Codility_

CodeCheck Report: training6CDQ3T-J4J

Test Name:

Check out Codility training tasks

Summary Timeline

Tasks summary

Task		Time spent	Score
FirstUnique Python	⚠	8 min	100%

Total score

100%

Tasks Details

Easy

1.
FirstUnique
Find the first unique number in a given sequence.

Task Score	Correctness	Performance
100%	100%	100%

Task description

A non-empty array A consisting of N integers is given. The *unique number* is the number that occurs exactly once in array A.

For example, the following array A:

```
A[0] = 4
A[1] = 10
A[2] = 5
A[3] = 4
A[4] = 2
A[5] = 10
```

contains two unique numbers (5 and 2).

Solution

Programming language used:	Python	
Total time used:	8 minutes	?
Effective time used:	8 minutes	?
Notes:	*not defined yet*	

Task timeline ?

You should find the first unique number in A. In other words, find the unique number with the lowest position in A.

For above example, 5 is in second position (because A[2] = 5) and 2 is in fourth position (because A[4] = 2). So, the first unique number is 5.

Write a function:

```
def solution(A)
```

that, given a non-empty array A of N integers, returns the first unique number in A. The function should return −1 if there are no unique numbers in A.

For example, given:

```
A[0] = 1
A[1] = 4
A[2] = 3
A[3] = 3
A[4] = 1
A[5] = 2
```

the function should return 4. There are two unique numbers (4 and 2 occur exactly once). The first one is 4 in position 1 and the second one is 2 in position 5. The function should return 4 bacause it is unique number with the lowest position.

Given array A such that:

```
A[0] = 6
A[1] = 4
A[2] = 4
A[3] = 6
```

the function should return −1. There is no unique number in A (4 and 6 occur more than once).

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

- N is an integer within the range [1..100,000];
- each element of array A is an integer within the range [0..1,000,000,000].

14:58:44 15:05:47

Code: 15:05:47 UTC, py, final, score: **100** show code in pop-up

```
# you can write to stdout for debuggin
# print("this is a debug message")

def solution(A):
    # Implement your solution here
    from collections import Counter

    counts = Counter(A)

    for num in counts:
        if counts[num] == 1:
            return num

    return -1

```

Analysis summary

The solution obtained perfect score.

Analysis

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

expand all **Example tests**

▶ example0 ✔ **OK**
example

▶ example1 ✔ **OK**
example

▶ example2 ✔ **OK**
example

expand all **Correctness tests**

▶ extreme_single ✔ **OK**
single element

▶ extreme_no_unique ✔ **OK**
no unique value and [1,2,3,4]

▶ extreme_min_max_value ✔ **OK**
min/max values

▶ small1 ✔ **OK**
small correctness test

▶ small2 ✔ **OK**
small correctness test

▶ small3 ✔ **OK**

small correctness tests

expand all

Performance tests

▶	medium1 medium tests with few unique values, N = 10,003,	✔ OK
▶	medium2 medium tests with few unique values, N = 10,209,	✔ OK
▶	large large tests with many minimal and maximal values, N = 50,000	✔ OK
▶	big1 large test, N = 100,000	✔ OK
▶	big2 large test, N = 100,000	✔ OK