South African Computer Olympiad
Training Camp 3, 2006
Day 2

Overview

Author	**Timothy Stranex**	**USAICO 2006**	**Bruce Merry (BIO 2004)**
Problem	**bank**	**specials**	**fsm**
Source	N/A	specials.c specials.cpp specials.pas	fsm.c fsm.cpp fsm.pas
Input file	bank.in	specials.in	fsm.in
Output file	bank.out	specials.out	fsm.out
Time limit	N/A	1 second	1 second
Number of tests	10	10	20
Points per test	10	10	5
Total points	**100**	**100**	**100**

The maximum total score is 300 points.

South African Computer Olympiad
Training Camp 3, 2006
Day 2

Bank Robbing

Author

Timothy Stranex

Introduction

The villainous Sheriff is overtaxing the people of Nottingham into poverty. He stores this money in the town bank and has instructed the clerks to allow only people who give the correct password to access his account. He is careful to change the password every day. Robin Hood wants to rob the Sheriff's account and return the money to its rightful owners.

Task

The Sheriff's password on day i is an integer p_i which is a linear combination of the previous K passwords, modulo M:

$$p_i = (a_1 p_{i-1} + a_2 p_{i-2} + \cdots + a_K p_{i-K}) \bmod M.$$

Robin has managed to obtain the numbers K, M, $a_1, a_2, \ldots, a_K$ and $p_1, p_2, \ldots, p_K$. You must help Robin to work out p_{K+N}, the password N days later.

Example

If $K = 2$, $a_1 = a_2 = 1$, $p_1 = 0$, $p_2 = 1$ and $M = 1000$, the passwords follow the Fibonacci sequence $0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots$ modulo 1000. After $N = 4$ days, the password will be 5. After $N = 16$ days, the Fibonacci number is 1597 so the password is 597.

Input (bank.in)

The first line contains three space-separated integers, K, M and N. The second line contains K space-separated integers $a_1, a_2, \ldots, a_K$. The third line contains K space-separated integers $p_1, p_2, \ldots, p_K$.

Sample input

```
2 1000 16
1 1
0 1
```

Output (bank.out)

The output is the single integer p_{K+N} followed by a new line.

Sample output

```
597
```

Constraints

- $1 \leq K \leq 50$
- $1 \leq M < 2^{15}$
- $0 \leq a_i < 2^{15}$
- $0 \leq p_i < M$
- $1 \leq N \leq 2^{31} - 1$

50% constraints

- $1 \leq N \leq 10^5$

Time limit

N/A.

Scoring

A correct answer scores 100%. An incorrect answer or badly formatted output scores 0%.

South African Computer Olympiad
Training Camp 3, 2006
Day 2

Specials

Author

USAICO 2006

Introduction

Fred the manic storekeeper is trying to plan his special offers for the coming year (he likes to plan in advance). In order to do this, he is going to mine his sales database to determine the most attractive offers. Fred's store has K different items for sale, which have product IDs from 1 to K. Fred has a complete list of all items sold last year, in the order that they were sold. He now wants to determine which items were sold over particular periods of time. He has asked for your help with this.

Task

You will be given the list of product IDs of the items as they are sold. You will also be given a list of queries. Each query is a contiguous range of sales from the list. Within each range, Fred ideally wants to know which items were sold at least once. However, you have pointed out that this may be a long list. He has settled for knowing the *sum* of the distinct product IDs in each range, modulo 1 000 000.

Example

Suppose the sales list was 1, 2, 2, 1, 3, 1. In the range $[2, 5]$, the distinct IDs are 1, 2 and 3, so the answer is 6. In the range $[1, 4]$, the distinct IDs are 1 and 2, so the answer is 3.

Input (specials.in)

The first line of input contains three integers separated by spaces, N, K and Q. N is the length of the sales list, K is the maximum product ID, and Q is the number of queries.

The next N lines contain the sales list, with one product ID per line.

The next Q lines describe the queries. Each query is a line with two integers A and B ($1 \leq A \leq B \leq N$), which is a query of items A through B inclusive of the sales list.

Sample input

```
6 3 2
1
2
2
1
3
1
2 5
1 4
```

Output (specials.out)

The output consists of Q lines. The ith line contains a single integer, which is the sum of the distinct product IDs in the range of query i, modulo 1 000 000.

Sample output

```
6
3
```

Constraints

- $1 \leq K \leq N \leq 100\ 000$
- $0 \leq Q \leq 25\ 000$

50% constraints

- $1 \leq K \leq 400$

Time limit

1 second.

Finite State Machines

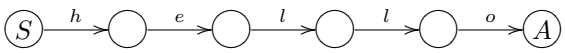
Author

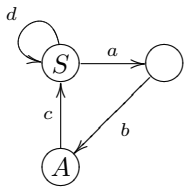
Bruce Merry (BIO 2004)

Introduction

Fred the manic storekeeper has learned about finite state machines (FSMs) in his evening classes. A FSM is a device for string matching. It consists of a finite number of states (hence the name), and a number of *transitions*. A transition takes the form "if in state A and the next character in the string is x, then change to state B". For each pair (A, x) there will be at most one transition, so there are no ambiguities (the machine is said to be *deterministic*).

Two of the states are special: the start state and the accept state. To process a string with a FSM, it is placed in the start state and the characters of the string are fed in one at a time. The transitions determine what state the machine will be in after the character is processed. If the machine is in the accept state after the last character is processed, the machine is said to *accept* the string. If at any point there is no matching transition, the machine "crashes" and does not accept the string.





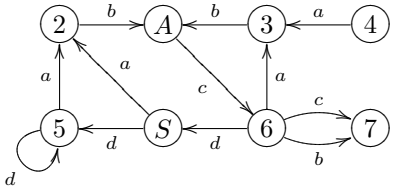
An an example, consider the two machines above. The circles represent states, with S being the start state and A the accept state. The arrows represent the transitions. The first machine will accept only the string *hello*, while the second will accept strings of the form $(abc|d)^*ab$ (where | indicates alternatives and * indicates repetition).

Task

Fred has designed a finite state machine, but since he doesn't think clearly and logically it is very complicated. He wants you to design a new machine that is equivalent to the old one, but as small as possible. Two machines are equivalent if they accept exactly the same strings.

Example

Consider the machine shown below (the numbers refer to the sample input file). Like the second machine above, it will accept strings of the form $(abc|d)^*ab$, but it is far more complicated.



Input (fsm.in)

The first line of input contains the integer N, the number of states in the finite state machine. The states are numbered from 1 to N, with 1 being the start state and N being the accept state. The second line contains K, the number of transitions in the state machine. The remaining K lines describe the transitions, one per line. Each line contains two integers P and Q followed by a lower-case English letter (all separated by single spaces), indicating a transition from state P to state Q via that letter.

Sample input

```
8
12
2 8 b
8 6 c
3 8 b
4 3 a
5 5 d
5 2 a
1 5 d
1 2 a
6 1 d
6 3 a
6 7 b
6 7 c
```

Output (fsm.out)

The output specifies the machine with the fewest states that accepts exactly the same strings as the input machine, in the same format as the input file. If there is more than one smallest answer, you may output any of them.

Sample output

```
3
4
1 2 a
1 1 d
2 3 b
3 1 c
```

Constraints

- $2 \leq N \leq 100$.
- $0 \leq K \leq 26N$.

50% constraints

$N \leq 10$.

Time limit

1 second.

Scoring

If your machine is invalid (e.g. if there are two transitions from a state with the same letter), or if it does not accept the same set of strings as Fred's, you will score 0%. If your machine is correct and as small as possible, you will score 100%. Otherwise, if your machine is correct and smaller than Fred's then you will score 20%.