C. Heap Operations

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya has recently learned data structure named "Binary heap".

The heap he is now operating with allows the following operations:

- put the given number into the heap;
- get the value of the minimum element in the heap;
- extract the minimum element from the heap;

Thus, at any moment of time the heap contains several integers (possibly none), some of them might be equal.

In order to better learn this data structure Petya took an empty heap and applied some operations above to it. Also, he carefully wrote down all the operations and their results to his event log, following the format:

- `insert` x — put the element with value x in the heap;
- `getMin` x — the value of the minimum element contained in the heap was equal to x;
- `removeMin` — the minimum element was extracted from the heap (only one instance, if there were many).

All the operations were correct, i.e. there was at least one element in the heap each time `getMin` or `removeMin` operations were applied.

While Petya was away for a lunch, his little brother Vova came to the room, took away some of the pages from Petya's log and used them to make paper boats.

Now Vova is worried, if he made Petya's sequence of operations inconsistent. For example, if one apply operations one-by-one in the order they are written in the event log, results of `getMin` operations might differ from the results recorded by Petya, and some of `getMin` or `removeMin` operations may be incorrect, as the heap is empty at the moment they are applied.

Now Vova wants to add some new operation records to the event log in order to make the resulting sequence of operations correct. That is, the result of each `getMin` operation is equal to the result in the record, and the heap is non-empty when `getMin` ad `removeMin` are applied. Vova wants to complete this as fast as possible, as the Petya may get back at any moment. He asks you to add the least possible number of operation records to the current log. Note that arbitrary number of operations may be added at the beginning, between any two other operations, or at the end of the log.

Input

The first line of the input contains the only integer n $(1 \le n \le 100\,000)$ — the number of the records left in Petya's journal.

Each of the following n lines describe the records in the current log in the order they are applied. Format described in the statement is used. All numbers in the input are integers not exceeding 10^9 by their absolute value.

Output

The first line of the output should contain a single integer m — the minimum possible number of records in the modified sequence of operations.

Next m lines should contain the corrected sequence of records following the format of the input (described in the statement), one per line and in the order they are applied. All the numbers in the output should be integers not exceeding 10^9 by their absolute value.

Note that the input sequence of operations must be the **subsequence** of the output sequence.

It's guaranteed that there exists the correct answer consisting of no more than $1\,000\,000$ operations.

Examples

input Copy
```
2
insert 3
getMin 4
```
output Copy
```
4
insert 3
removeMin
insert 4
getMin 4
```

input Copy
```
4
insert 1
insert 1
removeMin
getMin 2
```
output Copy
```
6
insert 1
insert 1
removeMin
removeMin
insert 2
getMin 2
```

Note

In the first sample, after number 3 is inserted into the heap, the minimum number is 3. To make the result of the first `getMin` equal to 4 one should firstly remove number 3 from the heap and then add number 4 into the heap.

In the second sample case number 1 is inserted two times, so should be similarly removed twice.

Codeforces Round #357 (Div. 2)
Finished

→ Virtual participation

Virtual contest is a way to take part in past contest, as close as possible to participation on time. It is supported only ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you - solve these problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you - solve this problem in the archive. Never use someone else's code, read the tutorials or communicate with other person during a virtual contest.

Start virtual contest

→ Problem tags

constructive algorithms | data structures | greedy | *1600

No tag edit access

→ Contest materials

- Announcement (en)
- Tutorial