**Little Shino and Pairs**

Attempted by: **956**

/

Accuracy: **9%**

/

Maximum Score: **20**

/

8 Votes

Tag(s):

Data Structures, Easy, Stack

**PROBLEM**

**EDITORIAL**

**MY SUBMISSIONS**

**ANALYTICS**

Given a permutation of number from 11 to NN. Among all the subarrays, find the number of unique pairs (a,b)(a,b) such that a≠ba≠b and aa is maximum and bb is second maximum in that subarray.

**Input:**  
First line contains an integer, NN (1≤N≤105)(1≤N≤105). Second line contains NN space separated distinct integers, AiAi (1≤Ai≤N)(1≤Ai≤N), denoting the permutation.

**Output:**  
Print the required answer.

**SAMPLE INPUT**

5

1 2 3 4 5

**SAMPLE OUTPUT**

4

**Explanation**

All the possible subarrays are:

11  
1212  
123123  
12341234  
1234512345  
22  
2323  
234234  
23452345  
33  
3434  
345345  
44  
4545  
55

The 44 unique pairs are:  
(2,1)(2,1)  
(3,2)(3,2)  
(4,3)(4,3)  
(5,4)(5,4)

**Time Limit:**1.0 sec(s) for each input file.

**Memory Limit:**256 MB

**Source Limit:**1024 KB

**Marking Scheme:**Marks are awarded when all the testcases pass.

**Allowed Languages:**C, C++, Clojure, C#, D, Erlang, F#, Go, Groovy, Haskell, Java, Java 8, JavaScript(Rhino), JavaScript(Node.js), Lisp, Lisp (SBCL), Lua, Objective-C, OCaml, Octave, Pascal, Perl, PHP, Python, Python 3, R(RScript), Racket, Ruby, Rust, Scala, Scala 2.11.8, Swift, Visual Basic

<https://www.hackerearth.com/practice/data-structures/stacks/basics-of-stacks/practice-problems/algorithm/little-shino-and-pairs/description/>

using System;

using System.Collections.Generic;

using System.Linq;

namespace HackerEarth

{

class Program

{

static void Main(string[] args)

{

int num = int.Parse(Console.ReadLine());

string[] valueStrings = Console.ReadLine().Split(' ');

int[] values = new int[num];

for (int i = 0; i < num; i++) { values[i] = int.Parse(valueStrings[i]); }

int secondMax = 0;

int max = values[0];

int count = 0;

for (int i = 0; i < values.Length - 1; i++)

{

max = values[i];

secondMax = 0;

for (int j = i + 1; j < values.Length; j++)

{

if (values[j] > max)

{

if (values[i] != max) break;

max = values[j];

secondMax = values[i];

count++;

}

else if (values[j] > secondMax)

{

if (values[i] != max) break;

secondMax = values[j];

count++;

}

}

}

Console.WriteLine(count);

}

}

}