

# Yet Another Maximize Permutation Subarrays

Input file:            **standard input**  
Output file:          **standard output**  
Time limit:          1.5 seconds  
Memory limit:        256 megabytes

You are given a permutation  $p$  of size  $n$ . You want to maximize the number of subarrays of  $p$  that are permutations. In order to do so, you must perform the following operation exactly once:

- Select integers  $i, j$ , where  $1 \leq i, j \leq n$ , then
- Swap  $p_i$  and  $p_j$ .

For example, if  $p = [5, 1, 4, 2, 3]$  and we choose  $i = 2, j = 3$ , the resulting array will be  $[5, 4, 1, 2, 3]$ . If instead we choose  $i = j = 5$ , the resulting array will be  $[5, 1, 4, 2, 3]$ .

Which choice of  $i$  and  $j$  will maximize the number of subarrays that are permutations?

## NOTE:

- A permutation of length  $n$  is an array of  $n$  distinct integers from 1 to  $n$  in arbitrary order. For example,  $[2, 3, 1, 5, 4]$  is a permutation, but  $[1, 2, 2]$  is not a permutation (2 appears twice in the array), and  $[1, 3, 4]$  is also not a permutation ( $n = 3$  but there is 4 in the array).
- An array  $a$  is a subarray of an array  $b$  if  $a$  can be obtained from  $b$  by deleting several (possibly, zero or all) elements from the beginning and several (possibly, zero or all) elements from the end.

## Input

The first line of the input contains a single integer  $t$  ( $1 \leq t \leq 10$ ) — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer  $n$  ( $1 \leq n \leq 10^6$ ) — the size of the permutation.

The next line of each test case contains  $n$  integers  $p_1, p_2, \dots, p_n$  ( $1 \leq p_i \leq n$ , all  $p_i$  are distinct) — the elements of the permutation  $p$ .

## Output

For each test case, output two integers  $i$  and  $j$  ( $1 \leq i, j \leq n$ ) — the indices to swap in  $p$ .

If there are multiple solutions, print any of them.

## Example

standard input	standard output
8	3 3
3	1 2
1 2 3	1 4
3	1 3
1 3 2	9 9
5	4 9
1 3 2 5 4	2 4
6	1 5
4 5 6 1 2 3	
9	
8 7 6 3 2 1 4 5 9	
10	
7 10 5 1 9 8 3 2 6 4	
10	
8 5 10 9 2 1 3 4 6 7	
10	
2 3 5 7 10 1 8 6 4 9	