_1, \ldots, x_N$ ranging from one to N, separated by single spaces. No trailing spaces are allowed.

Example

`gen.in`	`gen.out`
`3 2`	`1 3 2`

Problem H. (p, q)-Horse

Input file:	`horse.in`
Output file:	`horse.out`
Time limit:	1 second
Memory limit:	64 megabytes

(p, q)-horse is a generalisation of chess Knight who moves p steps one direction and q steps another (perpendicular) direction. Ordinary chess Knight is thus a $(2, 1)$-horse.

Your task is to determine how many moves (p, q)-horse needs to go from one cell on $M \times N$ chess-board to another.

Input

One and only line in the input file contains 8 integer numbers M, N, p, q, x_1, y_1, x_2, y_2 ($1 \le x_1, x_2 \le M$, $1 \le y_1, y_2 \le N$, $0 \le p \le M \le 100$, $0 \le q \le N \le 100$).

Output

First line of the output file must contain an integer number k – number of moves that (p, q)-horse needs to move from cell (x_1, y_1) to the cell (x_2, y_2). $k+1$ lines must follow, containing sequential positions of (p, q)-horse on the way.

If (p, q)-horse cannot reach (x_2, y_2) from (x_1, y_1), output `-1`.

Example

horse.in	horse.out

`3 3 1 1 1 1 3 3`	`2`
	`1 1`
	`2 2`
	`3 3`

`2 2 1 1 1 1 1 2`	`-1`

Problem I. Reversive Inversions

Input file:	`invers.in`
Output file:	`invers.out`
Time limit:	1 second
Memory limit:	64 megabytes

Inversion table for a permutation P of numbers $\{1, 2, \ldots, N\}$ is the table $A = (A_i)_{1 \le i \le N}$ which maps each $i = P_j$ into the number of indices j' such that $j' \le j$ but $P_{j'} > P_j = i$.

Given an inversion table for a permutation P, calculate the inversion table for the inverse permutation P^{-1}.

Input

File consists only of N integer numbers, delimited by spaces and newline characters, that form the inversion table of a permutation. You may assume that $1 \le N \le 5000$.

Output

Output N integer numbers separated by single spaces — inversion table for the inverse permutation. Leave no trailing spaces at the end of the single line of output.

If there are several possible answers, output any of them. If there are no answers, output the first N primes instead.

Example

invers.in	invers.out
`5 0 1 3 2 1 0`	`1 5 1 3 2 0 0`

Problem J. Joseph Problem

Input file:	`joseph.in`
Output file:	`joseph.out`
Time limit:	1 second
Memory limit:	64 megabytes

n boys are standing in circle. They start counting themselves clockwise, starting from 1. As soon as the count reaches p, the last boy counted leaves the circle, and they continue counting from the next boy, starting from 1 again.

Last remaining boy wins.

Can you calculate his number in clockwise order, if the boy from whom the counting originally started has number 1?

Input

Input file contains two integer numbers, n and p. ($1 \le n, p \le 1000000$).

Output

Output file should contain one number — the original number of the last boy.

Example

`joseph.in`	`joseph.out`
`3 4`	`2`

Problem K. Combinations-2

Input file:	`combo2.in`
Output file:	`combo2.ovt`
Time limit:	1 seconds
Memory limit:	64 megabytes

A *combination* of k elements out of n is an increasing sequence of k integer numbers in range from 1 to n.

Generate all combinations of k elements out of n in any order such that any two adjacent combinations have no more than one difference (that is, if S_1 and S_2 are two adjacent combinations, then $\#(S_1 - S_2) \leq 1$).

Input

Input file contains two integer numbers n and k such that $1 \leq k \leq n \leq 15$.

Output

Output $\binom{n}{k}$ lines – all combinations of k elements out of n in any order satisfying the conditions above.

Example

combo2.in	combo2.ovt

3 2	1 2
	1 3
	2 3

Problem L. Average Length

Input file:	`average.in`
Output file:	`average.out`
Time limit:	1 second
Memory limit:	64 megabytes

You are given a description of a working road network. Your task is to find average length of the shortest path between two towns in this network.

The average length is a sum (for all the pairs (a, b) where the shortest path exists from a to b and has length l_i) of all the l_i, divided by a number of such pairs. a and b in this definition are different integer numbers less or equal to n — the total number of towns.

Input

The first two numbers in the input are n and k ($1 \le n \le 100$, $1 \le k \le n(n-1)$). The next k lines contain three integers each (a_i b_i L_i, $1 \le a_i, b_i \le n$, $1 \le L_i \le 1000$) representing a one-way road from a_i to b_i of length L_i.

Output

You are to display only one number rounded to 6 digits after decimal point — average length of the shortest path.

Example

average.in	average.out

6 4
1 2 7
3 4 8
4 5 1
4 3 100
	25.000000

Problem M. Common Measure

Input file:	`common.in`
Output file:	`common.out`
Time limit:	1 second
Memory limit:	64 megabytes

The *common measure* of two segments is length of maximal segment such that any of the two segments can be obtained by repeating it several times on a straight line. E. g. the common measure of segments with lengths $\sqrt{50}$ and $\sqrt{8}$ is $\sqrt{2}$.

Of course, not all pairs of segments have common measure. For example, the side and the diagonal of a square are incommensurable.

You have to check whether two given segments have a common measure and to find it in the case of positive answer.

Input

Input file contains eight integers, all of them not exceeding 10^4 by an absolute value: $x_{(1,\text{source})}$, $y_{(1,\text{source})}$, $x_{(1,\text{but})}$, $y_{(1,\text{but})}$, $x_{(2,\text{source})}$, $y_{(2,\text{source})}$, $x_{(2,\text{but})}$, $y_{(2,\text{but})}$ — coordinates of the ends of the first and second segments respectively. The segments will not have zero length.

Output

The first line must contain either `YES` or `NO`. In the case of positive answer, output in the second line the common measure with the precision of six digits after decimal point.

Example

`common.in`	`common.out`

`0 0 2 2 10 15 15 10`	`YES`
	`1.414214`

`0 0 0 1 0 0 1 1`	`NO`

Problem N. Nearest Approximation

Input file:	`nearest.in`
Output file:	`nearest.out`
Time limit:	5 seconds
Memory limit:	64 megabytes

You are given N integer numbers. Your task is to insert only one plus or minus sign between any two adjacent of them so as to make the value of the resulting expression as close as possible to a given integer number A.

Input

First line of the input file contains two integer numbers: N ($1 \le N \le 10000$) and A which cannot be greater than 10000 by an absolute value. N lines follow, each containing only one integer number X_i which does not exceed 10000 by an absolute value. Also it is guaranteed that the total sum of absolute values of all N numbers does not exceed 10000.

Output

In the first line you are to output the value of the resulting function (which has to be as close as possible to A). In the second line the optimal expression giving such a value must be displayed in the form $X_1[+|-]X_2[+|-]\dots X_{N-1}[+|-]X_N$.

Example

nearest.in	nearest.out

3 0
3
-2
1
	0
	3+-2-1

Problem O. Least Common Multiple

Input file:	`lcm.in`
Output file:	`lcm.out`
Time limit:	1 second
Memory limit:	64 megabytes

Calculate the least common multiple of all integers between 1 and n.

Input

One integer $1 \le n \le 1000$.

Output

One integer.

Example

`lcm.in`	`lcm.out`
3	6

Problem P. Permutations

Input file:	`perm.in`
Output file:	`perm.out`
Time limit:	1 second
Memory limit:	64 megabytes

Generate all permutations of numbers $1, \ldots, n$ in such an order that any two adjacent permutations differ only by the order of two adjacent elements (that is, $1, 2, 3, 4$ can be followed by $1, 2, 4, 3$, but not by $1, 4, 3, 2$).

Input

Input file contains an integer number $1 \le n \le 8$.

Output

Output file should contain $n!$ lines, n numbers each (separated by spaces).

Example

`perm.in`	`perm.out`

`3`	`1 2 3`
	`1 3 2`
	`3 1 2`
	`3 2 1`
	`2 3 1`
	`2 1 3`

Problem Q. String

Input file:	`string.in`
Output file:	`string.out`
Time limit:	1 second
Memory limit:	64 megabytes

Find the number of string S in the lexicographically ordered set of strings composed of the same set of letters.

Input

The only line contains the string S, composed of not more than 32 lowercase Italian characters.

Output

Output the number required.

Example

`string.in`	`string.out`
`abba`	`3`
`baba`	`5`

Problem R. Square Roots

Input file:	`sqrooots.in`
Output file:	`sqrooots.out`
Time limit:	1 second
Memory limit:	64 megabytes

Your task is to find the real roots of the equation $ax^2 + bx + c = 0$.

Input

Input file consists of three integers a, b and c. Their absolute values may not exceed 10^9.

Output

The first line of the output file represents the number of roots of the equation. The next k lines contain one real number each rounded up to six digits after decimal point — the root of this equation. The roots have to be displayed in ascending order. If there are infinitely many roots, output a single number -1 instead of the number of roots.

Example

sqrooots.in	sqrooots.out
1 -2 -3	2 -1.000000 3.000000

Problem S. Segments

Input file:	`segnemts.in`
Output file:	`segments.out`
Time limit:	1 second
Memory limit:	64 megabytes

You are to find the total length of union of n segments located on a horizontal line.

Input

The first number in the input file is an integer n ($1 \le n \le 5000$) representing the number of segments. The rest of file is filled by $2n$ integers ($x_1\ y_1\ x_2\ y_2\ \dots$) meaning coordinates of two points confining i segment. All coordinates do not exceed 10^9 by an absolute value.

Output

Print only one number — the total length of union of given segments.

Example

`segnemts.in`	`segments.out`

`3 0 5`
`3 6`
`7 8`
	`7`

Problem T. Combinations

Input file:	`combo.in`
Output file:	`combo.out`
Time limit:	1 second
Memory limit:	64 megabytes

A *combination* of k elements out of n is an increasing sequence of k integer numbers in range from 1 to n.

Combinations can be sorted in lexicographical order. Your task will be to find a specified combination by its position in that order.

Input

Input file contains three integer numbers n, k, l ($1 \le n \le 100$, $1 \le k \le n$, $1 \le l \le \binom{n}{k}$).

Output

Output file must contain k integer numbers, separated by spaces – l-th combination of k elements out of n in lexicographical order.

Example

`combo.in`	`combo.out`
`3 3 1`	`1 2 3`

Problem U. Prime Number

Input file:	`prime.in`
Output file:	`prime.out`
Time limit:	1 second
Memory limit:	64 megabytes

Output the k-th prime number.

Input

$k \leq 10000$.

Output

The answer.

Example

`prime.in`	`prime.out`
3	5
7	17

Problem V. Two Circles

Input file:	`circles.in`
Output file:	`circles.out`
Time limit:	1 second
Memory limit:	64 megabytes

There are two circles lying on the plane. Your task is to find all points of their intersection.

Input

The first line of the input file contains number of test cases K ($1 \le K \le 10000$). Each test case consists of two lines: first contains the description of the first circle, and the second one does for the second. The description of each circle is written in the form x, y, r ($-1000 \le x, y \le 1000$, $0 < r \le 1000$). All numbers are integer.

Output

For each test case you are to output one of the following messages:

- "`There are no points!!!`" — if there are no intersection points.
- "`There are only` i `of them....`" — if the circles have exactly i intersection points. In this case next i lines contain coordinates of the points x'_j y'_j. Points are to be displayed in the lexicographical order (first with the smallest x, if x coordinates are equal, first with the smallest y). Numbers are to be displayed with 6 digits after the decimal point.
- "`I can't count them - too many points :(`" — if there are infinitely many points of intersection.

All messages have to be displayed without quotes.
Separate output for different cases with a single blank line.

Example

circles.in	circles.out

2
0 0 2
4 0 2
0 0 1
1000 1000 1
	There are only 1 of them....
	2.000000 0.000000
	
	There are no points!!!

Problem W. Windows

Input file:	`widows.in`
Output file:	`widows.out`
Time limit:	1 second
Memory limit:	64 megabytes

n rectangular windows have appeared on the screen. Your task is to find the area covered simultaneously by all windows displayed.

Input

The first line of the input file contains one integer number n ($1 \le n \le 30000$). The next n lines contain four integer numbers each: x_1 y_1 x_2 y_2, where (x_1, y_1) are coordinates of the upper-left corner of the window, and (x_2, y_2) are ones for the down-right. All coordinates do not exceed 5000 by an absolute value. Note that the y coordinates on the display grow downwards.

Output

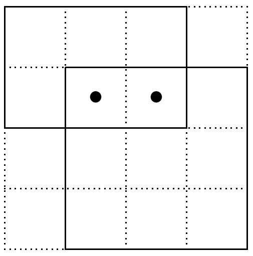
Write to the output file a single integer number — the area covered by all the windows displayed.

Example

widows.in	widows.out

2
0 0 3 2
1 1 4 4
	2

Picture



Problem X. Simple Sorting

Input file:	`sort.in`
Output file:	`sort.out`
Time limit:	1 second
Memory limit:	64 megabytes

You are given an unsorted array of 32-bit integer numbers. Your task is to sort this array and kill possible duplicated elements occuring in it.

Input

The first line of the input file contains an integer number N representing the quantity of numbers in this array ($1 \leq N \leq 65536$). Next N lines contain N 32-bit integer numbers (one number per each line) of the original array.

Output

Output file should contain at most N numbers sorted in descending order if the value of N was even, otherwise the numbers are to be sorted in ascending order. Every number in the output file should occure only once.

Example

sort.in	sort.out
6 8 8 7 3 7 7	8 7 3

Problem Y. Fidonacci Numbers

Input file:	`fido.in`
Output file:	`fido.out`
Time limit:	1 second
Memory limit:	64 megabytes

Fidonacci numbers are defined by conditions $F_0 = 0$, $F_1 = F_2 = 1$ and $F_k = F_{k-1} + F_{k-3}$ for all $k \geq 3$. You are to find F_N for a given N.

Input

One integer number N, $0 \leq N \leq 1000$.

Output

Just output the N-th Fidonacci number.

Example

`fido.in`	`fido.out`
0	0
1	1
2	1

Problem Z. Parentheses

Input file:	`paren.in`
Output file:	`paren.out`
Time limit:	1 second
Memory limit:	64 megabytes

Generate correct parenthesis sequences in lexicographic order.
Correct parenthesis sequences are given by the rule $S ::= \texttt{()} | \texttt{(}S\texttt{)} | SS$
Lexicographic order is imposed by assuming `(` less than `)`.

Input

Input file contains one integer number n, $(1 \le n \le 10)$.

Output

Output all correct parenthesis sequences of length $2n$, one sequence a line, without spaces.

Example

paren.in	paren.out
3	((())) (()()) (())() ()(()) ()()()