closed

Demo ticket

Session
ID: demoVVFYMY-SUX
Time limit: 120 min.

Status: closed
Started on: 2014-01-09 16:56 UTC

Score:

100

of 100

★★ 1. Triangle

score: 100 of 100

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

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:

```
class Solution { public int solution(int[]
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 input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

Solution

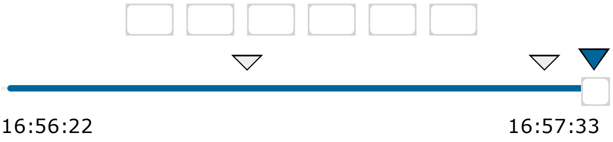
Programming language used: C#

Total time used: 2 minutes

Effective time used: 1 minutes

Notes: correct functionality and scalability

Task timeline



Code: 16:57:33 UTC, cs, final, score: 100.00

```
using System;
using System.Linq;
// you can also use other imports, for example:
// using System.Collections.Generic;
class Solution {
        public int solution(int[] A)
        {
            // write your code in C# with .NET 2.0
            if (A.Length < 0 || A.Length > 1000000) throw new ArgumentOutOfRangeException();
            var sorted = A.ToList<int>();
            sorted.Sort();
            for (var count = 0; count < sorted.Count - 2; count++)
            {
              if (sorted[count + 1] > sorted[count + 2] - sorted[count])
                return 1;
            }
            return 0;
        }
}
```

Analysis

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

test	time	result
example	0.090 s.	**OK**

example, positive answer, length=6		
example1 example, negative answer, length=4	0.090 s.	**OK**
example2 example, positive answer	0.090 s.	**OK**
example_grouped example, answer is zero	0.090 s.	**OK**
extreme_empty empty sequence + [5,3,3]	0.090 s.	**OK**
extreme_single 1-element sequence + [5,3,3]	0.090 s.	**OK**
extreme_two_elems 2-element sequence + [5,3,3]	0.090 s.	**OK**
extreme_negative1 three equal negative numbers	0.090 s.	**OK**
extreme_arith_overflow1 overflow test, 3 MAXINTs + [5,3,3]	0.090 s.	**OK**
extreme_arith_overflow2 overflow test, 10 and 2 MININTs + [5,3,3]	0.090 s.	**OK**
extreme_arith_overflow3 overflow test, 0 and 2 MAXINTs + [5,3,3]	0.090 s.	**OK**
medium1 chaotic sequence of values from [0..100K], length=30 + [1,5,10]	0.090 s.	**OK**
medium2 chaotic sequence of values from [0..1K], length=50 + [1,5,10]	0.090 s.	**OK**
medium3 chaotic sequence of values from [0..1K], length=100 + [1,5,10]	0.090 s.	**OK**
large1 chaotic sequence with values from [0..100K], length=10K + [1,5,10]	0.090 s.	**OK**
large2 1 followed by an ascending sequence of ~50K elements from [0..100K], length=~50K + [1,5,10]	0.090 s.	**OK**
large_random chaotic sequence of values from [0..1M], length=100K + [1,5,10]	0.090 s.	**OK**
large_negative chaotic sequence of negative values from [-1M..-1], length=100K + [1,5,10]	0.090 s.	**OK**
large_negative2 chaotic sequence of negative values from [-10..-1], length=100K + [5,3,3]	0.090 s.	**OK**
large_negative3 sequence of -1 value, length=100K + [5,3,3]	0.090 s.	**OK**

Get acco