A. Subset Mex

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

Given a set of integers (it can contain equal elements).

You have to split it into two subsets A and B (both of them can contain equal elements or be empty). You have to maximize the value of $mex(A) + mex(B)$.

Here mex of a set denotes the smallest non-negative integer that doesn't exist in the set. For example:

- $mex(\{1, 4, 0, 2, 2, 1\}) = 3$
- $mex(\{3, 3, 2, 1, 3, 0, 0\}) = 4$
- $mex(\varnothing) = 0$ (mex for empty set)

The set is splitted into two subsets A and B if for any integer number x the number of occurrences of x into this set is equal to the sum of the number of occurrences of x into A and the number of occurrences of x into B.

Input

The input consists of multiple test cases. The first line contains an integer t ($1 \leq t \leq 100$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains an integer n ($1 \leq n \leq 100$) — the size of the set.

The second line of each testcase contains n integers $a_1, a_2, \dots a_n$ ($0 \leq a_i \leq 100$) — the numbers in the set.

Output

For each test case, print the maximum value of $mex(A) + mex(B)$.

Example

input

```
4
6
0 2 1 5 0 1
3
0 1 2
4
0 2 0 1
6
1 2 3 4 5 6
```

output

```
5
3
4
0
```

Note

In the first test case, $A = \{0, 1, 2\}, B = \{0, 1, 5\}$ is a possible choice.

In the second test case, $A = \{0, 1, 2\}, B = \varnothing$ is a possible choice.

In the third test case, $A = \{0, 1, 2\}, B = \{0\}$ is a possible choice.

In the fourth test case, $A = \{1, 3, 5\}, B = \{2, 4, 6\}$ is a possible choice.

B. Maximum Product

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given an array of integers $a_1, a_2, \ldots, a_n$. Find the maximum possible value of $a_i a_j a_k a_l a_t$ among all five indices (i, j, k, l, t) ($i < j < k < l < t$).

Input

The input consists of multiple test cases. The first line contains an integer t ($1 \le t \le 2 \cdot 10^4$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer n ($5 \le n \le 10^5$) — the size of the array.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($-3 \times 10^3 \le a_i \le 3 \times 10^3$) — given array.

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

Output

For each test case, print one integer — the answer to the problem.

Example

input

```
4
5
-1 -2 -3 -4 -5
6
-1 -2 -3 1 2 -1
6
-1 0 0 0 -1 -1
6
-9 -7 -5 -3 -2 1
```

output

```
-120
12
0
945
```

Note

In the first test case, choosing a_1, a_2, a_3, a_4, a_5 is a best choice: $(-1) \cdot (-2) \cdot (-3) \cdot (-4) \cdot (-5) = -120$.

In the second test case, choosing a_1, a_2, a_3, a_5, a_6 is a best choice: $(-1) \cdot (-2) \cdot (-3) \cdot 2 \cdot (-1) = 12$.

In the third test case, choosing a_1, a_2, a_3, a_4, a_5 is a best choice: $(-1) \cdot 0 \cdot 0 \cdot 0 \cdot (-1) = 0$.

In the fourth test case, choosing a_1, a_2, a_3, a_4, a_6 is a best choice: $(-9) \cdot (-7) \cdot (-5) \cdot (-3) \cdot 1 = 945$.

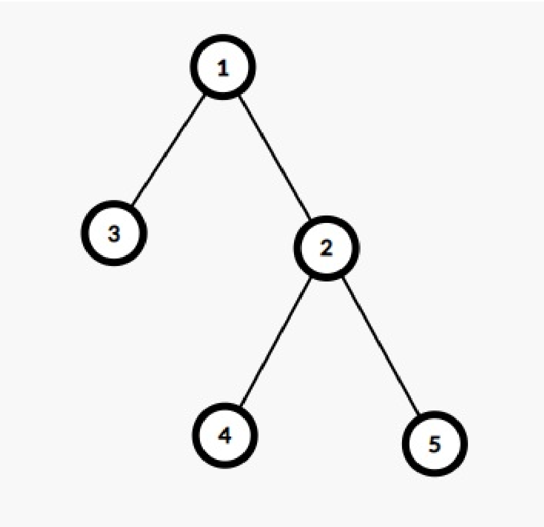
C. Link Cut Centroids

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

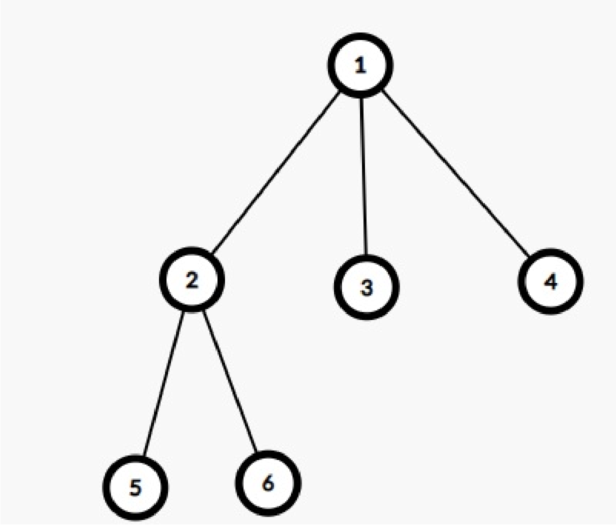
Fishing Prince loves trees, and he especially loves trees with only one centroid. The tree is a connected graph without cycles.

A vertex is a **centroid** of a tree only when you cut this vertex (remove it and remove all edges from this vertex), the size of the largest connected component of the remaining graph is the smallest possible.

For example, the centroid of the following tree is 2, because when you cut it, the size of the largest connected component of the remaining graph is 2 and it can't be smaller.



However, in some trees, there might be more than one centroid, for example:



Both vertex 1 and vertex 2 are centroids because the size of the largest connected component is 3 after cutting each of them.

Now Fishing Prince has a tree. He should cut one edge of the tree (it means to remove the edge). After that, he should add one edge. The resulting graph after these two operations should be a tree. He can add the edge that he cut.

He wants the centroid of the resulting tree to be unique. Help him and find any possible way to make the operations. It can be proved, that at least one such way always exists.

Input

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

The first line of each test case contains an integer n ($3 \leq n \leq 10^5$) — the number of vertices.

Each of the next $n-1$ lines contains two integers x, y ($1 \leq x, y \leq n$). It means, that there exists an edge connecting vertices x and y.

It's guaranteed that the given graph is a tree.

It's guaranteed that the sum of n for all test cases does not exceed 10^5.

Output

For each test case, print two lines.

In the first line print two integers x_1, y_1 ($1 \leq x_1, y_1 \leq n$), which means you cut the edge between vertices x_1 and y_1. There should exist edge connecting vertices x_1 and y_1.

In the second line print two integers x_2, y_2 ($1 \leq x_2, y_2 \leq n$), which means you add the edge between vertices x_2 and y_2.

The graph after these two operations should be a tree.

If there are multiple solutions you can print any.

Example

input

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

output

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

Note

Note that you can add the same edge that you cut.

In the first test case, after cutting and adding the same edge, the vertex 2 is still the only centroid.

In the second test case, the vertex 2 becomes the only centroid after cutting the edge between vertices 1 and 3 and adding the edge between vertices 2 and 3.

D. Three Sequences

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given a sequence of n integers $a_1, a_2, \ldots, a_n$.

You have to construct two sequences of integers b and c with length n that satisfy:

- for every i ($1 \leq i \leq n$) $b_i + c_i = a_i$
- b is non-decreasing, which means that for every $1 < i \leq n$, $b_i \geq b_{i-1}$ must hold
- c is non-increasing, which means that for every $1 < i \leq n$, $c_i \leq c_{i-1}$ must hold

You have to minimize $\max(b_i, c_i)$. In other words, you have to minimize the maximum number in sequences b and c.

Also there will be q changes, the i-th change is described by three integers l, r, x. You should add x to $a_l, a_{l+1}, \ldots, a_r$.

You have to find the minimum possible value of $\max(b_i, c_i)$ for the initial sequence and for sequence after each change.

Input

The first line contains an integer n ($1 \leq n \leq 10^5$).

The secound line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq i \leq n$, $-10^9 \leq a_i \leq 10^9$).

The third line contains an integer q ($1 \leq q \leq 10^5$).

Each of the next q lines contains three integers l, r, x ($1 \leq l \leq r \leq n, -10^9 \leq x \leq 10^9$), desribing the next change.

Output

Print $q + 1$ lines.

On the i-th ($1 \leq i \leq q + 1$) line, print the answer to the problem for the sequence after $i - 1$ changes.

Examples

input

```
4
2 -1 7 3
2
2 4 -3
3 4 2
```

output

```
5
5
6
```

input

```
6
-9 -10 -9 -6 -5 4
3
2 6 -9
1 2 -10
4 6 -3
```

output

```
3
3
3
1
```

input

```
1
0
2
1 1 -1
1 1 -1
```

output

```
0
0
-1
```

Note

In the first test:

- The initial sequence $a = (2, -1, 7, 3)$. Two sequences $b = (-3, -3, 5, 5), c = (5, 2, 2, -2)$ is a possible choice.
- After the first change $a = (2, -4, 4, 0)$. Two sequences $b = (-3, -3, 5, 5), c = (5, -1, -1, -5)$ is a possible choice.

- After the second change $a = (2, -4, 6, 2)$. Two sequences $b = (-4, -4, 6, 6), c = (6, 0, 0, -4)$ is a possible choice.

E. Deleting Numbers

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

This is an interactive problem.

There is an unknown integer x ($1 \le x \le n$). You want to find x.

At first, you have a set of integers $\{1, 2, \ldots, n\}$. You can perform the following operations no more than 10000 times:

- A a: find how many numbers are multiples of a in the current set.
- B a: find how many numbers are multiples of a in this set, and then delete all multiples of a, but x will never be deleted (even if it is a multiple of a). In this operation, a must be greater than 1.
- C a: it means that you know that $x = a$. This operation can be only performed once.

Remember that in the operation of type B $a > 1$ must hold.

Write a program, that will find the value of x.

Input

The first line contains one integer n ($1 \le n \le 10^5$). The remaining parts of the input will be given throughout the interaction process.

Interaction

In each round, your program needs to print a line containing one uppercase letter A, B or C and an integer a ($1 \le a \le n$ for operations A and C, $2 \le a \le n$ for operation B). This line desribes operation you make.

If your operation has type C your program should terminate immediately.

Else your program should read one line containing a single integer, which is the answer to your operation.

After outputting each line, don't forget to flush the output. To do it use:

- fflush(stdout) in C/C++;
- System.out.flush() in Java;
- sys.stdout.flush() in Python;
- flush(output) in Pascal;
- See the documentation for other languages.

It is guaranteed, that the number x is fixed and won't change during the interaction process.

Hacks:

To make a hack, use such input format:

The only line should contain two integers n, x ($1 \le x \le n \le 10^5$).

Example

input

```
10

2

4

0
```

output

```

B 4

A 2

A 8

C 4
```

Note

Note that to make the sample more clear, we added extra empty lines. You shouldn't print any extra empty lines during the interaction process.

In the first test $n = 10$ and $x = 4$.

Initially the set is: $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$.

In the first operation, you ask how many numbers are multiples of 4 and delete them. The answer is 2 because there are two numbers divisible by 4: $\{4, 8\}$. 8 will be deleted but 4 won't, because the number x will never be deleted. Now the set is $\{1, 2, 3, 4, 5, 6, 7, 9, 10\}$.

In the second operation, you ask how many numbers are multiples of 2. The answer is 4 because there are four numbers divisible by 2: $\{2, 4, 6, 10\}$.

In the third operation, you ask how many numbers are multiples of 8. The answer is 0 because there isn't any number divisible by 8 in the current set.

In the fourth operation, you know that $x = 4$, which is the right answer.