codility

Check out Codility training tasks

Candidate Report: Anonymous

Test Name:

Summary Timeline

Test Score

100 out of 100 points

100%

Tasks in Test

	Time Spent	Task Score
MinAvgTwoSlice Submitted in: Python	1 min	100%

TASKS DETAILS

MEDIUM

1. MinAvgTwoSlice

Find the minimal average of any slice containing at least two elements.

Task Score	Correctness	Performance
100%	100%	100%

Task description

A non-empty array A consisting of N integers is given. A pair of integers (P, Q), such that 0 ≤ P < Q < N, is called a *slice* of array A (notice that the slice contains at least two elements). The *average* of a slice (P, Q) is the sum of A[P] + A[P + 1] + ... + A[Q] divided by the length of the slice. To be precise, the average equals (A[P] + A[P + 1] + ... + A[Q]) / (Q − P + 1).

For example, array A such that:

```
A[0] = 4
A[1] = 2
A[2] = 2
A[3] = 5
A[4] = 1
A[5] = 5
A[6] = 8
```

contains the following example slices:

- slice (1, 2), whose average is (2 + 2) / 2 = 2;
- slice (3, 4), whose average is (5 + 1) / 2 = 3;
- slice (1, 4), whose average is (2 + 2 + 5 + 1) / 4 = 2.5.

The goal is to find the starting position of a slice whose average is minimal.

Write a function:

```
def solution(A)
```

that, given a non-empty array A consisting of N integers, returns the starting position of the slice with the minimal average. If there is more than one slice with a minimal average, you should return the smallest starting position of such a slice.

Solution

Programming language used:	Python
Total time used:	1 minutes
Effective time used:	1 minutes
Notes:	*not defined yet*

Task timeline

07:36:07 07:36:41

Code: 07:36:40 UTC, py, final, score: **100** show code in pop-up

```
1  def solution(a):
2      """Returns position of first slice with minimal mean
3
4      def slices_summary(slices):
5          """Return slices summary: minimal mean value and
6          if len(slices) < 1:
7              return 0, 0
```

For example, given array A such that:

```
A[0] = 4
A[1] = 2
A[2] = 2
A[3] = 5
A[4] = 1
A[5] = 5
A[6] = 8
```

the function should return 1, as explained above.

Write an **efficient** algorithm for the following assumptions:

- N is an integer within the range [2..100,000];
- each element of array A is an integer within the range [−10,000..10,000].


```
8          min_value = min(slices)
9          min_index = slices.index(min_value)
10         return min_value, min_index
11
12     slice_means_2 = [sum(x) / len(x) for x in zip(a[:-1],
13     slice_means_3 = [sum(x) / len(x) for x in zip(a[:-2],
14
15     _, position = min(slices_summary(slice_means_2), slic
16
17     return position
```

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_quadruple two or four elements	✓ OK
▶	simple1 simple test, the best slice has length 3	✓ OK
▶	simple2 simple test, the best slice has length 3	✓ OK
▶	small_random random, length = 100	✓ OK
▶	medium_range increasing, decreasing (legth = ~100) and small functional	✓ OK
expand all	**Performance tests**	
▶	medium_random random, N = ~700	✓ OK
▶	large_ones numbers from -1 to 1, N = ~100,000	✓ OK
▶	large_random random, N = ~100,000	✓ OK
▶	extreme_values all maximal values, N = ~100,000	✓ OK
▶	large_sequence many seqeneces, N = ~100,000	✓ OK