Demo ticket

Session

ID: demoU7P8NV-5JB
Time limit: 120 min.

Status: closed

Created on: 2014-03-17 16:22 UTC
Started on: 2014-03-17 16:22 UTC
Finished on: 2014-03-17 16:31 UTC

Tasks in test

Task score

Test score

100%

100 out of 100 points

MEDIUM

1. MaxSliceSum

Find a maximum sum of a compact subsequence of array 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. The *sum* of a slice (P, Q) is the total of A[P] + A[P+1] + ... + A[Q].
Write a function:

```
int solution(const vector<int> &A);
```

that, given an array A consisting of N integers, returns the maximum sum of any slice of A.
For example, given array A such that:

```
A[0] = 3  A[1] = 2  A[2] = -6
A[3] = 4  A[4] = 0
```

the function should return 5 because:

- (3, 4) is a slice of A that has sum 4,
- (2, 2) is a slice of A that has sum −6,
- (0, 1) is a slice of A that has sum 5,
- no other slice of A has sum greater than (0, 1).

Assume that:

- N is an integer within the range [1..1,000,000];
- each element of array A is an integer within the range [−1,000,000..1,000,000];
- the result will be an integer within the range [−2,147,483,648..2,147,483,647].

Complexity:

- expected worst-case time complexity is O(N);
- expected worst-case space complexity is O(1), beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

Solution

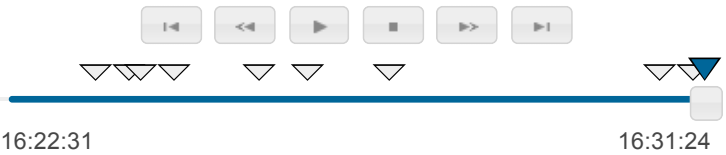
Programming language used: C++

Total time used: 9 minutes

Effective time used: 9 minutes

Notes: correct functionality and scalability

Task timeline



16:22:31 16:31:24

Code: 16:31:24 UTC, cpp, final, score: **100.00**

```cpp
// you can also use includes, for example:
#include <algorithm>
int solution(const vector<int> &A) {
    // in case one element only
    long long max_end = A[0];
    long long max_slice = A[0];

    // start from 2nd element!
    for (int i = 1; i < (int)A.size(); i++) {
        long long num = A[i];
        max_end = max(num, max_end + A[i]);
        max_slice = max(max_slice, max_end);
    }
    return max_slice;
}
```

Analysis

Detected time complexity:

O(N)

test	time	result
example	0.020 s.	**OK**
one_element	0.020 s.	**OK**
two_elements	0.020 s.	**OK**
three_elements	0.020 s.	**OK**
simple	0.020 s.	**OK**
extreme_minimum	0.020 s.	**OK**
fifty_random	0.020 s.	**OK**
neg_const	0.020 s.	**OK**
pos_const	0.020 s.	**OK**
high_low_1Kgarbage	0.020 s.	**OK**
1Kgarbage_high_low	0.020 s.	**OK**
growing_saw	0.020 s.	**OK**
blocks	0.040 s.	**OK**
growing_negative	0.080 s.	**OK**

Training center