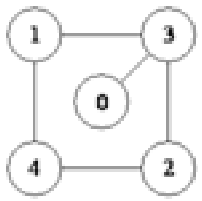
Problem I. Graph of Inversions

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

For given permutation $p_0, p_1, \ldots, p_{n-1}$ of $\{0, 1, \ldots, n-1\}$ two indices i, j are inversion iff $0 \leq i < j < n$ and $p_i > p_j$. Let's build a graph of inversions. It contains n vertices and edges (i, j) for each inversion of indices i, j.

For example, the graph of inversions for permutation $(1, 3, 4, 0, 2)$ is:



Subset of graph vertices S is *independent* if there is no edge (u, v) where $u \in S$ and $v \in S$. Subset of graph vertices S is *dominating* if each $v \notin S$ connected with at least one vertex from S.

You are given a graph of inversions of some permutation. Write a program to find number of graph vertices subsets S such that S is *independent* and *dominating* at the same time.

Input

The first line of input contains two integers n ($1 \leq n \leq 100, 0 \leq m \leq \min(1000, n \cdot (n-1)/2)$), where n is number of vertices and m is number of edges. Following m lines contain edges, one edge per line. There are no multiple edges between a pair of vertices.

Output

Print the answer. The answer will not exceed 10^{18}.

Example

standard input	standard output

5 5
1 3
1 4
0 3
2 3
2 4
	3

4 6
0 1
0 2
0 3
1 2
1 3
2 3
	4

Problem J. Common Permutation

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

Given two strings of lowercase letters, a and b, print the longest string x of lowercase letters such that there is a permutation of x that is a subsequence (do not confuse it with substring) of a and there is a permutation of x that is a subsequence of b.

Input

Input consists of pairs of lines. The first line of a pair contains a and the second contains b. Each string is on a separate line and consists of at most 1000 lowercase letters.

Output

For each subsequent pair of input lines, output a line containing x. If several x satisfy the criteria above, choose the first one in alphabetical order.

Example

standard input	standard output
pretty women walking down the street	e nw et

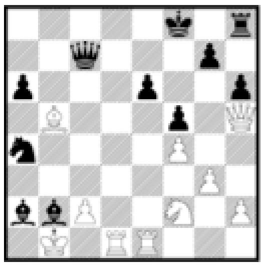
Problem K. Chessboard in FEN

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

In the FEN (Forsyth-Edwards Notation), a chessboard is described as follows:

- The Board-Content is specified starting with the top row and ending with the bottom row.
- Character "/" is used to separate data of adjacent rows.
- Each row is specified from left to right.
- White pieces are identified by uppercase piece letters: `PNBRQK`.
- Black pieces are identified by lowercase piece letters: `pnbrqk`.
- Empty squares are represented by the numbers one through eight.
- A number used represents the count of contiguous empty squares along a row.
- Each row's sum of numbers and characters must equal 8.

For example: `5k1r/2q3p1/p3p2p/1B3p1Q/n4P2/6P1/bbP2N1P/1K1RR3` is the FEN notation description of the following chessboard:



The chessboard of the beginning of a chess game is described in FEN as: `rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR`

Your task is simple: given a chessboard description in a FEN notation you are asked to compute the number of unoccupied squares on the board which are not attacked by any piece.

Input

Input is a sequence of lines, each line containing a FEN description of a chessboard. Note that the description does not necessarily give a legal chess position. Input lines do not contain whitespace.

Output

For each line of input, output one line containing an integer which gives the number of unoccupied squares which are not attacked.

Example

standard input
5k1r/2q3p1/p3p2p/1B3p1Q/n4P2/6P1/bbP2N1P/1K1RR3 rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR

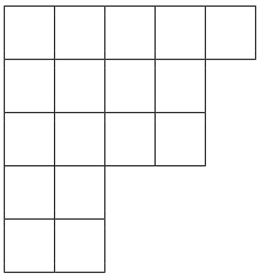
standard output
3 16

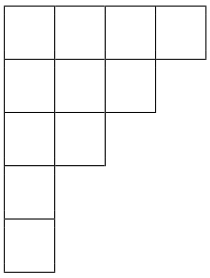
Problem L. Young Diagrams

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

Often it is useful to represent some object graphically. For partitions like $n = l_1 + l_2 + \cdots + l_k$ (where $l_i \geq l_{i+1}$) one can draw Young diagram. It is a finite collection of boxes, or cells, arranged in left-justified rows, with the row lengths weakly decreasing (each row has the same or shorter length than its predecessor). The length of the i-th row is l_i. Listing the number of boxes in each row gives a partition of a non-negative integer n, the total number of boxes of the diagram. The Young diagram is said to be of shape partition, and it carries the same information as that partition.

Picture shows diagrams for partitions $5+4+4+2+2$ and $4+3+2+1+1$.





Sometimes it is interesting to use a pair of two objects: a partition of n and a permutation of n elements. They are connected by a partition's Young diagram with numbered boxes. Boxes are numbered from 1 to n in such a way that in each row (from left to right) and each column (from top to bottom) the values are in increasing order. The numbers written from left to right in each row from top to bottom should make the permutation.

Write a program to find the number of Young diagrams by a given permutation.

Input

The first line of the input contains integer n ($1 \leq n \leq 1000$). The second line contains the given permutation.

Output

Print the number of required Young diagrams.

Example

standard input	standard output

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

Two possible Young diagrams are: