Problem 1 – Sequences

You are given an integer array **arr**, consisting of **N** integers. Find the number of non-decreasing consecutive subsequences in **arr**. Every sequence starts after the previous one. For example: if the array **arr** consists of the numbers *1, 2, -3, 4, 4, 0, 1,* the number of non-decreasing consecutive subsequences is 3 (the first is *1, 2*, the second is *-3, 4, 4* and the third is *0, 1*)

Your task is to write a JavaScript method named “**Solve**” that solves the problem.

Input

The method **Solve** accepts a zero-based array of strings. Each of the string represents an integer. Element 0 of the array is the number N. Next N elements (from 1 to N) construct the array **arr**.

Output

Your method should return a single number - the number of non-decreasing consecutive subsequences.

Example code

```
function Solve(params) {
      var N = parseInt(params[0]);
      var answer = 0;
      // Your code here...
      return answer;
}
```

Constraints

- **N** will be between 1 and 10 000.
- Each element of **arr** will be between -2 000 000 000 and +2 000 000 000.
- Allowed working time for your program: 0.1 seconds. Allowed memory: 16 MB.

Examples (each line represents an element from the only argument of Solve)

Example input	Example output
9 1 8 8 7 6 5 7 7 6	5

Решение на задачата 100/100 за 0,171 секунди

```
function solve(params) {
    var n = +params[0];
    //construct array
    var array = [],
        i;
    for (i = 1; i <= n; i++) {

         array.push(+params[i]);
    }
    // count seq
    var count = 1,
        len = array.length,
        s;
    for (s = 0; s < len; s++) {
        if (array[s] > array[s + 1]) { count++; }
    }



    return count;
}
var a = [9, 1, 8, 8, 7, 6, 5, 7, 7, 6];
console.log(solve(a));
```