D. Huge Strings

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You are given n strings $s_1, s_2, ..., s_n$ consisting of characters 0 and 1. m operations are performed, on each of them you concatenate two existing strings into a new one. On the i-th operation the concatenation $s_{a_i}s_{b_i}$ is saved into a new string s_{n+i} (the operations are numbered starting from 1). After each operation you need to find the maximum positive integer k such that all possible strings consisting of 0 and 1 of length k (there are 2^k such strings) are substrings of the new string. If there is no such k, print 0.

Input

The first line contains single integer n ($1 \le n \le 100$) — the number of strings. The next n lines contain strings $s_1, s_2, ..., s_n$ ($1 \le |s_i| \le 100$), one per line. The total length of strings is not greater than 100.

The next line contains single integer m ($1 \le m \le 100$) — the number of operations. m lines follow, each of them contains two integers a_i abd b_i ($1 \le a_i, b_i \le n + i - 1$) — the number of strings that are concatenated to form s_{n+i}.

Output

Print m lines, each should contain one integer — the answer to the question after the corresponding operation.

Example

input

```
5
01
10
101
11111
0
3
1 2
6 5
4 4
```

output

```
1
2
0
```

Note

On the first operation, a new string "`0110`" is created. For $k = 1$ the two possible binary strings of length k are "`0`" and "`1`", they are substrings of the new string. For $k = 2$ and greater there exist strings of length k that do not appear in this string (for $k = 2$ such string is "`00`"). So the answer is 1.

On the second operation the string "`01100`" is created. Now all strings of length $k = 2$ are present.

On the third operation the string "`1111111111`" is created. There is no zero, so the answer is 0.