

Summary

Timeline

Tasks summary

Task	Time spent	Score
Triangle Java 8	30 min	100%

Total score

100%

Tasks Details

Easy	1. Triangle	Determine whether a triangle can be built from a given set of edges.	Task Score	Correctness	Performance
			100%	100%	100%

Task description

An array A consisting of N integers is given. A triplet (P, Q, R) is *triangular* if  $0 \leq P < Q < R < N$  and:

- $A[P] + A[Q] > A[R]$ ,
- $A[Q] + A[R] > A[P]$ ,
- $A[R] + A[P] > A[Q]$ .

For example, consider array A such that:

A[0] = 10    A[1] = 2    A[2] = 5  
A[3] = 1    A[4] = 8    A[5] = 20

Triplet (0, 2, 4) is triangular.

Write a function:

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

that, given an array A consisting of N integers, returns 1 if there exists a triangular triplet for this array and returns 0 otherwise.

For example, given array A such that:

A[0] = 10    A[1] = 2    A[2] = 5  
A[3] = 1    A[4] = 8    A[5] = 20

the function should return 1, as explained above. Given array A such that:

Solution

Programming language used:	Java 8	
Total time used:	30 minutes	?
Effective time used:	30 minutes	?
Notes:	not defined yet	

Task timeline

05:45:0506:14:10

Code: 06:14:10 UTC, java, final, score: 100

```
1 // you can also use imports, for example:
2 // import java.util.*;
3
4 // you can write to stdout for debugging purposes,
```

A[0] = 10      A[1] = 50      A[2] = 5  
A[3] = 1

the function should return 0.

Write an **efficient** algorithm for the following assumptions:

- N is an integer within the range [0..100,000];
- each element of array A is an integer within the range [-2,147,483,648..2,147,483,647].

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

```
5 // System.out.println("this is a debug message");
6 import java.util.*;
7 class Solution {
8     public int solution(int[] A) {
9         int N = A.length;
10        if (N < 3) return 0;
11        Arrays.sort(A);
12
13        for (int i = 0; i < N - 2; i++) {
14            if (A[i] >= 0 && A[i] > A[i + 2] - A[i +
15
16                return 1;
17        }
18    }
19
20    return 0;
21 }
22 }
```

Analysis summary

The solution obtained perfect score.

Analysis

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

expand all	Example tests
▶ example	✓ OK example, positive answer, length=6
▶ example1	✓ OK example, answer is zero, length=4
expand all	Correctness tests
▶ extreme_empty	✓ OK empty sequence
▶ extreme_single	✓ OK 1-element sequence
▶ extreme_two_elems	✓ OK 2-element sequence
▶ extreme_negative1	✓ OK three equal negative numbers
▶ extreme_arith_overflow1	✓ OK overflow test, 3 MAXINTs
▶ extreme_arith_overflow2	✓ OK overflow test, 10 and 2 MININTs
▶ extreme_arith_overflow3	✓ OK overflow test, 0 and 2 MAXINTs
▶ medium1	✓ OK chaotic sequence of values from [0..100K], length=30
▶ medium2	✓ OK chaotic sequence of values from [0..1K], length=50
▶ medium3	✓ OK chaotic sequence of values from [0..1K], length=100
expand all	Performance tests
▶ large1	✓ OK chaotic sequence with values from

[0..100K], length=10K		
▶	large2 1 followed by an ascending sequence of ~50K elements from [0..100K], length=~50K	✓ OK
▶	large_random chaotic sequence of values from [0..1M], length=100K	✓ OK
▶	large_negative chaotic sequence of negative values from [-1M..-1], length=100K	✓ OK
▶	large_negative2 chaotic sequence of negative values from [-10..-1], length=100K	✓ OK
▶	large_negative3 sequence of -1 value, length=100K	✓ OK