

CodeCheck Report: trainingHM94EN-HNA

Test Name:

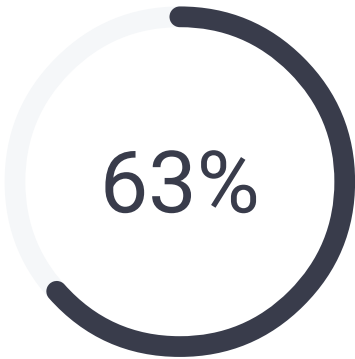
[Check out Codility training tasks](#)

Summary   Timeline    AI Assistant Transcript

Tasks summary

Task	Time spent	Score
ArrayInversionCount C#	2 min	63%

Total score



Tasks Details

Medium	1.	Task Score	Correctness	Performance	
	<b>ArrayInversionCount</b> Compute number of inversion in an array.			100%	20%

Task description

An array A consisting of N integers is given. An *inversion* is a pair of indexes (P, Q) such that P < Q and A[Q] < A[P].

Write a function:

```
class Solution { public int solution(int[] A); }
```

that computes the number of inversions in A, or returns -1 if it exceeds 1,000,000,000.

For example, in the following array:

A[0] = -1 A[1] = 6 A[2] = 3  
A[3] = 4 A[4] = 7 A[5] = 4



there are four inversions:

(1,2) (1,3) (1,5) (4,5)

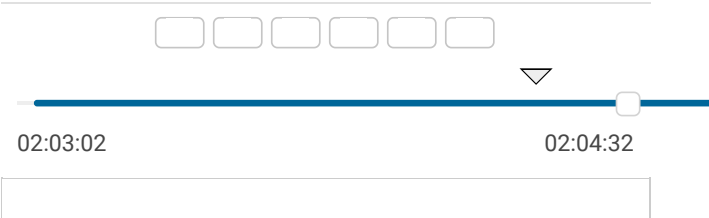
so the function should return 4.

Write an **efficient** algorithm for the following assumptions:

Solution

Programming language used:	C#	
Total time used:	2 minutes	
Effective time used:	2 minutes	
Notes:	not defined yet	

Task timeline



- N is an integer within the range [0..100,000];
- each element of array A is an integer within the range [-2,147,483,648..2,147,483,647].

Copyright 2009–2024 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

Code: 02:04:32 UTC, cs, final, [show code in pop-up](#)  
score: 63

```
1 using System;
2 // you can also use other imports, for example:
3 // using System.Collections.Generic;
4
5 // you can write to stdout for debugging purposes,
6 // Console.WriteLine("this is a debug message");
7
8 class Solution {
9     public int solution(int[] A) {
10         // Implement your solution here
11         int inversionsCount = 0;
12
13         for (int i = 0; i < A.Length; i++)
14         {
15             for (int j = i+1; j < A.Length; j++)
16             {
17                 if (A[i] > A[j])
18                 {
19                     inversionsCount++;
20                 }
21             }
22         }
23
24         return inversionsCount;
25     }
26 }
```

Analysis summary

The following issues have been detected: timeout errors.

Analysis

Detected time complexity: **O(N\*\*2)**

expand all	Example tests	
▶	example1 example test	✓ OK
expand all	Correctness tests	
▶	simple1	✓ OK
▶	simple2	✓ OK
▶	simple3	✓ OK
▶	extreme_0_inv [0], [], [1,2,3], [1,1,1]	✓ OK
▶	medium1 n=100	✓ OK
▶	medium2 n=200	✓ OK
expand all	Performance tests	
▶	medium3 n=1000	✓ OK
▶		

big1 n=10000	<b>✗ TIMEOUT ERROR</b> running time: 0.188 sec., time limit: 0.100 sec.
▶ big2 n=20000	<b>✗ TIMEOUT ERROR</b> running time: 0.732 sec., time limit: 0.112 sec.
▶ big3 n=30000	<b>✗ TIMEOUT ERROR</b> running time: 1.724 sec., time limit: 0.128 sec.
▶ big_monotonic long descending and non-ascending sequence	<b>✗ TIMEOUT ERROR</b> running time: 1.332 sec., time limit: 0.144 sec.