



Candidate Report: trainingCM9TKJ-79Y

[Check out Codility training tasks](#)

Test Name:

Summary    Timeline

Tasks summary

Task	Time spent	Score
Triangle Python	10 min	100%

Total score

100%

Tasks Details

Easy	1. <b>Triangle</b>	Task Score	Correctness	Performance
	Determine whether a triangle can be built from a given set of edges.	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:



```
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

Solution

Programming language used:	Python	
Total time used:	10 minutes	
Effective time used:	10 minutes	
Notes:	not defined yet	

Task timeline

07:33:5707:43:47

Code: 07:43:46 UTC, py, final, score: 100

[show code in pop-up](#)

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 2009–2020 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

```
1  # you can write to stdout for debugging purposes, e.g.
2  # print("this is a debug message")
3
4  def solution(A):
5      # write your code in Python 3.6
6      A.sort(reverse=True)
7      for i in range(len(A)-2):
8          if A[i]-A[i+1]<A[i+2]:
9              return 1
10     return 0
11
```

Analysis summary

The solution obtained perfect score.

Analysis ?

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

collapse all		Example tests	
▼	example		✓ OK
example, positive answer, length=6			
1.	0.036 s	OK	
▼	example1		✓ OK
example, answer is zero, length=4			
1.	0.036 s	OK	
collapse all		Correctness tests	
▼	extreme_empty		✓ OK
empty sequence			
1.	0.036 s	OK	
2.	0.036 s	OK	
3.	0.036 s	OK	
4.	0.036 s	OK	
5.	0.036 s	OK	
6.	0.036 s	OK	
▼	extreme_single		✓ OK
1-element sequence			
1.	0.036 s	OK	
2.	0.036 s	OK	
3.	0.036 s	OK	
4.	0.036 s	OK	
5.	0.036 s	OK	
6.	0.036 s	OK	
▼	extreme_two_elems		✓ OK
2-element sequence			
1.	0.036 s	OK	

2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
▼ extreme_negative1 <span>✓ OK</span> three equal negative numbers		
1.	0.036 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
▼ extreme_arith_overflow1 <span>✓ OK</span> overflow test, 3 MAXINTs		
1.	0.036 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
▼ extreme_arith_overflow2 <span>✓ OK</span> overflow test, 10 and 2 MININTs		
1.	0.036 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
▼ extreme_arith_overflow3 <span>✓ OK</span> overflow test, 0 and 2 MAXINTs		
1.	0.036 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
▼ medium1 <span>✓ OK</span> chaotic sequence of values from [0..100K], length=30		
1.	0.036 s	OK
2.	0.036 s	OK
3.	0.036 s	OK

4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
<div>▼ medium2 <span>✓ OK</span></div> <div>chaotic sequence of values from [0..1K], length=50</div>		
1.	0.036 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
<div>▼ medium3 <span>✓ OK</span></div> <div>chaotic sequence of values from [0..1K], length=100</div>		
1.	0.036 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
collapse all <span>Performance tests</span>		
<div>▼ large1 <span>✓ OK</span></div> <div>chaotic sequence with values from [0..100K], length=10K</div>		
1.	0.048 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
<div>▼ large2 <span>✓ OK</span></div> <div>1 followed by an ascending sequence of ~50K elements from [0..100K], length=~50K</div>		
1.	0.084 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
<div>▼ large_random <span>✓ OK</span></div> <div>chaotic sequence of values from [0..1M], length=100K</div>		
1.	0.176 s	OK

2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
<b>▼ large_negative</b> <span>✓ OK</span> chaotic sequence of negative values from [-1M..-1], length=100K		
1.	0.184 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
<b>▼ large_negative2</b> <span>✓ OK</span> chaotic sequence of negative values from [-10..-1], length=100K		
1.	0.168 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	OK
<b>▼ large_negative3</b> <span>✓ OK</span> sequence of -1 value, length=100K		
1.	0.144 s	OK
2.	0.036 s	OK
3.	0.036 s	OK
4.	0.036 s	OK
5.	0.036 s	OK
6.	0.036 s	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.