GPC-UPC DP Short Contest

A. Rectangle Painting 1

1 second, 256 megabytes

There is a square grid of size $n \times n$. Some cells are colored in black, all others are colored in white. In one operation you can select some rectangle and color all its cells in white. It costs $\max(h, w)$ to color a rectangle of size $h \times w$. You are to make all cells white for minimum total cost.

Input
The first line contains a single integer n ($1 \le n \le 50$) — the size of the square grid.

Each of the next n lines contains a string of length n, consisting of characters '.' and '#'. The j-th character of the i-th line is '#' if the cell with coordinates (i, j) is black, otherwise it is white.

Output
Print a single integer — the minimum total cost to paint all cells in white.

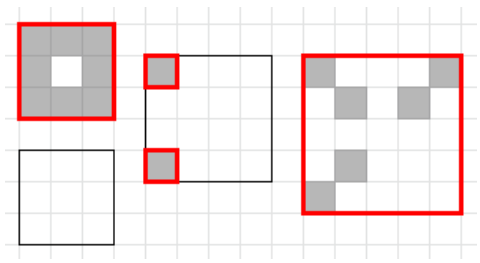
input
3 ### #.# ###
output
3

input
3 ... ...
output
0

input
4 #... .... #...
output
2

input
5 #...# .#.#. ..... .#... #....
output
5

The examples and some of optimal solutions are shown on the pictures below.



B. Queries for Number of Palindromes

5 seconds, 256 megabytes

You've got a string $s = s_1 s_2 ... s_{|s|}$ of length $|s|$, consisting of lowercase English letters. There also are q queries, each query is described by two integers l_i, r_i ($1 \le l_i \le r_i \le |s|$). The answer to the query is the number of substrings of string $s[l_i ... r_i]$, which are palindromes.

String $s[l ... r] = s_l s_{l+1} ... s_r$ ($1 \le l \le r \le |s|$) is a *substring* of string $s = s_1 s_2 ... s_{|s|}$.

String t is called a *palindrome*, if it reads the same from left to right and from right to left. Formally, if $t = t_1 t_2 ... t_{|t|} = t_{|t|} t_{|t|-1} ... t_1$.

Input
The first line contains string s ($1 \le |s| \le 5000$). The second line contains a single integer q ($1 \le q \le 10^6$) — the number of queries. Next q lines contain the queries. The i-th of these lines contains two space-separated integers l_i, r_i ($1 \le l_i \le r_i \le |s|$) — the description of the i-th query.

It is guaranteed that the given string consists only of lowercase English letters.

Output
Print q integers — the answers to the queries. Print the answers in the order, in which the queries are given in the input. Separate the printed numbers by whitespaces.

input
caaaba 5 1 1 1 4 2 3 4 6 4 5
output
1 7 3 4 2

Consider the fourth query in the first test case. String $s[4 ... 6]$ = «aba». Its palindrome substrings are: «a», «b», «a», «aba».

C. Antimatter

1 second, 256 megabytes

Iahub accidentally discovered a secret lab. He found there n devices ordered in a line, numbered from 1 to n from left to right. Each device i ($1 \le i \le n$) can create either a_i units of matter or a_i units of antimatter.

Iahub wants to choose some contiguous subarray of devices in the lab, specify the production mode for each of them (produce matter or antimatter) and finally take a photo of it. However he will be successful only if the amounts of matter and antimatter produced in the selected subarray will be the same (otherwise there would be overflowing matter or antimatter in the photo).

You are requested to compute the number of different ways Iahub can successful take a photo. A photo is different than another if it represents another subarray, or if at least one device of the subarray is set to produce matter in one of the photos and antimatter in the other one.

Input
The first line contains an integer n ($1 \le n \le 1000$). The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 1000$).

The sum $a_1 + a_2 + ... + a_n$ will be less than or equal to 10000.

Output
Output a single integer, the number of ways Iahub can take a photo, modulo 1000000007 ($10^9 + 7$).

input
4 1 1 1 1

output
12

The possible photos are [1+, 2-], [1-, 2+], [2+, 3-], [2-, 3+], [3+, 4-], [3-, 4+], [1+, 2+, 3-, 4-], [1+, 2-, 3+, 4-], [1+, 2-, 3-, 4+], [1-, 2+, 3+, 4-], [1-, 2+, 3-, 4+] and [1-, 2-, 3+, 4+], where "i+" means that the i-th element produces matter, and "i-" means that the i-th element produces antimatter.

D. The least round way

2 seconds, 64 megabytes

There is a square matrix $n \times n$, consisting of non-negative integer numbers. You should find such a way on it that

- starts in the upper left cell of the matrix;
- each following cell is to the right or down from the current cell;
- the way ends in the bottom right cell.

Moreover, if we multiply together all the numbers along the way, the result should be the least "round". In other words, it should end in the least possible number of zeros.

Input
The first line contains an integer number n ($2 \le n \le 1000$), n is the size of the matrix. Then follow n lines containing the matrix elements (non-negative integer numbers not exceeding 10^9).

Output
In the first line print the least number of trailing zeros. In the second line print the correspondent way itself.

input
3 1 2 3 4 5 6 7 8 9

output
0 DDRR

E. Longest Regular Bracket Sequence

2 seconds, 256 megabytes

This is yet another problem dealing with regular bracket sequences.

We should remind you that a bracket sequence is called regular, if by inserting «+» and «1» into it we can get a correct mathematical expression. For example, sequences «(())()», «()» and «(()(()))» are regular, while «)(», «(()» and «(()))(» are not.

You are given a string of «(» and «)» characters. You are to find its longest substring that is a regular bracket sequence. You are to find the number of such substrings as well.

Input
The first line of the input file contains a non-empty string, consisting of «(» and «)» characters. Its length does not exceed 10^6.

Output
Print the length of the longest substring that is a regular bracket sequence, and the number of such substrings. If there are no such substrings, write the only line containing "0 1".

input
)(((((((()()())

output
6 2

input
))(

output
0 1