codility

Congratulations

You have completed a Codility training test.

Tweet this!
I scored 100% in #java on @Codility!
https://codility.com/demo/take-sample-test/tape_equilibriur

Sign up for our newsletter!

Like us on Facebook!

Training ticket

Session

ID: training52PFRH-CRR
Time limit: 120 min.

Status: closed

Created on: 2016-06-19 18:39 UTC
Started on: 2016-06-19 18:39 UTC
Finished on: 2016-06-19 18:40 UTC

Tasks in test	Correctness	Performance	Task score
1 TapeEquilibrium Submitted in: Java	100%	100%	100%

Test score

100%

100 out of 100 points

EASY

1. TapeEquilibrium

Minimize the value |(A[0] + ... + A[P-1]) - (A[P] + ... + A[N-1])|.

score: 100 of 100

Task description

A non-empty zero-indexed array A consisting of N integers is given. Array A represents numbers on a tape.

Any integer P, such that 0 < P < N, splits this tape into two non-empty parts: A[0], A[1], ..., A[P − 1] and A[P], A[P + 1], ..., A[N − 1].

The *difference* between the two parts is the value of: |(A[0] + A[1] + ... + A[P − 1]) − (A[P] + A[P + 1] + ... + A[N − 1])|

In other words, it is the absolute difference between the sum of the first part and the sum of the second part.

For example, consider array A such that:

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

We can split this tape in four places:

- P = 1, difference = |3 − 10| = 7
- P = 2, difference = |4 − 9| = 5
- P = 3, difference = |6 − 7| = 1
- P = 4, difference = |10 − 3| = 7

Write a function:

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

Solution

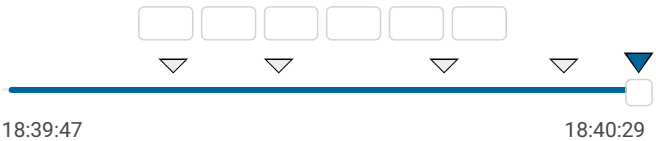
Programming language used: Java

Total time used: 1 minutes

Effective time used: 1 minutes

Notes: *not defined yet*

Task timeline



18:39:47 18:40:29

Code: 18:40:29 UTC, java, final, score: **100**

show code in pop-up

```
1  import java.util.ArrayList;
2  import java.util.List;
3  import java.util.Optional;
4
5  public class Solution {
6
7      public int solution(int[] A) {
8          List<Integer> integral = countIntegral(A);
```

that, given a non-empty zero-indexed array A of N integers, returns the minimal difference that can be achieved.

For example, given:

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

the function should return 1, as explained above.

Assume that:

- N is an integer within the range [2..100,000];
- each element of array A is an integer within the range [−1,000..1,000].

Complexity:

- expected worst-case time complexity is O(N);
- expected worst-case space complexity is O(N), beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.


```
 9          int sum = integral.get(integral.size()-1);
10          Optional<Integer> min = integral.stream()
11                  .limit(integral.size()-1)
12                  .map(value -> Math.abs(value - (sum - v
13                  .min(Integer::compare);
14
15          return min.get();
16      }
17
18      private List<Integer> countIntegral(int[] A) {
19          List<Integer> sums = new ArrayList<>(A.length);
20
21          sums.add(A[0]);
22          for (int i = 1; i < A.length; i++) {
23              sums.add(A[i] + sums.get(i-1));
24          }
25
26          return sums;
27      }
28  }
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity:
O(N)

expand all	Example tests	
▶	example example test	✔ OK
expand all	Correctness tests	
▶	double two elements	✔ OK
▶	simple_positive simple test with positive numbers, length = 5	✔ OK
▶	simple_negative simple test with negative numbers, length = 5	✔ OK
▶	small_random random small, length = 100	✔ OK
▶	small_range range sequence, length = ~1,000	✔ OK
▶	small small elements	✔ OK
expand all	Performance tests	
▶	medium_random1 random medium, numbers from 0 to 100, length = ~10,000	✔ OK
▶	medium_random2 random medium, numbers from -1,000 to 50, length = ~10,000	✔ OK
▶	large_ones large sequence, numbers from -1 to 1, length = ~100,000	✔ OK
▶	large_random random large, length = ~100,000	✔ OK
▶	large_sequence large sequence, length = ~100,000	✔ OK
▶	large_extreme large test with maximal and minimal values, length = ~100,000	✔ OK

Training center