MANAGER - POJ 1281

https://vjudge.net/problem/poj-1281

One of the programming paradigm in parallel processing is the producer/consumer paradigm that can be implemented using a system with a "manager" process and several "client" processes. The clients can be producers, consumers, etc. The manager keeps a trace of client processes. Each process is identified by its cost that is a strictly positive integer in the range 1 .. 10000. The number of processes with the same cost cannot exceed 10000. The queue is managed according to three types of requests, as follows:

- a x - add to the queue the process with the cost x;
- r - remove a process, if possible, from the queue according to the current manager policy;
- p i - enforce the policy i of the manager, where i is 1 or 2. The default manager policy is 1
- e - ends the list of requests.

There are two manager policies:

- 1 - remove the minimum cost process
- 2 - remove the maximum cost process

The manager will print the cost of a removed process only if the ordinal number of the removed process is in the removal list.

Your job is to write a program that simulates the manager process.

Input

The input is from the standard input. Each data set in the input has the following format:

- the maximum cost of the processes
- the length of the removal list
- the removal list - the list of ordinal numbers of the removed processes that will be displayed; for example 1 4 means that the cost of the first and fourth removed processes will be displayed
- the list of requests each on a separate line.

Each data set ends with an e request. The data sets are separated by empty lines.

Output

The program prints on standard output the cost of each process that is removed, provided that the ordinal number of the remove request is in the list and the queue is not empty at that moment. If the queue is empty the program prints -1. The results are printed on separate lines. An empty line separates the results of different data sets.

An example is given in the following:

Sample Input

```
5
2
1 3
a 2
a 3
r
a 4
p 2
r
a 5
r
e
```

Sample Output

```
2
5
```