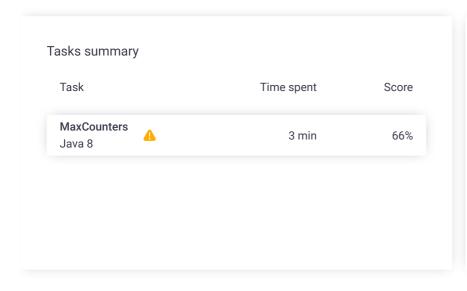
# Codility\_

# CodeCheck Report: trainingQR8RAS-DDP

Test Name:

Summary Timeline Check out Codility training tasks





#### **Tasks Details**

### 1. MaxCounters

Calculate the values of counters after applying all alternating operations: increase counter by 1; set value of all counters to current maximum.

Task Score 66% Correctness

Performance

100%

40%

## Task description

You are given N counters, initially set to 0, and you have two possible operations on them:

- increase(X) counter X is increased by 1,
- max counter all counters are set to the maximum value of any counter.

A non-empty array A of M integers is given. This array represents consecutive operations:

- if A[K] = X, such that  $1 \le X \le N$ , then operation K is increase(X),
- if A[K] = N + 1 then operation K is max counter.

For example, given integer N = 5 and array A such that:

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

# Solution

Programming language used: Java 8 Total time used: 3 minutes Effective time used: 3 minutes Notes: not defined yet

Task timeline



Code: 01:05:01 UTC, java, final, score: 66

show code in pop-up

the values of the counters after each consecutive operation will he:

(0, 0, 1, 0, 0) (0, 0, 1, 1, 0) (0, 0, 1, 2, 0) (2, 2, 2, 2, 2) (3, 2, 2, 2, 2) (3, 2, 2, 3, 2) (3, 2, 2, 4, 2)

The goal is to calculate the value of every counter after all operations.

Write a function:

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

that, given an integer N and a non-empty array A consisting of M integers, returns a sequence of integers representing the values of the counters.

Result array should be returned as an array of integers.

For example, given:

A[0] = 3 A[1] = 4 A[2] = 4 A[3] = 6 A[4] = 1 A[5] = 4 A[6] = 4

the function should return [3, 2, 2, 4, 2], as explained above.

Write an efficient algorithm for the following assumptions:

- N and M are integers within the range [1..100,000];
- each element of array A is an integer within the range [1..N + 1].

Copyright 2009–2022 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

#### Test results - Codility

```
1
     import java.util.*;
2
     import java.util.Arrays;
 3
4
     class Solution {
 5
         // https://github.com/burakozcelik5028/JavaHel
 6
         public static int[] solution(int N, int[] A) {
 7
                      int[] B = new int[N];
 8
                      Arrays.fill(B, 0, N, 0);
 9
                      for(int x=0; x < A.length; x++) {
10
11
                               if(A[x] <= N && A[x] >= 1)
12
                                       B[A[x]-1] = B[A[x]
13
                               } else if(A[x] == N+1) {
14
                                       int max = maximum(
15
                                       Arrays.fill(B, ∅,
16
                               }
17
                      }
18
19
                      return B;
20
             }
             public static int maximum(int[] numbers)
21
22
         {
23
              int maxSoFar = numbers[0];
24
25
              // for each loop
26
              for (int num : numbers)
27
28
                  if (num > maxSoFar)
29
                  {
30
                      maxSoFar = num;
31
                  }
32
             }
33
          return maxSoFar;
34
         }
35
```

# Analysis summary

The following issues have been detected: timeout errors.

### **Analysis**

Detected time complexity: O(N\*M)

expand all	Exampl	e tests	
example example test		√ OK	
expand all	Correctne	ess tests	
extreme_sm     all max_counter		√ OK	
single only one count	er	√ OK	
small_rando small random t operations	m1 est, 6 max_count	✓ OK er	
small_rando small random t operations	m2 est, 10 max_coun	✓ <b>OK</b> ter	
expand all	Performa	nce tests	
medium_rar medium randor operations	ndom1 m test, 50 max_co	✓ <b>OK</b> ounter	

### Test results - Codility

•	medium_random2 medium random test, 500 max_counter operations	✓	OK
•	large_random1 large random test, 2120 max_counter operations	X	TIMEOUT ERROR running time: 2.124 sec., time limit: 1.648 sec.
•	large_random2 large random test, 10000 max_counter operations	X	TIMEOUT ERROR Killed. Hard limit reached: 8.000 sec.
•	extreme_large all max_counter operations	X	TIMEOUT ERROR Killed. Hard limit reached: