Open Feedback Dialog

codility

Training center

Check out Codility training tasks

Demo ticket

Session

ID: demoW7M67S-ARJ
Time limit: 120 min.

Status: closed

Created on: 2014-03-15 03:37 UTC
Started on: 2014-03-15 03:38 UTC
Finished on: 2014-03-15 03:49 UTC

Tasks in test

Task score

Test score

100%

100 out of 100 points

MEDIUM

1. Triangle

Determine whether a triangle can be built from a given set of edges.

score: 100 of 100

Task description

A zero-indexed 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:

```
def solution(A)
```

that, given a zero-indexed 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.
Assume that:

- N is an integer within the range [0..1,000,000];
- each element of array A is an integer within the range [−2,147,483,648..2,147,483,647].

Complexity:

- expected worst-case time complexity is O(N*log(N));
- expected worst-case space complexity is O(N), beyond

Solution

Programming language used: Python

Total time used: 12 minutes

Effective time used: 12 minutes

Notes: correct functionality and scalability

Task timeline

03:38:13 03:49:59

Code: 03:49:59 UTC, py, final, score: **100.00**

```
01. def solution(A):
02.     # write your code in Python 2.6
03.     A.sort()
04.     size = len(A)
05.     for i in xrange(size-2):
06.         if A[i] <= 0:
07.             continue
08.         if A[i] + A[i+1] > A[i+2]:
09.             return 1
10.     return 0
11.
```

Analysis

Detected time complexity:

O(N*log(N))

test	time	result
example		

input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

example example, positive answer, length=6	0.050 s.	**OK**
example1 example, negative answer, length=4	0.050 s.	**OK**
example2 example, positive answer	0.050 s.	**OK**
example_grouped example, answer is zero	0.050 s.	**OK**
extreme_empty empty sequence + [5,3,3]	0.050 s.	**OK**
extreme_single 1-element sequence + [5,3,3]	0.050 s.	**OK**
extreme_two_elems 2-element sequence + [5,3,3]	0.050 s.	**OK**
extreme_negative1 three equal negative numbers	0.050 s.	**OK**
extreme_arith_overflow1 overflow test, 3 MAXINTs + [5,3,3]	0.050 s.	**OK**
extreme_arith_overflow2 overflow test, 10 and 2 MININTs + [5,3,3]	0.050 s.	**OK**
extreme_arith_overflow3 overflow test, 0 and 2 MAXINTs + [5,3,3]	0.050 s.	**OK**
medium1 chaotic sequence of values from [0..100K], length=30 + [1,5,10]	0.050 s.	**OK**
medium2 chaotic sequence of values from [0..1K], length=50 + [1,5,10]	0.050 s.	**OK**
medium3 chaotic sequence of values from [0..1K], length=100 + [1,5,10]	0.050 s.	**OK**
large1 chaotic sequence with values from [0..100K], length=10K + [1,5,10]	0.050 s.	**OK**
large2 1 followed by an ascending sequence of ~50K elements from [0..100K], length=~50K + [1,5,10]	0.050 s.	**OK**
large_random chaotic sequence of values from [0..1M], length=100K + [1,5,10]	0.050 s.	**OK**
large_negative chaotic sequence of negative values from [-1M..-1], length=100K + [1,5,10]	0.050 s.	**OK**
large_negative2 chaotic sequence of negative values from [-10..-1], length=100K + [5,3,3]	0.050 s.	**OK**
large_negative3 sequence of -1 value, length=100K + [5,3,3]	0.050 s.	**OK**

Training center