2490 Clock Synchronisation

Consider a set of C clocks. Each clock has a single hour hand that can only point to one of the following hours: 3, 6, 9 or 12. These clocks can be moved in B different ways called block moves. Each block move applies to a specified subset of clocks, and consists of moving the hand of each clock in the subset clockwise by 3 hours. Your task is to find a minimal sequence of block moves that sets all clocks to 12 hours. Multiple repetitions of a single block move are allowed. If two solutions of the same length exist, then give the lesser one in dictionary order. If there is no solution then write '`No solution.`'. If no move is required then write '`No moves required.`'.

For example, consider a set of 9 clocks ($C = 9$) with initial positions as shown in the following table, and 9 available block moves ($B = 9$), where ticks indicate the clocks affected by each block move:

Clock numbers	0	1	2	3	4	5	6	7	8
Clock positions	9	9	12	6	6	6	6	3	6
Block move 0	√	√		√	√				
Block move 1	√	√	√						
Block move 2		√	√		√	√			
Block move 3	√			√			√		
Block move 4		√		√	√	√		√	
Block move 5			√			√			√
Block move 6				√	√		√	√	
Block move 7							√	√	√
Block move 8					√	√		√	√

With clocks in the initial positions shown in the above table, applying block move 3 would result in the following updated clock positions (clocks 0, 3, and 6 have been moved clockwise by 3 hours):

Clock numbers	0	1	2	3	4	5	6	7	8
Clock positions	12	9	12	9	6	6	9	3	6

Further applying block moves 4, 7, and 8 would result in all clocks being set to 12 hours. The move sequence 3, 4, 7, 8 is actually the required solution in this case.

Input

Input consists of a number of scenarios. Each scenario starts with a line containing a scenario title, which is a string of 1 to 20 letters, digits, and underscores (with no intervening spaces). A single '`#`' on a line indicates the end of input.

The "title" line is followed by one line consisting of two integers C and B, separated by a single space:

- C is the number of clocks, $1 \leq C \leq 11$, where the clocks are numbered sequentially starting with 0 (i.e., $0, 1, 2, 3, \ldots, C-1$),
- B is the number of blocks, $0 \leq B \leq 11$, where the blocks are numbered sequentially starting with 0 (i.e., $0, 1, 2, 3, \ldots, B-1$).

This line is followed by one line representing the starting positions of our clocks. This line consists of C clock positions separated by single spaces, each position being an integer number in the set {3, 9, 6, 12}.

This "clocks" line is followed by B other lines, one line for each block move operation. Each "block" line consists of 1 or more clock numbers (in no particular order), separated by single spaces, representing the clocks that will advance together by 3 hours in this block.

Output

Each set of output data consists of a single output line showing in order: the problem title, a colon '`:`', a space '` `', and one of the following answers:

- the text '`No moves required.`', if no moves are required to solve the problem,
- the text '`No solution.`', if the problem doesn't have any solution,
- the block numbers making up the minimal solution (in the required length/dictionary order), separated by single spaces.

Sample Input

```
Test_1_0
1 0
12
Test_1_1
1 1
12
0
Test_3_3
3 3
9 9 9
0 1
1 2
0 2
Test_9_9
9 9
9 9 12 6 6 6 6 3 6
1 3 4 0
0 1 2
2 4 5 1
0 3 6
4 5 7 1 3
2 5 8
3 4 6 7
8 7 6
8 7 4 5
#
```

Sample Output

```
Test_1_0: No moves required.
Test_1_1: No moves required.
Test_3_3: No solution.
```

```
Test_9_9: 3 4 7 8
```