Codility_

Candidate Report: Anonymous

Check out Codility training tasks

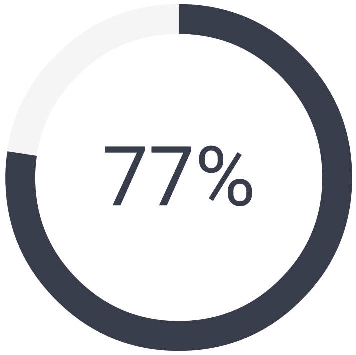
Test Name:

Summary Timeline Feedback

Tasks summary

Task	Time spent	Score
OddOccurrencesInArray JavaScript	13 min	77%

Total score



Tasks Details

Easy

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

Task Score	Correctness	Performance
77%	100%	50%

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:

```
    function solution(A);
```

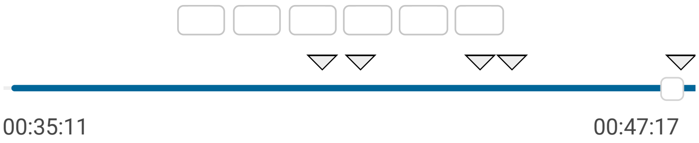
that, given an array A consisting of N integers fulfilling the above conditions, returns the value of the unpaired element.

For example, given array A such that:

Solution

Programming language used:	JavaScript
Total time used:	13 minutes
Effective time used:	13 minutes
Notes:	*not defined yet*

Task timeline



Code: 00:47:17 UTC, js, final, score: 77

show code in pop-up

```
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.

```
// you can write to stdout for debugging purposes, e.g
// console.log('this is a debug message');

function solution(A) {
    // write your code in JavaScript (Node.js 8.9.4)
    let occ = {};
    for (i=0; i<A.length; i++) {
        const item=A[i];
        if (occ[item]) { occ[item]++; }
        else { occ[item]=1; }

        if (A.length-1 === i) {
            const keys = Object.keys(occ);
            const unique = keys.filter(key => {
                return(occ[key]%2 !== 0);
            });
            return parseInt(unique[0]);
        }
    }
}
```

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 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	✘ TIMEOUT ERROR running time: 1.372 sec., time limit: 1.328 sec.
▶ big2 big random test n=999,999	✘ TIMEOUT ERROR running time: 1.988 sec., time limit: 1.472 sec.

The PDF version of this report that may be downloaded on top of this site may contain sensitive data including personal information. For security purposes, we recommend you remove it from your system once reviewed.