Codility_

CodeCheck Report: trainingPXXHD3-DQS

Test Name:

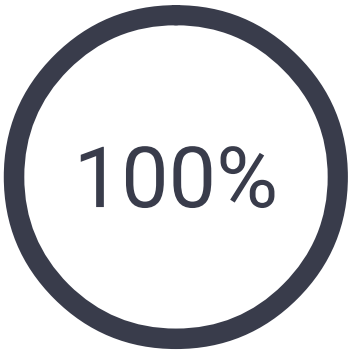
Check out Codility training tasks

Summary Timeline

Tasks summary

Task	Time spent	Score
MinAbsSum Java 8	35 min	100%

Total score



Tasks Details

Hard

1. MinAbsSum

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

Task Score	Correctness	Performance
100%	100%	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] 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:

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

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
A[2] =  2
A[3] = -2
```

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:

Solution

Programming language used:	Java 8
Total time used:	35 minutes
Effective time used:	35 minutes
Notes:	*not defined yet*

Task timeline

08:25:10 08:59:36

Code: 08:59:36 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,
```

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


```
// System.out.println("this is a debug message");

class Solution {
   public int solution(int[] a){
    if (a.length == 0) return 0;
    if (a.length == 1) return a[0];
    int sum = 0;
    for (int i=0;i<a.length;i++){
        sum += Math.abs(a[i]);
    }
    int[] indices = new int[a.length];
    indices[0] = 0;
    int half = sum/2;
    int localSum = Math.abs(a[0]);
    int minLocalSum = Integer.MAX_VALUE;
    int placeIndex = 1;
    for (int i=1;i<a.length;i++){
        if (localSum<half){
            if (Math.abs(2*minLocalSum-sum) > Math
                minLocalSum = localSum;
            localSum += Math.abs(a[i]);
            indices[placeIndex++] = i;
        }else{
            if (localSum == half)
                return Math.abs(2*half - sum);

            if (Math.abs(2*minLocalSum-sum) > Math
                minLocalSum = localSum;
            if (placeIndex > 1) {
                localSum -= Math.abs(a[indices[pla
                i = indices[placeIndex];
            }
        }
    }
    return (Math.abs(2*minLocalSum - sum));

}
}
```

Analysis summary

The solution obtained perfect score.

Analysis

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

expand all	Example tests	
▶	example1 example test	✓ OK
expand all	Correctness tests	
▶	simple1 simple 1	✓ OK
▶	simple2 simple 2	✓ OK
▶	simple3 simple 3	✓ OK
▶	range range 2..20	✓ OK
▶	extreme empty and single element	✓ OK
▶	functional small functional test	✓ OK

expand 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