Permutation

Input/output:input.txt, output.txt
Time limit: 4 seconds
Author: Volkov I.A.
Tests: http://byoi.narod.ru/tests/t02_1.zip

An array of integers A[1..N] consists of N elements. (1<=N<=20000). The elements of the array are integer numbers from 1 to N, all numbers are different. An arrayB[1..N] is built from the array A by the following rule:

B[A[A[i]]]=i, for each i from 1 to N

Your program should input an array B and find an array A, if it is possible. . If there are many solutions, output any of them.

Input. First line of the file contains an integer N - the count of elements in array B. In N the following lines integer numbers B[1], … , B[N] are written. Each line consists one integer.

Output. The first line of output should contain an integer N - the count of elements in array A, or the number 0, if there is no such array A. In case, when N>0 the following N lines should contain integer numbers A[1], … , A[N]. Each line should contain one number.

Example.

Input	*Output*
4 3 4 1 2	4 2 3 4 1

Stars

Input Data: Input1.Txt, Input2.Txt, …, Input10.Txt
Output Data: Output1.Txt, Output2.Txt, ..., Output10.Txt
Time limit: No
Author: Metelsky I.S.
Tests: http://byoi.narod.ru/tests/t02_2.zip

Petya Bulochkin decided to become a programmer and to find a job. Whole days he visited different computer firms, but everywhere they refused to get him job. At least, firm “No Bugs” agreed to get him job, but only after he passed an exams .

Petya was given a program, which inputs an integerN (1<=N<=1000000) and numbers A_i (1<=A_i<=N, all A_i are different), and writes on the screen some count of stars T. To pass exams, Petya should determine, how much stars the program writes. Petya was given 10 test examples. He had to give an answers for this examples, and it is not important, which way way he get this answers.

The program was as follows :

```
{Pascal}
var
   A: Array[0..1000000] of LongInt;
   i, j, n, s: LongInt;
begin
   A[0]:= 0;
   ReadLn(n);
```

```
      For i:= 1 to n do
         ReadLn(A[i]);
      For i:= 1 to n do begin
         j:= i;
         While (j>1) and (A[j] < A[j-1]) do begin
            Write('*');
            s:= A[j]; A[j]:= A[j-1]; A[j-1]:= s;
            j:= j-1;
         end;
      end;
   end.

   // C++
   #include <iostream>
   using namespace std;
   int A[1000001]; // an array is defined A, indexes from 0 to
1000000
   int main()
   {
      int n, i, j, s;
      A[0] = 0;
      cin >> n;
      for (i=1; i<=n; i++) cin >> A[i];
      for (i=1; i<=n; i++)
      {
         j = i;
         while ((j>1) && (A[j] < A[j-1]))
         {
            cout << "*";
            s = A[j]; A[j] = A[j-1]; A[j-1] = s;
            j--;
         }
      }
      return 0;
   }
```

Petya asked you to help him.

So, you have 10 input files (format is described below). Each of the files contains a series of numbers - input for the program. You should create 10 output files, each, containing one integer number - the count of stars, program will print on corresponding input data. You should not present any program, **you should only find the right output files**. Output for the input file *InputK.Txt* (1 <= K <= 10) should be file *OutputK.Txt*.

Input. First line of input contains an integer N. The next N lines contains integer numbers A_1, A_2, … , A_N . Each line contains one number.

Example.

Input0.Txt	*Output0.Txt*
5 1 5 2	4

4 3	

Taxi

Input/Output:input.txt, output.txt
Time limit: 3 seconds
Author: Metelsky I.S.
Tests: http://byoi.narod.ru/tests/t02_3.zip

On the map of the city , in which Petya Bulochking works as a taxi driver, there are N (1<=N<=1000) intersections, numbered from 1 to N. Some intersections are connected by roads. This roads are so narrow, that one-way traffic was entered on them . Experienced Petya knows, that, starting from some intersection and having driven some roads, it is never possible to come to the starting intersection . There are no two roads, connecting the same pair of intersections. The only common points of roads are intersections.

Once, on intersection A Petya picked up a passenger, who asked to take him to intersection B, and to stop during the travel on intersections C_1, C_2, … , C_k (the passenger had meetings with some people on this intersections). An order of visiting of intermediate intersections is not important. “How much money will I take from this passenger?” - Petya thought. He decided to take from the passenger such amount of roubles, as there exists different ways to get from the intersection A to the intersection B, and visit intermediate intersections C_1, C_2, … , C_k.

Task. Your program should input information about roads in the city, and intersections numbers A, B, C_1, C_2, … , C_k and define, how much ways there exist to get from the intersection A to the intersection B, and visit intermediate intersections C_1, C_2, … , C_k.

Input. The first line of input contains integers N - the count of intersections and M - the count of roads. The next M lines contain the description of roads. Each of this line contains two integers - starting and finishing intersection for the road. Pay attention, that it is not possible to drive on a road in reverse direction (from finishing intersection to starting), because the traffic is one-way. Next line contains integers A - starting intersection, B - finishing intersection and K - the count of intermediate intersections. And, at least, next K lines contain integers C_1, C_2, … , C_k . Each line contains one integer. The integers A, B, C_1, C_2, … , C_k are all different.

Output. You should output the unique number S - the count of ways to get from the intersection A to the intersection B, and visit intermediate intersections C_1, C_2, … , C_k. If there are no way to do it, output should contain 0. There will be no inputs, where S exceeds 2000000000.

Example.

Input	*Output*
4 4 1 2 2 3 1 4 4 3 1 3 1 2	1