

CodeCheck Report: training225D8P-DYU

Test Name:

[Check out Codility training tasks](#)

Summary

Timeline

Tasks summary

Task	Time spent	Score
Triangle Python	13 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:

Solution

Programming language used: Python

Total time used: 13 minutes ?

Effective time used: 13 minutes ?

Notes: not defined yet

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

Triplet (0, 2, 4) is triangular.

Write a function:

```
def solution(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];
- each element of array A is an integer within the range [-2,147,483,648..2,147,483,647].

Copyright 2009–2023 by Codility Limited. All Rights Reserved.
Unauthorized copying, publication or disclosure prohibited.

Task timeline



07:40:15

07:52:45

Code: 07:52:44 UTC, py, [show code in pop-up](#)
final, score: 100

```

1  # you can write to stdout for debuggi
2  # print("this is a debug message")
3
4  def solution(A):
5      # Implement your solution here
6      # pass
7      if len(A) <= 2:
8          return 0
9
10     A.sort()
11
12     for i in range(0, len(A) - 2):
13         if A[i] + A[i+1] > A[i+2]:
14             return 1
15
16     return 0
17
18
```

Analysis summary

The solution obtained perfect score.

Analysis

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

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	✓ OK
▶ 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], length=100	✓ OK
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	✓ 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