codility

Check out Codility training tasks

Candidate Report: Anonymous

Test Name:

SUMMARY TIMELINE

Test Score

100 out of 100 points

100%

Tasks in Test

	Time Spent	Task Score
BinaryGap Submitted in: C#	2 min	100%

TASKS DETAILS

EASY

1.
BinaryGap
Find longest sequence of zeros in binary representation of an integer.

Task Score	Correctness	Performance
100%	100%	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:	2 minutes
Effective time used:	2 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].

21:55:58 21:57:32

Code: 21:57:32 UTC, cs, final, score: **100**

show code in pop-up

```
using System;
// you can also use other imports, for example
// using System.Collections.Generic;

// you can write to stdout for debugging purpo
// Console.WriteLine("this is a debug message"

class Solution {
    public int solution(int N) {
        if (N <= 0)
                        {
                                throw new Argu
                        }

                        var digits = Convert.T

                        int biggestGap = 0;
                        int gap = 0;
                        var state = SearchStat
                        foreach (var digit in 
                        {
                                if (digit == '
                                {
                                        switch
                                        {











                                        }
                                }
                                else
                                {
                                        switch
                                        {













                                        }
                                }
                        }

                        return biggestGap;
    }

    public enum SearchState
        {
                Init,
                Found_1,
                Counting_0
        }
}
```

Analysis summary

The solution obtained perfect score.

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	✓ OK
▶	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	✓ OK
▶	simple3 n=1162=10010001010_2 and n=5=101_2	✓ OK
▶	medium1 n=51712=110010100000000_2 and n=20=10100_2	✓ OK
▶	medium2 n=561892=10001001001011100100_2 and n=9=1001_2	✓ OK
▶	medium3 n=66561=10000010000000001_2	✓ OK
▶	large1 n=6291457=110000000000000000	✓ OK

	00001_2	
▶	large2 n=74901729=100011101101110100011100001	✓ OK
▶	large3 n=805306373=110000000000000000000000000101_2	✓ OK
▶	large4 n=1376796946=1010010000100000100000100010010_2	✓ OK
▶	large5 n=1073741825=1000000000000000000000000000001_2	✓ OK
▶	large6 n=1610612737=1100000000000000000000000000001_2	✓ OK