

Tasks summary

Task	Time spent	Score
MinAvgTwoSlice C#	9 min	100%

Total score

100%

Tasks Details

Medium

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

Task Score

100%

Correctness

100%

Performance

100%

Task description

A non-empty array *A* consisting of *N* integers is given. A pair of integers (*P*, *Q*), such that $0 \leq P < Q < N$, 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 $A[P] + A[P + 1] + \dots + A[Q]$ divided by the length of the slice. To be precise, the average equals $(A[P] + A[P + 1] + \dots + A[Q]) / (Q - P + 1)$.

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 $(2 + 2) / 2 = 2$;

Solution

Programming language used: C#

Total time used:

9 minutes

?

Effective time used:

9 minutes

?

Notes:

not defined yet

Task timeline

15:19:37

15:28:07

- 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); }
```

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.

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].

Copyright 2009–2022 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

Code: 15:28:06 UTC, cs, final,
score: 100

[show code in pop-up](#)

```
1  using System;
2
3  /**
4   * 5.4 - Min Avg Two Slice
5   * Paulo Santos
6   * 07.Dec.2022
7   *
8   * Thanks to Mr.Duhart
9   * (https://stackoverflow.com/users/4281529/
10  */
11  class Solution {
12      public int solution(int[] A) {
13
14          /*
15           * Find prefix sum.
16           */
17          int N = A.Length;
18          var ps = new int[N + 1];
19
20          for (int i = 1; i <= N; i++) {
21              ps[i] = A[i - 1] + ps[i - 1];
22          }
23
24          int lftIdx, minLftIdx;
25          double avgHere, minAvg, avgOfTwo, avgWithPrev;
26
27          /*
28           * Initialize variables at the first possible s
29           */
30          lftIdx = minLftIdx = 0;
31          avgHere = minAvg = (A[0] + A[1]) / 2.0;
32
33          /*
34           * Find min average of every slice that ends at
35           * starting at i = 2.
36           */
37          for (int i = 2; i < N; i++) {
38
39              /*
40               * average of A[lftIdx : i]
41               */
42              avgWithPrev = ((double) ps[i + 1] - ps[lftI
43                          (i - lftIdx + 1));
44
45              /*
46               * average of A[i - 1 : i]
47               */
48              avgOfTwo = (A[i - 1] + A[i]) / 2.0;
49
50              /*
51               * Find minimum and update lftIdx of slice
52               * (previous lftIdx or i - 1).
53               */
54              if (avgOfTwo < avgWithPrev) {
55                  avgHere = avgOfTwo;
56                  lftIdx = i - 1;
57              }
58              else
59                  avgHere = avgWithPrev;
60
61              /*
62               * Keep track of minimum so far and its lef
63               */
64              if (avgHere < minAvg) {
65                  minAvg = avgHere;
66                  minLftIdx = lftIdx;
67              }
68          }
69
70          return minLftIdx;
```

71	}
72	}

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: O(N)	
expand all	Example tests
▶ example	✓ OK
example test	
expand all	Correctness tests
▶ double_quadruple	✓ OK
two or four elements	
▶ simple1	✓ OK
simple test, the best slice has length 3	
▶ simple2	✓ OK
simple test, the best slice has length 3	
▶ small_random	✓ OK
random, length = 100	
▶ medium_range	✓ OK
increasing, decreasing (length = ~100) and small functional	
expand all	Performance tests
▶ medium_random	✓ OK
random, N = ~700	
▶ large_ones	✓ OK
numbers from -1 to 1, N = ~100,000	
▶ large_random	✓ OK
random, N = ~100,000	
▶ extreme_values	✓ OK
all maximal values, N = ~100,000	
▶ large_sequence	✓ OK
many sequences, N = ~100,000	