Codility_

Screen Report: Anonymous
Test Name:

Summary   Timeline

Tasks summary

Task		Effective time spent	Score
OddOccurrencesInArray C++	⚠️	6 min	100%

Total score

100%

Tasks Details

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

Easy

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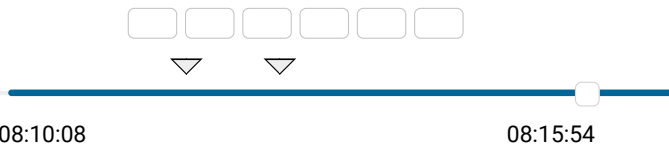
- the elements at indexes 0 and 2 have value 9,
- the elements at indexes 1 and 3 have value 3,
- the elements at indexes 4 and 6 have value 9,
- the element at index 5 has value 7 and is unpaired.

Write a function:

Solution

Programming language used:	C++
Time spent on task:	6 minutes
Notes:	_not defined yet_

Task timeline

08:10:08   08:15:54

```
int solution(vector<int> &A);
```

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: 08:15:53 UTC, cpp, final, score: **100**   show code in pop-up

```
1   // you can use includes, for example:
2   // #include <algorithm>
3
4   // you can write to stdout for debugging purposes,
5   // cout << "this is a debug message" << endl;
6
7   int solution(vector<int> &A) {
8       // Implement your solution here
9
10      int unpairedResult{0};
11
12      for (auto element : A)
13      {
14          unpairedResult ^= element;
15      }
16      return unpairedResult;
17  }
```

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   ✓ **OK**
big random test n=999,999