ETH Zurich
Institut für Theoretische Informatik
Prof. Dr. Angelika Steger
Prof. Dr. Emo Welzl
Prof. Dr. Peter Widmayer

Algorithms Lab

Exercise – *Search Snippets*

A high-school friend of yours has recently founded a startup. He is marketing a new product for searching the extensive archives that major companies tend to accumulate over the years of their existence.

A first user study at the offices of National Office Hardware (NOH) was recently conducted, to compare the new product to certain well-known web search engines. Most of these engines display short text snippets that show some context of where the search terms came up. Users at NOH asked for a similar feature in your archive search tool.

Inevitably, your friend asks you to assist him in designing a proper way of finding suitable snippets. You decide to look for a *single, contiguous, shortest possible* (measured in the number of words contained) snippet that contains all the search terms.

You are given, for each word and document, a list of the positions at which this word occurs. For instance, in the preceding three paragraphs the word "snippets" appears as the 69th and the 110th word, and so the corresponding list of positions (counting from zero) is `68 109`.

As your program will be run for every document matching a query, it must be very fast even on large documents!

Input The first line of the input contains the number of test cases $t \leq 40$. Each of the following t test cases describes a word/position index for one document.

- It starts with a line that contains one integer $2 \leq n \leq 2^{11}$, the number of words in the query.
- The following line contains integers $m_0, \ldots, m_{n-1}$ ($1 \leq m_i \leq 2^{16}/n$), separated by single spaces, where m_i denotes the number of times word i occurs within the document.
- n lines follow. The i-th such line contains m_i integers $p_{i,0}, \ldots, p_{i,m_i-1}$ ($0 \leq p_{i,0} < \ldots < p_{i,m_i-1} \leq 2^{30}$), separated by single spaces. Each p_{ij} denotes one position of the i-th word in the document. Within a test case, you may assume that all p_{ij} are pairwise distinct.

Output For every test case the corresponding output appears on a separate line. It consists of one non-negative integer, the length $b - a + 1$ of a shortest possible interval $[a, b]$ such that every word occurs at least once within $[a, b]$. (That is, for every $i \in \{0, \ldots, n-1\}$, there is a $j \in \{0, \ldots, m_i - 1\}$ with $p_{ij} \in [a, b]$.)

Points There are three groups of test sets. The individual points are specified below; the total number of points is 100.

1. For the first group of test sets, worth 50 points, you may assume that $n = 2$. Corresponding example test sets are contained in the files `snippets-test1.in/out`.

2. For the second group of test sets, worth 25 points, you may assume that $n \leq 2^8$. Corresponding example test sets are contained in the files `snippets-test2.in/out`.
3. For the third (hidden) group of test sets, worth 25 points, there are no additional assumptions. Corresponding example test sets are contained in the files `snippets-test3.in/out`.

Sample Input

```
2
2
4 3
1 2 5 18
4 15 16
3
3 2 2
1 7 10
2 3
6 11
```

Sample Output

```
2
5
```

Note: Put `std::ios_base::sync_with_stdio(false)` as the first line of the main procedure for faster I/O.