Codility_

CodeCheck Report: trainingU9FG36-N7E

Test Name:

Check out Codility training tasks

Summary Timeline

Tasks summary

Task	Time spent	Score
OddOccurrencesInArray C#	1 min	100%

Total score

Tasks Details

1. OddOccurrencesInArray
Easy
Find value that occurs in odd number of elements.

Task Score	Correctness	Performance
100%	100%	100%

Task description

A non-empty array A consisting of N integers is given. The array contains an odd number of elements, and each element of the array can be paired with another element that has the same value, except for one element that is left unpaired.

For example, in array A such that:

```
A[0] = 9  A[1] = 3  A[2] = 9
A[3] = 3  A[4] = 9  A[5] = 7
A[6] = 9
```

- the elements at indexes 0 and 2 have value 9,
- the elements at indexes 1 and 3 have value 3,
- the elements at indexes 4 and 6 have value 9,
- the element at index 5 has value 7 and is unpaired.

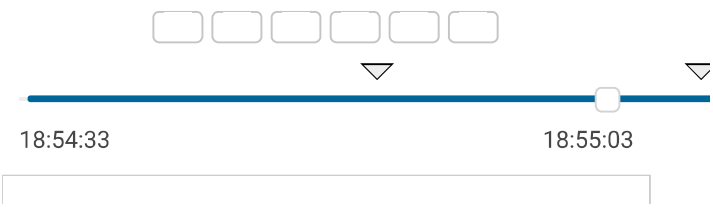
Write a function:

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

Solution

Programming language used:	C#
Total time used:	1 minutes
Effective time used:	1 minutes
Notes:	*not defined yet*

Task timeline



that, given an array A consisting of N integers fulfilling the above conditions, returns the value of the unpaired element.

For example, given array A such that:

```
A[0] = 9  A[1] = 3  A[2] = 9
A[3] = 3  A[4] = 9  A[5] = 7
A[6] = 9
```

the function should return 7, as explained in the example above.

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

- N is an odd integer within the range [1..1,000,000];
- each element of array A is an integer within the range [1..1,000,000,000];
- all but one of the values in A occur an even number of times.

Code: 18:55:02 UTC, cs, final, score: **100** show code in pop-up

```
1  using System;
2  using System.Linq;
3  using System.Collections.Generic;
4
5  /* Lesson 2.2 - Odd Occurrences in Array
6   * Paulo Santos
7   * 24.Nov.2022
8   */
9  class Solution {
10     public int solution(int[] A) {
11
12                 /*
13                  * Check the input
14                  */
15                 if (A == null)
16                         throw new ArgumentNullExce
17
18         var dic = new Dictionary<int, int>(); // c
19         foreach(var n in A) {
20             if (dic.ContainsKey(n))
21                 dic[n]++;
22             else
23                 dic[n] = 1;
24         }
25
26                 /*
27                  * Return the one that
28                  * has an odd number of items.
29                  */
30         return dic.Where(x => (x.Value % 2) == 1)
31                   .Select(x => x.Key)
32                   .Single();
33     }
34 }
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: **O(N) or O(N*log(N))**

expand all	Example tests	
▶	example1 example test	✓ OK
expand all	Correctness tests	
▶	simple1 simple test n=5	✓ OK
▶	simple2 simple test n=11	✓ OK
▶	extreme_single_item [42]	✓ OK

▶	small1 small random test n=201	✓ OK
▶	small2 small random test n=601	✓ OK

expand all Performance tests

▶	medium1 medium random test n=2,001	✓ OK
▶	medium2 medium random test n=100,003	✓ OK
▶	big1 big random test n=999,999, multiple repetitions	✓ OK
▶	big2 big random test n=999,999	✓ OK