Finding Keywords

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

In search algorithms, we often encounter the following problem: there are many entries and a sequence of keywords, and you want to find entries that match the keywords. For example, the entry "2025 China Collegiate Programming Contest (CCPC) Jinan Regional Contest" matches the keywords "2025 CCPC Jinan". Note that the keyword sequence does not need to appear consecutively in the entry; it only needs to appear as a subsequence of the entry.

Based on practical considerations, we can define the following relevance algorithm: select as many **non-overlapping** subsequences from the entry as possible, such that each subsequence is exactly equal to the given keyword sequence; then the relevance is the number of subsequences.

In this problem, each word can be represented by a positive integer. Given a keyword sequence of length m, to simplify the problem, the keywords are **all distinct**. Additionally, n entries are provided, where the i-th entry is a sequence of positive integers of length l_i. Your goal is to calculate the relevance for each entry.

Input

The first line of input contains an integer m ($1 \leq m \leq 10$), representing the length of the keyword sequence.

The next line contains m integers $b_1, b_2, \ldots, b_m$ ($1 \leq b_i \leq m$), representing the keyword sequence. It is guaranteed that $b_1, b_2, \ldots, b_m$ are all distinct.

The next line contains an integer n ($1 \leq n \leq 100$), representing the number of entries.

The following n lines: the i-th line first reads an integer l_i ($1 \leq l_i \leq 10^6$), representing the length of the i-th entry. Then it reads l_i integers $a_{i,1}, a_{i,2}, \ldots, a_{i,l_i}$ ($1 \leq a_{i,j} \leq 10^6$), representing the content of the i-th entry.

It is guaranteed that the total length of all entries does not exceed 10^6.

Output

Output n lines, where the i-th line contains an integer representing the relevance of the i-th entry.

Example

standard input	standard output
4 4 2 1 3 3 8 4 2 1 3 4 2 3 1 9 1 4 2 4 1 2 1 3 3 12 1 1 2 3 4 2 1 1 2 1 3 3	1 2 1

Note

Below, the index sequences are used to represent subsequences.

For the first entry, you can choose $[1, 2, 3, 7]$ as a subsequence, at which point $[a_1, a_2, a_3, a_7] = [4, 2, 1, 3]$, which is exactly equal to the keyword sequence.

For the second entry, you can choose $[2, 3, 5, 9]$ and $[4, 6, 7, 8]$ as subsequences. Note that $[2, 3, 7, 9]$ and $[4, 6, 7, 9]$ cannot be chosen as selected subsequences because their positions overlap.