Open Feedback Dialog
Open Feedback Dialog

codility

ing center
Codility training tasks

Congratulations

You have completed a Codility demo.

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

Sign up for our newsletter!

Like us on Facebook!

Demo ticket

Session

ID: demo7ANYMF-GGQ
Time limit: 120 min.

Status: closed

Created on: 2014-12-09 09:45 UTC
Started on: 2014-12-09 09:48 UTC
Finished on: 2014-12-09 10:49 UTC

Tasks in test	Correctness	Performance	Task score
1 TapeEquilibrium	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:

```
def solution(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.

Solution

Programming language used: Ruby

Total time used: 62 minutes

Effective time used: 62 minutes

Notes: *not defined yet*

Task timeline

09:48:00 10:49:32

Code: 10:49:32 UTC, rb, final, score: **100.00**

```ruby
# you can use puts for debugging purposes, e.g.
# puts "this is a debug message"

def solution(a)
  sum_a = sum(a)
  sum = sum_a[-1]
  a = sum - sum_a[0]
  b = sum_a[0]
  min = (a - b).abs
  (1..sum_a.count - 2).each do |i|
    a = sum - sum_a[i]
    b = sum_a[i]
    n =  (a - b).abs
    if n  < min
      min = n
```

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.


```
16      end
17    end
18    min
19  end
20
21  def sum(a)
22    r = []
23    sum = 0
24    a.each do |n|
25      sum += n
26      r << sum
27    end
28    r
29  end
```

Analysis

Detected time complexity:

O(N)

test	time	result
Example tests		
example example test	0.060 s	**OK**
Correctness tests		
double two elements	0.060 s	**OK**
simple_positive simple test with positive numbers, length = 5	0.056 s	**OK**
simple_negative simple test with negative numbers, length = 5	0.064 s	**OK**
small_random random small, length = 100	0.064 s	**OK**
small_range range sequence, length = ~1,000	0.056 s	**OK**
small small elements	0.064 s	**OK**
Performance tests		
medium_random1 random medium, numbers from 0 to 100, length = ~10,000	0.076 s	**OK**
medium_random2 random medium, numbers from -1,000 to 50, length = ~10,000	0.080 s	**OK**
large_ones large sequence, numbers from -1 to 1, length = ~100,000	0.224 s	**OK**
large_random random large, length = ~100,000	0.228 s	**OK**
large_sequence large sequence, length = ~100,000	0.144 s	**OK**
large_extreme large test with maximal and minimal values, length = ~100,000	0.200 s	**OK**

Training center