Matrissor is a special kind of processor which can multiply a sequence of matrices in quick time. It has certain capacity K which means the maximum number of computations (multiplications here) it can perform at one step. For example if K is 1000, then it can multiply 2 matrices of 10×10 dimension. But it cannot multiply a (10×11) matrix and another (11×10) matrix which require 1100 multiplications. There is a limitation of matrissor. It cannot multiply a sequence of matrices optimally. If it is to multiply m matrices, it processes first $(m-1)$ matrices first and then multiples the resultant matrix with mth matrix.

Your task is to multiply a sequence of matrices optimally using the matrissor with capacity K. Here optimality depends on one criterion. You have to use the matrissor minimum number of times. Say you have 4 matrices available - $M_1(10 \times 1)$, $M_2(1 \times 10)$, $M_3(10 \times 1)$ and $M_4(1 \times 10)$. Now if you use a 100 capacity matrissor, then you can multiply M_2, M_3 and M_4 in one step and in last step you can multiply M_1, (M_2, M_3, M_4). This can be expressed as $(M_1, (M_2, M_3, M_4))$, where (M_2, M_3, M_4) denotes the resultant matrix after multiplying M_2, M_3, M_4.

Input

The input file contains the number of test cases T first, which is at most 30. Each test case begins with a positive integer $N (2 \leq N \leq 50)$ which is the number of matrices. Following N lines contain the dimensions of matrices, one line per matrix. Dimensions will be valid and any dimension will be in between 1 to 50. Next line will contain another integer $Q (1 \leq Q \leq N)$ which is the number of queries, followed by the capacities of the matrissor in one line. Each test case will be followed by a blank line.

Output

For each set of input, print a line '`Matrix #`D' in first line, where D is the test case number starting from 1. In next Q lines print the minimum number of steps to multiply all the matrices. If it is not possible to multiply the matrices, then print '`Impossible.`'. Put a blank line after each output set. See sample output for details.

Sample Input

```
2
4
10 1
1 10
10 1
1 10
3
100 99 300

4
1 1
1 1
1 1
1 1
2
1 2
```

Sample Output

```
Matrix #1
2
Impossible.
1

Matrix #2
3
2
```