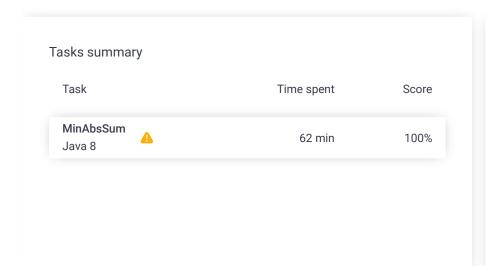
Codility_

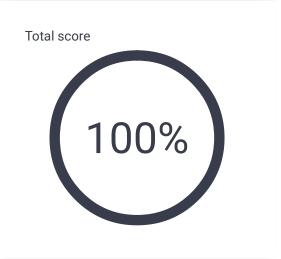
CodeCheck Report: trainingTMCRVH-47A

Test Name:

Check out Codility training tasks

Summary Timeline





Tasks Details

1. MinAbsSum

Hard

Given array of integers, find the lowest absolute sum of elements. **Task Score**

Correctness

100%

Performance

eriorinance

100%

Task description

For a given array A of N integers and a sequence S of N integers from the set $\{-1, 1\}$, we define val(A, S) as follows:

$$val(A, S) = |sum\{A[i]*S[i] \text{ for } i = 0..N-1\}|$$

(Assume that the sum of zero elements equals zero.)

For a given array A, we are looking for such a sequence S that minimizes val(A,S).

Write a function:

that, given an array A of N integers, computes the minimum value of val(A,S) from all possible values of val(A,S) for all possible sequences S of N integers from the set $\{-1, 1\}$.

For example, given array:

$$A[0] = 1$$

A[1] = 5

Solution

Programming language used: Java 8

Total time used: 62 minutes

100%

Effective time used: 62 minutes

Notes: not defined yet

Task timeline

08:03:17 09:04:46

your function should return 0, since for S = [-1, 1, -1, 1], val(A, S) =0, which is the minimum possible value.

Write an efficient algorithm for the following assumptions:

- N is an integer within the range [0..20,000];
- each element of array A is an integer within the range [-100..100].

Copyright 2009-2021 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

Test results - Codility

Code: 09:04:46 UTC, java,

```
show code in pop-up
 final, score: 100
 1
     // you can also use imports, for example:
 2
     // import java.util.*;
 3
 4
     // you can write to stdout for debugging purposes,
 5
     // System.out.println("this is a debug message");
 6
     class Solution {
7
        public int solution(int[] a){
 8
 9
         if (a.length == 0) return 0;
10
         if (a.length == 1) return a[0];
11
         int sum = 0;
12
         for (int i=0;i<a.length;i++){</pre>
13
              sum += Math.abs(a[i]);
14
         int[] indices = new int[a.length];
15
         indices[0] = 0;
16
         int half = sum/2;
17
18
         int localSum = Math.abs(a[0]);
19
         int minLocalSum = Integer.MAX_VALUE;
20
         int placeIndex = 1;
21
         for (int i=1;i<a.length;i++){</pre>
22
              if (localSum<half){</pre>
23
                  if (Math.abs(2*minLocalSum-sum) > Math.
24
                      minLocalSum = localSum;
25
                  localSum += Math.abs(a[i]);
26
                  indices[placeIndex++] = i;
27
              }else{
28
                  if (localSum == half)
29
                      return Math.abs(2*half - sum);
30
                  if (Math.abs(2*minLocalSum-sum) > Math.
31
                      minLocalSum = localSum;
32
33
                  if (placeIndex > 1) {
                      localSum -= Math.abs(a[indices[plac
34
35
                      i = indices[placeIndex];
36
                  }
37
              }
38
         return (Math.abs(2*minLocalSum - sum));
39
40
41
     }
42
     }
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: max(abs(A))**2)

expand all	Example tests	
example1 example test	✓ OK	
expand all	Correctness tests	

Test results - Codility

simp		✓ OK
•	simple2 simple 2	√ OK
•	simple3	√ OK
•	range range 220	√ OK
•	extreme empty and single ele	✓ OK ment
•	functional small functional tes	√ OK
ехра	nd all	Performance tests
•	medium1 medium random	√ OK
•	medium2 multiples of 10 + 5	√ OK
•	big1 multiples of 5 + 42	√ OK
•	big3 all 4s and one 3	√ OK
•	big4 multiples of 10	√ OK