C. Insertion Sort

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya is a beginner programmer. He has already mastered the basics of the C++ language and moved on to learning algorithms. The first algorithm he encountered was insertion sort. Petya has already written the code that implements this algorithm and sorts the given integer zero-indexed array a of size n in the non-decreasing order.

```
for (int i = 1; i < n; i = i + 1)
{
   int j = i;
   while (j > 0 && a[j] < a[j - 1])
   {
      swap(a[j], a[j - 1]); // swap elements a[j] and a[j - 1]
      j = j - 1;
   }
}
```

Petya uses this algorithm only for sorting of arrays that are permutations of numbers from 0 to $n - 1$. He has already chosen the permutation he wants to sort but he first decided to swap some two of its elements. Petya wants to choose these elements in such a way that the number of times the sorting executes function `swap`, was minimum. Help Petya find out the number of ways in which he can make the swap and fulfill this requirement.

It is guaranteed that it's always possible to swap two elements of the input permutation in such a way that the number of `swap` function calls decreases.

Input

The first line contains a single integer n $(2 \leq n \leq 5000)$ — the length of the permutation. The second line contains n different integers from 0 to $n - 1$, inclusive — the actual permutation.

Output

Print two integers: the minimum number of times the `swap` function is executed and the number of such pairs (i, j) that swapping the elements of the input permutation with indexes i and j leads to the minimum number of the executions.

Examples

input

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

output

```
3 2
```

input

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

output

```
3 4
```

Note

In the first sample the appropriate pairs are $(0, 3)$ and $(0, 4)$.

In the second sample the appropriate pairs are $(0, 4)$, $(1, 4)$, $(2, 4)$ and $(3, 4)$.