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/min_avg_two_s

Sign up for our newsletter!

Like us on Facebook!

Training ticket

Session

ID: trainingGDJXFG-NJ5
Time limit: 120 min.

Status: closed

Created on: 2016-06-22 18:30 UTC
Started on: 2016-06-22 18:30 UTC
Finished on: 2016-06-22 18:30 UTC

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

Test score

100%

100 out of 100 points

MEDIUM

1. MinAvgTwoSlice

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

score: 100 of 100

Task description

A non-empty zero-indexed 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:

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

Solution

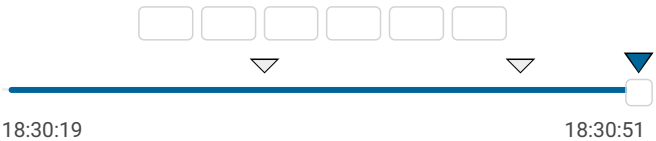
Programming language used: Java

Total time used: 1 minutes

Effective time used: 1 minutes

Notes: *not defined yet*

Task timeline



Code: 18:30:51 UTC, java, final, score: **100**

show code in pop-up

```
1  class Solution {
2          int[] values;
3      double min = Double.MAX_VALUE;
4      int minId = -1;
5
6      public int solution(int[] A) {
7          this.values = A;
8          for (int id = 0; id < values.length - 2; id++)
```

that, given a non-empty zero-indexed 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.

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.

Assume that:

- 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].

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 2009–2016 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

```
 9            double twoSlice = countAvg(id, id+1);
10            double threeSlice = countAvg(id, id+2);
11            checkMin(twoSlice, id);
12            checkMin(threeSlice, id);
13        }
14        checkMin(countAvg(values.length-2, values.lengt
15        return minId;
16    }
17
18    private void checkMin(double value, int id) {
19        if (value < min) {
20            min = value;
21            minId = id;
22        }
23    }
24
25    private double countAvg(int from, int to){
26        int sum = 0;
27        for(int i = from; i<=to; i++){
28            sum += values[i];
29        }
30        return (double) sum / (to-from+1);
31    }
32 }
```

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

Training center