codility

Check out Codility training tasks

Candidate Report: Anonymous

Test Name:

SUMMARY TIMELINE

Test Score

60 out of 100 points

60%

Tasks in Test

	Time Spent	Task Score
BinaryGap Submitted in: C#	47 min	60%

TASKS DETAILS

EASY

1. BinaryGap

Find longest sequence of zeros in binary representation of an integer.

Task Score	Correctness	Performance
60%	60%	Not assessed

Task description

A *binary gap* within a positive integer N is any maximal sequence of consecutive zeros that is surrounded by ones at both ends in the binary representation of N.

For example, number 9 has binary representation `1001` and contains a binary gap of length 2. The number 529 has binary representation `1000010001` and contains two binary gaps: one of length 4 and one of length 3. The number 20 has binary representation `10100` and contains one binary gap of length 1. The number 15 has binary representation `1111` and has no binary gaps. The number 32 has binary representation `100000` and has no binary gaps.

Write a function:

Solution

Programming language used:	C#
Total time used:	47 minutes
Effective time used:	47 minutes
Notes:	*not defined yet*

Task timeline

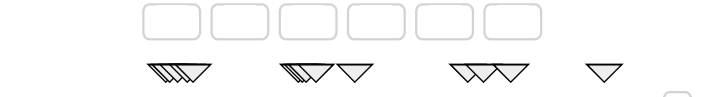
```
class Solution { public int solution(int
N); }
```

that, given a positive integer N, returns the length of its longest binary gap. The function should return 0 if N doesn't contain a binary gap.

For example, given N = 1041 the function should return 5, because N has binary representation `10000010001` and so its longest binary gap is of length 5. Given N = 32 the function should return 0, because N has binary representation '100000' and thus no binary gaps.

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

- N is an integer within the range [1..2,147,483,647].



10:43:26 11:29:43

Code: 11:29:43 UTC, cs, final, score: **60** show code in pop-up

```
1  using System;
2  // you can also use other imports, for example:
3  // using System.Collections.Generic;
4
5  // you can write to stdout for debugging purpose
6  // Console.WriteLine("this is a debug message");
7
8  class Solution {
9      public int solution(int N) {
10         var sequenceCounter = 0;
11         var iterations = Math.Floor(Math.Log(N,
12         var maxSequence = 0;
13         var startingOne = false;
14
15         for (int i = 0; i < iterations; i++)
16         {
17             // check if digit is 0
18             var isCero = (N & (1 << i)) == 0;
19
20             if (isCero)
21             {
22                 sequenceCounter++;
23             }
24             else
25             {
26                 if (startingOne && sequenceCount
27                     maxSequence = sequenceCounte
28                     sequenceCounter = 0;
29                 }
30                 startingOne = true;
31             }
32         }
33
34         return maxSequence;
35     }
36 }
```

Analysis summary

The following issues have been detected: wrong answers.

For example, for the input 6 the solution returned a wrong answer (got 1 expected 0).

Analysis

expand all **Example tests**

▶	example1 example test n=1041=10000010001_2	✓ OK
▶	example2 example test n=15=1111_2	✓ OK
▶	example3 example test n=32=100000_2	✓ OK

expand all **Correctness tests**

▶	extremes n=1, n=5=101_2 and n=2147483647=2**31-1	✓ OK
▶	trailing_zeroes n=6=110_2 and n=328=101001000_2	✗ WRONG ANSWER got 1 expected 0
▶	power_of_2 n=5=101_2, n=16=2**4 and n=1024=2**10	✓ OK
▶	simple1 n=9=1001_2 and n=11=1011_2	✓ OK
▶	simple2 n=19=10011 and n=42=101010_2	✗ WRONG ANSWER got 2 expected 1
▶	simple3 n=1162=10010001010_2 and n=5=101_2	✓ OK
▶	medium1 n=51712=110010100000000_2 and n=20=10100_2	✗ WRONG ANSWER got 10 expected 2
▶	medium2 n=561892=10001001001011100100_2 and n=9=1001_2	✗ WRONG ANSWER got 5 expected 3
▶	medium3 n=66561=10000010000000001_2	✓ OK
▶	large1 n=6291457=11000000000000000000001_2	✓ OK
▶	large2 n=74901729=100011101101110100011100001	✗ WRONG ANSWER got 5 expected 4
▶	large3 n=805306373=11000000000000000000000000101_2	✓ OK
▶	large4 n=1376796946=1010010000100000100000100010010_2	✗ WRONG ANSWER got 9 expected 5
▶	large5 n=1073741825=1000000000000000	✓ OK

000000000000001_2

▶ large6 ✓ OK

n=1610612737=1100000000000000

000000000000001_2