Codility_

Candidate Report: trainingW9JXX7-6SP

Check out Codility training tasks

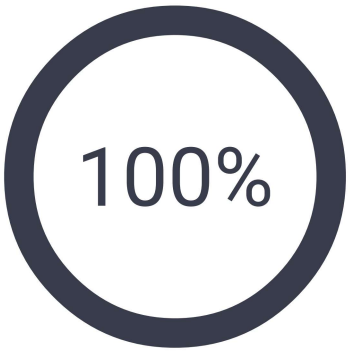
Test Name:

Summary Review (0) Timeline

Tasks summary

Task	Time spent	Score
Dominator Java 8	2 min	100%

Total score



Tasks Details

Easy

1. Dominator

Find an index of an array such that its value occurs at more than half of indices in the array.

Task Score	Correctness	Performance
100%	100%	100%

Task description

An array A consisting of N integers is given. The *dominator* of array A is the value that occurs in more than half of the elements of A.

For example, consider array A such that

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

The dominator of A is 3 because it occurs in 5 out of 8 elements of A (namely in those with indices 0, 2, 4, 6 and 7) and 5 is more than a half of 8.

Write a function

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

that, given an array A consisting of N integers, returns index of any element of array A in which the dominator of A occurs. The function should return −1 if array A does not have a dominator.

For example, given array A such that

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

the function may return 0, 2, 4, 6 or 7, as explained above.

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

- N is an integer within the range [0..100,000];

Solution

Programming language used:	Java 8
Total time used:	2 minutes
Effective time used:	2 minutes
Notes:	*not defined yet*

Task timeline

22:33:12 22:34:37

Code: 22:34:37 UTC, java, final, score: **100** show code in pop-up

```
1  // you can also use imports, for example:
2  // import java.util.*;
3
4  // you can write to stdout for debugging purposes, e.g.
5  // System.out.println("this is a debug message");
6  import java.util.Hashtable;
7  class Solution {
```

- each element of array A is an integer within the range [−2,147,483,648..2,147,483,647].


```
8     public int solution(int[] A) {
9
10                // write your code in Java SE 8
11                Hashtable<Integer, Integer> occuranceTable
12
13                if (A != null) {
14                        int dominLimit = A.length / 2;
15                        if (A.length ==1)
16                                return 0;
17                        else if(A.length>1){
18                                for (int i = 0; i<A.length
19                                        if (occuranceTable
20                                                occuranceT
21                                        } else {
22                                                int count
23                                                count++;
24                                                if (count
25                                                        re
26                                                else
27                                                        oc
28                                        }
29                                }
30
31                        }
32                }
33                return -1;
34
35    }
36 }
```

Analysis summary

The solution obtained perfect score.

Analysis

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

expand all	Example tests	
▶	example example test	✓ OK
expand all	Correctness tests	
▶	small_nondominator all different and all the same elements	✓ OK
▶	small_half_positions half elements the same, and half + 1 elements the same	✓ OK
▶	small small test	✓ OK
▶	small_pyramid decreasing and plateau, small	✓ OK
▶	extreme_empty_and_single_item empty and single element arrays	✓ OK
▶	extreme_half1 array with exactly N/2 values 1, N even + [0,0,1,1,1]	✓ OK
▶	extreme_half2 array with exactly floor(N/2) values 1, N odd + [0,0,1,1,1]	✓ OK
▶	extreme_half3 array with exactly ceil(N/2) values 1 + [0,0,1,1,1]	✓ OK
	Performance tests	

expand all

▶	medium_pyramid decreasing and plateau, medium	✓ OK
▶	large_pyramid decreasing and plateau, large	✓ OK
▶	medium_random random test with dominator, N = 10,000	✓ OK
▶	large_random random test with dominator, N = 100,000	✓ OK