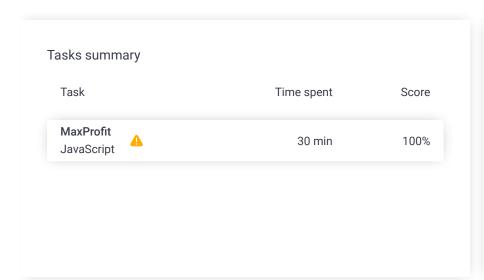
Codility_

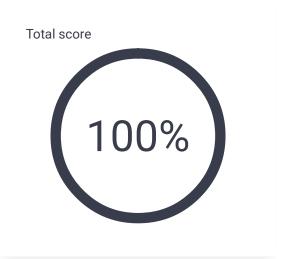
CodeCheck Report: trainingE2QDW3-SX8

Test Name:

Summary Timeline

Check out Codility training tasks





Tasks Details

1. MaxProfit

Easy

Given a log of stock prices compute the maximum possible earning.

Task Score

Correctness

100%

100%

Performance

100%

Task description

An array A consisting of N integers is given. It contains daily prices of a stock share for a period of N consecutive days. If a single share was bought on day P and sold on day Q, where $0 \le P \le Q < N$, then the *profit* of such transaction is equal to A[Q] - A[P], provided that $A[Q] \ge A[P]$. Otherwise, the transaction brings *loss* of A[P] - A[Q].

For example, consider the following array A consisting of six elements such that:

A[0] = 23171

A[1] = 21011

A[2] = 21123

A[3] = 21366

A[4] = 21013

A[5] = 21367

If a share was bought on day 0 and sold on day 2, a loss of 2048 would occur because A[2] – A[0] = 21123 – 23171 = –2048. If a share was bought on day 4 and sold on day 5, a profit of 354 would occur because A[5] – A[4] = 21367 – 21013 = 354.

Solution

Programming language used: JavaScript

Total time used: 30 minutes

Effective time used: 30 minutes

Notes: not defined yet

Task timeline ②

04:38:05 05:07:09

Maximum possible profit was 356. It would occur if a share was bought on day 1 and sold on day 5.

Write a function,

```
function solution(A);
```

that, given an array A consisting of N integers containing daily prices of a stock share for a period of N consecutive days, returns the maximum possible profit from one transaction during this period. The function should return 0 if it was impossible to gain any profit.

For example, given array A consisting of six elements such that:

```
A[0] = 23171
A[1] = 21011
A[2] = 21123
A[3] = 21366
A[4] = 21013
A[5] = 21367
```

the function should return 356, as explained above.

Write an efficient algorithm for the following assumptions:

- N is an integer within the range [0..400,000];
- each element of array A is an integer within the range [0..200,000].

Copyright 2009–2021 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

Test results - Codility

```
Code: 05:07:09 UTC, is, final,
                                     show code in pop-up
 score: 100
     // you can write to stdout for debugging purposes,
 2
     // console.log('this is a debug message');
     function solution(A) {
 3
 4
         var N = A.length;
 5
         var max_profit = 0;
 6
 7
         var min = 200001;
 8
 9
         for (var i = 0; i < N; i++) {
10
             if (min > A[i]) {
11
                  min = A[i];
12
             } else {
13
                  max_profit = Math.max(max_profit, A[i]
14
         }
15
16
17
18
         return max_profit;
19
     }
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: O(N)

ехра	nd all	Example tests	S
•	example example, length=6	1	√ OK
ехра	nd all	Correctness tes	sts
•	simple_1 V-pattern sequence	ee, length=7	√ OK
•	simple_desc descending and a length=5	scending sequence,	√ OK
•	simple_empty empty and [0,2000	000] sequence	√ OK
•	two_hills two increasing su	osequences	√ OK
•	ore_min	er_max_and_bef global maximum and mum	√ OK
ехра	nd all	Performance te	sts
•	• , ,	llowed by short V- om [15]) repeated	√ OK
>	large_1 large value (99) fo	llowed by short	√ OK

Test results - Codility

	pattern (values from [16]) repeated 10K times		
•	large_2 chaotic sequence of 200K values from [100K120K], then 200K values from [0100K]	√ OK	
•	large_3 chaotic sequence of 200K values from [1200K]	√ OK	