02x18 - Disciplined Pairs

Lets define a disciplined Array.
An array of length at least 2 having distinct integers is said to be disciplined if and only if the second highest element lies strictly to the left of the highest value. For example, [1, 2, 13, 10, 15] is disciplined as the second-highest value 13 lies to the left of highest value 15.
For every disciplined array, we define a disciplined pair **(a, b)** where **a** denotes the index of the second-highest value (1-indexed) and **b** denotes the index of the highest value (1-indexed). In the above array, the fantabulous pair is (3, 5) as arr[3] = 13 and arr[5] = 15.
Your task is to solve the following problem. Given an array, find the total number of distinct disciplined pairs over all its subarrays.

INPUT

The first line contains an integer **N** *(1 <= N <= 10^6)* denoting the length of the array.
The next line contains **N** distinct integers denoting the elements of the array, all of which will be greater than 0 and less than 10^9.
It is guaranteed that all array elements will be distinct.

OUTPUT

Output a single integer which is the answer to the problem.

Sample Input 0

```
4
1 3 2 4
```

Sample Output 0

```
3
```

Explanation 0

Let us consider all the subarrays of the given array.
The subarray [1] is not disciplined.
The subarray [2] is not disciplined.
The subarray [3] is not disciplined.
The subarray [4] is not disciplined.
The disciplined pair for subarray [1, 3] is (1, 2).
The subarray [3, 2] is not disciplined.
The disciplined pair for subarray [2, 4] is (1, 2).
The subarray [1, 3, 2] is not disciplined.
The disciplined pair for subarray [3, 2, 4] is (1, 3).
The disciplined pair for subarray [1, 3, 2, 4] is (2, 4).
So, there are the 3 distinct disciplined pairs, which are (1, 2), (1, 3) and (2, 4).

Current Buffer (saved locally, editable)

C++14

```cpp
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;


int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}

```

Line: 1 Col: 1

Upload Code as File

Test against custom input

Run Code

Submit Code