# Codility\_

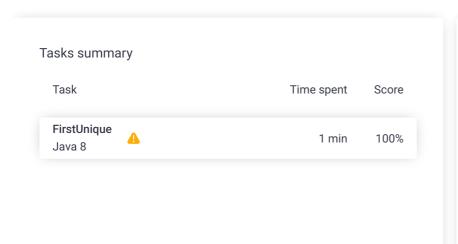
# CodeCheck Report: training6FDCHA-M8M

Test Name:

Summary

Timeline

Check out Codility training tasks





#### **Tasks Details**

#### 1. FirstUnique

Eas

Find the first unique number in a given sequence. Task Score

Correctness

100%

100%

Performance

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).

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:

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

#### Solution

Programming language used: Java 8

Total time used: 1 minutes

Effective time used: 1 minutes

Notes: not defined yet

# Task timeline



Code: 20:40:57 UTC, java,

show code in pop-up

final, score: 100

1 // you can also use imports, for example:

import java.util.\*;

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].

Copyright 2009–2023 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

```
Test results - Codility
```

```
// you can write to stdout for debugging purp
 5
     // System.out.println("this is a debug messag
 6
 7
     class Solution {
 8
             public int solution(int[] A) {
 9
                      Map<Integer, Integer> firstIr
10
11
                      for (int i = 0; i < A.length;</pre>
                               if (firstIndicies.cor
12
13
                                       firstIndicies
                              } else {
14
15
                                       firstIndicies
16
                              }
                      }
17
18
19
                      int minIndex = Integer.MAX_VA
20
                      for (int index : firstIndici€
21
                              minIndex = Integer.mi
22
23
24
                      return minIndex == Integer.MA
25
         }
     }
26
```

#### Analysis summary

The solution obtained perfect score.

## Analysis

# Detected time complexity: $\frac{O(N * log(N))}{log(N)}$

expa	and all	Example test	ts	
<b></b>	example0		V	OK
	example			
	example1		~	OK
	example			
	example2		~	OK
	example			
ехра	and all	Correctness te	ests	S
	extreme_single	е	V	OK
	single element			
	oxoo	-	~	OK
	no unique value ar	nd [1,2,3,4]		
	extreme_min_i	max_value	~	OK
	min/max values			
	small1		~	OK
	small correctness	test		
	small2		~	OK
	small correctness	test		
	small3		~	OK
	small correctness			
ехра	and all	Performance to	est	is
	medium1		~	OK
		n few unique values,		
	N = 10,003,			

## Test results - Codility

•	medium2 medium tests with few unique values, N = 10,209,	<b>✓</b> OK
•	large large tests with many minimal and maximal values, N = 50,000	<b>∨</b> OK
•	big1 large test, N = 100,000	<b>✓</b> OK
•	big2 large test, N = 100,000	<b>✓</b> OK