SoC 2022 Week 1

A. Triple

1 second, 256 megabytes

Given an array a of n elements, print any value that appears at least three times or print `-1` if there is no such value.

Input

The first line contains an integer t ($1 \leq t \leq 10^4$) — the number of test cases.

The first line of each test case contains an integer n ($1 \leq n \leq 2 \cdot 10^5$) — the length of the array.

The second line of each test case contains n integers $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq n$) — the elements of the array.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print any value that appears at least three times or print `-1` if there is no such value.

input

```
7
1
1
3
2 2 2
7
2 2 3 3 4 2 2
8
1 4 3 4 3 2 4 1
9
1 1 1 2 2 2 3 3 3
5
1 5 2 4 3
4
4 4 4 4
```

output

```
-1
2
2
4
3
-1
4
```

In the first test case there is just a single element, so it can't occur at least three times and the answer is `-1`.

In the second test case, all three elements of the array are equal to 2, so 2 occurs three times, and so the answer is 2.

For the third test case, 2 occurs four times, so the answer is 2.

For the fourth test case, 4 occurs three times, so the answer is 4.

For the fifth test case, 1, 2 and 3 all occur at least three times, so they are all valid outputs.

For the sixth test case, all elements are distinct, so none of them occurs at least three times and the answer is `-1`.

B. Permutation Minimization by Deque

2 seconds, 256 megabytes

In fact, the problems E1 and E2 do not have much in common. You should probably think of them as two separate problems.

A permutation p of size n is given. A *permutation* of size n is an array of size n in which each integer from 1 to n occurs exactly once. For example, $[1, 4, 3, 2]$ and $[4, 2, 1, 3]$ are correct permutations while $[1, 2, 4]$ and $[1, 2, 2]$ are not.

Let us consider an empty deque (double-ended queue). A deque is a data structure that supports adding elements to both the beginning and the end. So, if there are elements $[1, 5, 2]$ currently in the deque, adding an element 4 to the beginning will produce the sequence $[4, 1, 5, 2]$, and adding same element to the end will produce $[1, 5, 2, 4]$.

The elements of the permutation are sequentially added to the initially empty deque, starting with p_1 and finishing with p_n. Before adding each element to the deque, you may choose whether to add it to the beginning or the end.

For example, if we consider a permutation $p = [3, 1, 2, 4]$, one of the possible sequences of actions looks like this:

1.	add 3 to the end of the deque:	deque has a sequence $[3]$ in it;
2.	add 1 to the beginning of the deque:	deque has a sequence $[1, 3]$ in it;
3.	add 2 to the end of the deque:	deque has a sequence $[1, 3, 2]$ in it;
4.	add 4 to the end of the deque:	deque has a sequence $[1, 3, 2, 4]$ in it;

Find the lexicographically smallest possible sequence of elements in the deque after the entire permutation has been processed.

A sequence $[x_1, x_2, \ldots, x_n]$ is *lexicographically smaller* than the sequence $[y_1, y_2, \ldots, y_n]$ if there exists such $i \leq n$ that $x_1 = y_1$, $x_2 = y_2$, $\ldots$, $x_{i-1} = y_{i-1}$ and $x_i < y_i$. In other words, if the sequences x and y have some (possibly empty) matching prefix, and the next element of the sequence x is strictly smaller than the corresponding element of the sequence y. For example, the sequence $[1, 3, 2, 4]$ is smaller than the sequence $[1, 3, 4, 2]$ because after the two matching elements $[1, 3]$ in the start the first sequence has an element 2 which is smaller than the corresponding element 4 in the second sequence.

Input

The first line contains an integer t ($1 \leq t \leq 1000$) — the number of test cases.

The next $2t$ lines contain descriptions of the test cases.

The first line of each test case description contains an integer n ($1 \leq n \leq 2 \cdot 10^5$) — permutation size. The second line of the description contains n space-separated integers p_i ($1 \leq p_i \leq n$; all p_i are all unique) — elements of the permutation.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

Print t lines, each line containing the answer to the corresponding test case. The answer to a test case should contain n space-separated integer numbers — the elements of the lexicographically smallest permutation that is possible to find in the deque after executing the described algorithm.

input

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

output

```
1 3 2 4
1 2 3
1 3 2
1 2
1 2
```

One of the ways to get a lexicographically smallest permutation $[1, 3, 2, 4]$ from the permutation $[3, 1, 2, 4]$ (the first sample test case) is described in the problem statement.

C. Remove Duplicates

1 second, 256 megabytes

Petya has an array a consisting of n integers. He wants to remove duplicate (equal) elements.

Petya wants to leave only the rightmost entry (occurrence) for each element of the array. The relative order of the remaining unique elements should not be changed.

Input

The first line contains a single integer n ($1 \le n \le 50$) — the number of elements in Petya's array.

The following line contains a sequence $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 1\,000$) — the Petya's array.

Output

In the first line print integer x — the number of elements which will be left in Petya's array after he removed the duplicates.

In the second line print x integers separated with a space — Petya's array after he removed the duplicates. For each unique element only the rightmost entry should be left.

input

```
6
1 5 5 1 6 1
```

output

```
3
5 6 1
```

input

```
5
2 4 2 4 4
```

output

```
2
2 4
```

input

```
5
6 6 6 6 6
```

output

```
1
6
```

In the first example you should remove two integers 1, which are in the positions 1 and 4. Also you should remove the integer 5, which is in the position 2.

In the second example you should remove integer 2, which is in the position 1, and two integers 4, which are in the positions 2 and 4.

In the third example you should remove four integers 6, which are in the positions 1, 2, 3 and 4.

D. Second Order Statistics

2 seconds, 256 megabytes

Once Bob needed to find the second order statistics of a sequence of integer numbers. Lets choose each number from the sequence exactly once and sort them. The value on the second position is the second order statistics of the given sequence. In other words it is the smallest element strictly greater than the minimum. Help Bob solve this problem.

Input

The first input line contains integer n ($1 \le n \le 100$) — amount of numbers in the sequence. The second line contains n space-separated integer numbers — elements of the sequence. These numbers don't exceed 100 in absolute value.

Output

If the given sequence has the second order statistics, output this order statistics, otherwise output `NO`.

input

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

output

```
1
```

input

```
5
1 2 3 1 1
```

output

```
2
```

E. Glass Carving

2 seconds, 256 megabytes

Leonid wants to become a glass carver (the person who creates beautiful artworks by cutting the glass). He already has a rectangular w mm $\times$ h mm sheet of glass, a diamond glass cutter and lots of enthusiasm. What he lacks is understanding of what to carve and how.

In order not to waste time, he decided to practice the technique of carving. To do this, he makes vertical and horizontal cuts through the entire sheet. This process results in making smaller rectangular fragments of glass. Leonid does not move the newly made glass fragments. In particular, a cut divides each fragment of glass that it goes through into smaller fragments.

After each cut Leonid tries to determine what area the largest of the currently available glass fragments has. Since there appear more and more fragments, this question takes him more and more time and distracts him from the fascinating process.

Leonid offers to divide the labor — he will cut glass, and you will calculate the area of the maximum fragment after each cut. Do you agree?

Input

The first line contains three integers w, h, n ($2 \le w, h \le 200\,000$, $1 \le n \le 200\,000$).

Next n lines contain the descriptions of the cuts. Each description has the form $H\ y$ or $V\ x$. In the first case Leonid makes the horizontal cut at the distance y millimeters ($1 \le y \le h - 1$) from the lower edge of the original sheet of glass. In the second case Leonid makes a vertical cut at distance x ($1 \le x \le w - 1$) millimeters from the left edge of the original sheet of glass. It is guaranteed that Leonid won't make two identical cuts.

Output

After each cut print on a single line the area of the maximum available glass fragment in mm^2.

input

```
4 3 4
H 2
V 2
V 3
V 1
```

output

```
8
4
4
2
```

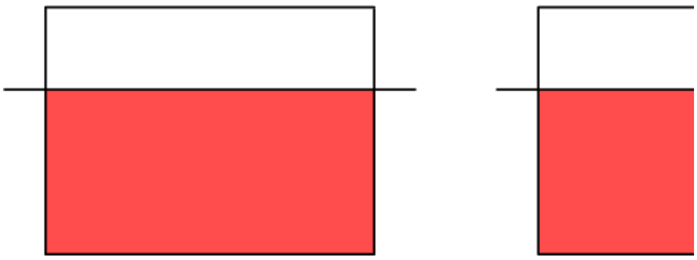
input

```
7 6 5
H 4
V 3
V 5
H 2
V 1
```

output

```
28
16
12
6
4
```

Picture for the first sample test:



Picture for the second sample test: