Codility_

Candidate Report: trainingR3D9ZU-JZ4

Check out Codility training tasks

Test Name:

Summary Timeline

Tasks summary

Task	Time spent	Score
MaxProfit Java 8 ⚠	42 min	88%

Total score

Identity verification
Add another layer of security with Identity Verification, learn more.

Tasks Details

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

Task Score	Correctness	Performance
88%	80%	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 ≤ P ≤ Q < N, then the *profit* of such transaction is equal to A[Q] − A[P], provided that A[Q] ≥ 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. Maximum possible profit was 356. It would occur if a share was bought on day 1 and sold on day 5.

Write a function,

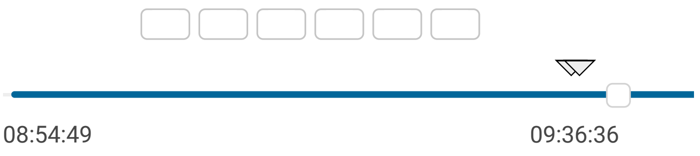
```
class Solution { public int solution(int[] A); }
```

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.

Solution

Programming language used:	Java 8
Total time used:	42 minutes
Effective time used:	42 minutes
Notes:	*not defined yet*

Task timeline



Code: 09:36:36 UTC, java, final, score: **88**

show code in pop-up

```
// you can also use imports, for example:
// import java.util.*;

// you can write to stdout for debugging purposes,
// System.out.println("this is a debug message");

class Solution {
    public int solution(int[] A) {
        int min=A[0];
        int mp=0;
        int p=0;
```

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].

```
        for(int i=0;i<A.length;i++){
            if(min>A[i]){
                min=A[i];
            }
            else{
                mp=A[i]-min;
                if(p<mp){
                    p=mp;
                }
            }


        }
        return p;
        // write your code in Java SE 8
    }
}
```

Analysis summary

The following issues have been detected: runtime errors.

For example, for the input [ ] the solution terminated unexpectedly.

Analysis

Detected time complexity: **O(N)**

expand all	Example tests
▸ example example, length=6	✓ OK

expand all	Correctness tests
▸ simple_1 V-pattern sequence, length=7	✓ OK
▸ simple_desc descending and ascending sequence, length=5	✓ OK
▸ simple_empty empty and [0,200000] sequence	✗ RUNTIME ERROR tested program terminated with exit code 1
▸ two_hills two increasing subsequences	✓ OK
▸ max_profit_after_max_and_before_min max profit is after global maximum and before global minimum	✓ OK

expand all	Performance tests
▸ medium_1 large value (99) followed by short V-pattern (values from [1..5]) repeated 100 times	✓ OK
▸ large_1 large value (99) followed by short pattern (values from [1..6]) repeated 10K times	✓ OK
▸ large_2 chaotic sequence of 200K values from [100K..120K], then 200K values from [0..100K]	✓ OK
▸ large_3 chaotic sequence of 200K values from [1..200K]	✓ OK

The PDF version of this report that may be downloaded on top of this site may contain sensitive data including personal information. For security purposes, we recommend you remove it from your system once reviewed.