Codility_

CodeCheck Report: trainingEQGNJM-TFQ

Test Name:

Check out Codility training tasks

Summary Timeline

Tasks summary

Task	Time spent	Score
MinAbsSumOfTwo ⚠ Python	26 min	100%

Total score

Tasks Details

1. MinAbsSumOfTwo

Medium

Find the minimal absolute value of a sum of two elements.

Task Score	Correctness	Performance
100%	100%	100%

Task description

Let A be a non-empty array consisting of N integers.

The *abs sum of two* for a pair of indices (P, Q) is the absolute value |A[P] + A[Q]|, for 0 ≤ P ≤ Q < N.

For example, the following array A:

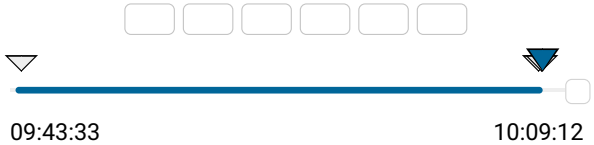
```
A[0] =  1
A[1] =  4
A[2] = −3
```

has pairs of indices (0, 0), (0, 1), (0, 2), (1, 1), (1, 2), (2, 2).
The abs sum of two for the pair (0, 0) is A[0] + A[0] = |1 + 1| = 2.
The abs sum of two for the pair (0, 1) is A[0] + A[1] = |1 + 4| = 5.
The abs sum of two for the pair (0, 2) is A[0] + A[2] = |1 + (−3)| = 2.
The abs sum of two for the pair (1, 1) is A[1] + A[1] = |4 + 4| = 8.
The abs sum of two for the pair (1, 2) is A[1] + A[2] = |4 + (−3)|

Solution

Programming language used:	Python
Total time used:	26 minutes
Effective time used:	26 minutes
Notes:	*not defined yet*

Task timeline



Code: 10:09:11 UTC, py, show code in pop-up

= 1.
The abs sum of two for the pair (2, 2) is A[2] + A[2] = |(−3) + (−3)| = 6.

Write a function:

```
def solution(A)
```

that, given a non-empty array A consisting of N integers, returns the minimal abs sum of two for any pair of indices in this array.

For example, given the following array A:

```
A[0] =  1
A[1] =  4
A[2] = −3
```

the function should return 1, as explained above.

Given array A:

```
A[0] = −8
A[1] =  4
A[2] =  5
A[3] =−10
A[4] =  3
```

the function should return |(−8) + 5| = 3.

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 [−1,000,000,000..1,000,000,000].

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

final, score: **100**

```python
# you can write to stdout for debugging pur
# print("this is a debug message")

def solution(A):
    # Implement your solution here
    # pass
    A.sort()
    N = len(A)

    front, back = 0, N - 1
    min_abs_sum = float('inf')

    while front <= back:
        current_sum = A[front] + A[back]
        min_abs_sum = min(min_abs_sum, abs(

        if current_sum <= 0:
            front += 1
        else:
            back -= 1

    return min_abs_sum
```

Analysis summary

The solution obtained perfect score.

Analysis

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

expand all	Example tests	
▶	example1 first example	✔ OK
▶	example2 second example	✔ OK
expand all	**Correctness tests**	
▶	extreme_single sequences of 1 elements	✔ OK
▶	extreme_double sequences of 2 elements	✔ OK
▶	positive_small only positive numbers	✔ OK
▶	negative_small only negative numbers	✔ OK
expand all	**Performance tests**	
▶	random_small random sequence, length = ~1000	✔ OK
▶	random_medium random sequence, length = ~10,000	✔ OK
▶	arithmetic_medium arithemtic sequence, length = ~10,000	✔ OK

▶	random_large random sequence, length = ~100,000	✔ OK
▶	extreme_large sequence of MAX_INT, length = ~100,000	✔ OK
▶	arithmetic_large arithmetic sequence, length = ~100,000	✔ OK
▶	constant_distance constant distance between all elements, length = 100,000	✔ OK