

IOI '00 P4 - Walls

Time limit: 1.0s Memory limit: 16M

IOI '00 - Beijing, China

In a country, great walls have been built in such a way that every great wall connects exactly two towns. The great walls do not cross each other. Thus, the country is divided into such regions that to move from one region to another, it is necessary to go through a town or cross a great wall. For any two towns A and B , there is at most one great wall with one end in A and the other in B , and further, it is possible to go from A to B by always walking in a town or along a great wall. The input format implies additional restrictions.

There is a club whose members live in the towns. In each town, there is only one member or there are no members at all. The members want to meet in one of the regions (outside of any town). The members travel riding their bicycles. They do not want to enter any towns, because of the traffic, and they want to cross as few great walls as possible, as it is a lot of trouble. To go to the meeting region, each member needs to cross a number (possibly 0) of great walls. They want to find such an optimal region that the sum of these numbers (crossing-sum, for short) is minimized.

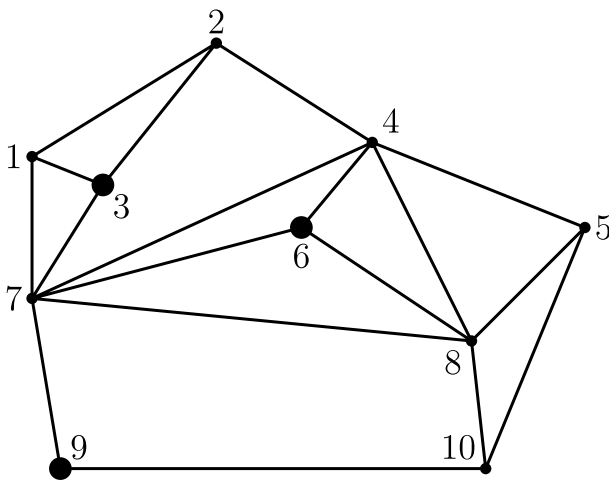


Figure 1

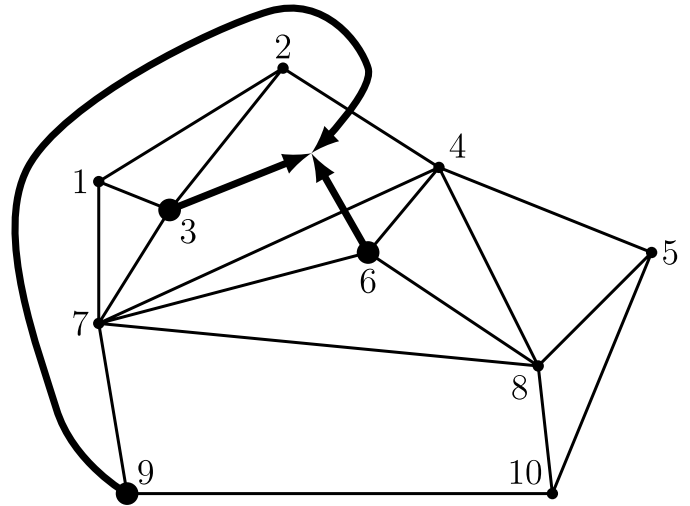


Figure 2

The towns are labeled with integers from 1 to N , where N is the number of towns. In Figure 1, the labeled nodes represent the towns and the lines connecting the nodes represent the great walls. Suppose that there are three members, who live in towns 3, 6, and 9. Then, an optimal meeting region and respective routes for members are shown in Figure 2. The crossing-sum is 2: the member from town 9 has to cross the great wall between towns 2 and 4, and the member from town 6 has to cross the great wall between towns 4 and 7.

You are to write a program which, given the towns, the regions, and the club member home towns, computes an optimal region and the minimal crossing-sum.

Input Specification

The first line contains one integer: the number of regions M ($2 \leq M \leq 200$). The second line contains one integer: the number of towns N ($3 \leq N \leq 250$). The third line contains one integer: the number of club members L ($1 \leq L \leq 30$). Of course, $L \leq N$. The fourth line contains L distinct integers in increasing order: the labels of the towns where the members live.

After that the file contains $2M$ lines so that there is a pair of lines for each region: the first pair of lines describe the first region, the following pair the second and so on. Of the pair, the first line shows the number of towns I on the border of that region. The second line of the pair contains I integers: the labels of these I towns in some order in which they can be passed when making a trip clockwise along the border of the region, with the following exception: the last region is the "outside region" surrounding all towns and other regions, and for it the order of the labels corresponds to a trip in the counterclockwise direction. The order of the regions gives an integer labelling of the regions: the first region has label 1, the second has label 2, and so on. Note that the input includes all regions formed by the towns and great walls, including the "outside region".

Output Specification

The output file should have two lines. The first line should contain one integer: the minimal crossing-sum. The second line should contain one integer: the label of an optimal region. There may be several different solutions for the region and your program needs to output only one of them.

Sample Input

```
10
10
3
3 6 9
3
1 2 3
3
1 3 7
4
2 4 7 3
3
4 6 7
3
4 8 6
3
6 8 7
3
4 5 8
4
7 8 10 9
3
5 10 8
7
7 9 10 5 4 2 1
```

Sample Output

2

3

The sample input and output shown correspond to the diagram above.