

**Congratulations**

You have completed a Codility demo.

Tweet this!

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

Sign up for our newsletter!

Like us on Facebook!

Demo ticket**Session**ID: demoBJHGH9-ZBS
Time limit: 120 min.**Status: closed**Created on: 2015-07-21 14:43 UTC
Started on: 2015-07-21 14:44 UTC
Finished on: 2015-07-21 15:04 UTC**Tasks in test**1 | **TapeEquilibrium****Correctness**

100%

Performance

100%

Task score

100%

Test score**100%**

100 out of 100 points

EASY

1. TapeEquilibriumMinimize the value $|(A[0] + \dots + A[P-1]) - (A[P] + \dots + A[N-1])|$.**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: $A[0], A[1], \dots, A[P-1]$ and $A[P], A[P+1], \dots, A[N-1]$. The *difference* between the two parts is the value of: $|(A[0] + A[1] + \dots + A[P-1]) - (A[P] + A[P+1] + \dots + A[N-1])|$. 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:

```
object Solution { def solution(A: Array[Int]): Int
}
```

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 $[2..100,000]$;

Solution**Programming language used:** Scala**Total time used:** 20 minutes**Effective time used:** 20 minutes**Notes:** *not defined yet***Task timeline**

14:44:13

15:04:02

Code: 15:04:02 UTC, scala, final, score: **100.00**
[show code in pop-up](#)

```
1 import scala.collection._
2
3 object Solution {
4   def solution(A: Array[Int]): Int = {
5     val left = A(0)
6     val right = A.tail.sum
7     A.tail.foldLeft((left, right, Set[Int]())) {
8       case ((l, r, b: Set[Int]), e) => (l+e, r-e, b + Math.ab
9     }.min
10  }
11 }
```

Analysis**Detected time complexity:**

- 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.

Copyright 2009–2015 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

$O(N)$

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