codility

Check out Codility training tasks

Candidate Report: Anonymous

Test Name:

Summary Timeline

Test Score

68 out of 100 points

68%

Tasks in Test

	Time Spent	Task Score
Triangle Submitted in: Java 8	1 min	68%

TASKS DETAILS

		Task Score	Correctness	Performance
EASY	1. **Triangle** Determine whether a triangle can be built from a given set of edges.	68%	90%	33%

Task description

An array A consisting of N integers is given. A triplet (P, Q, R) is *triangular* if 0 ≤ 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:

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

the function should return 0.

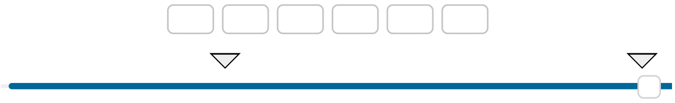
Write an **efficient** algorithm for the following assumptions:

- N is an integer within the range [0..100,000];

Solution

Programming language used:	Java 8
Total time used:	1 minutes
Effective time used:	1 minutes
Notes:	*not defined yet*

Task timeline



12:02:52 12:03:44

Code: 12:03:44 UTC, java, final, score: 68

show code in pop-up

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

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

class Solution {
    public static int solution(int[] a) {
                // System.out.println(a.length);
```

- each element of array A is an integer within the range [−2,147,483,648..2,147,483,647].

```

                if (a.length < 3)
                        return 0;

                for (int p = 0; p < a.length - 2; p++) {
                        // System.out.println(p);

                        for (int q = p + 1; q < a.length -
                                // System.out.println(p +

                                for (int r = q + 1; r < a.
                                        // System.out.prin
                                        if (ck(a[p], a[q],
                                                return 1;
                                }
                        }
                }
                return 0;
    }

        public static boolean ck(int ap, int aq, int ar) {
                return ap + aq > ar && aq + ar > ap && ar
        }
}
```

Analysis summary

The following issues have been detected: wrong answers, timeout errors.

Analysis ?

Detected time complexity: **O(N**3)**

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

length=100

expand all

	Performance tests	
▶	large1 chaotic sequence with values from [0..100K], length=10K	✓ OK
▶	large2 1 followed by an ascending sequence of ~50K elements from [0..100K], length=~50K	✗ TIMEOUT ERROR Killed. Hard limit reached: 6.000 sec.
▶	large_random chaotic sequence of values from [0..1M], length=100K	✓ OK
▶	large_negative chaotic sequence of negative values from [-1M..-1], length=100K	✗ TIMEOUT ERROR Killed. Hard limit reached: 7.000 sec.
▶	large_negative2 chaotic sequence of negative values from [-10..-1], length=100K	✗ TIMEOUT ERROR Killed. Hard limit reached: 6.000 sec.
▶	large_negative3 sequence of -1 value, length=100K	✗ TIMEOUT ERROR Killed. Hard limit reached: 6.000 sec.

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.