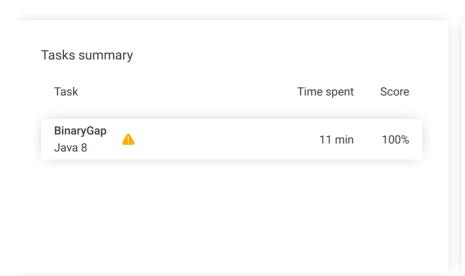
# Codility\_

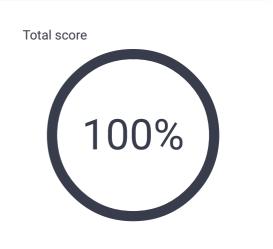
# CodeCheck Report: trainingTX4RCT-C8R

Test Name:

Summary Timeline

Check out Codility training tasks





## **Tasks Details**

## 1. BinaryGap

Find longest sequence of zeros in binary representation of an

Task Score

Correctness

100%

Performance

100% Not assessed

## Task description

integer.

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:

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

# Solution

Programming language used: Java 8

Total time used: 11 minutes

Effective time used: 11 minutes

Notes: not defined yet

11:33:31

Code: 11:44:08 UTC, java, show code in pop-up final, score: 100

1 // you can also use imports, for example:

import java.util.\*;

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].

Copyright 2009–2023 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

#### Test results - Codility

```
// you can write to stdout for debugging purpo
     // System.out.println("this is a debug message
 5
 6
7
     class Solution {
 8
             public int solution(int N) {
9
                     BitSet bs = BitSet.valueOf(nev
10
                     int from = bs.nextSetBit(0);
                     if (from == -1) {
11
12
                              return 0;
13
                     }
                     int maxBinaryGap = 0;
14
                     for (int end; (end = bs.nextSe
15
                             maxBinaryGap = Integer
16
17
18
                     return maxBinaryGap;
19
20
21
     }
```

# Analysis summary

The solution obtained perfect score.

# Analysis

lla	pse all Example test	S	
7	example1	~	ОК
	example test n=1041=10000010001_2		
1.	0.008 s <b>OK</b>		
▼	example2	V	OK
	example test n=15=1111_2		
1.	0.008 s <b>OK</b>		
▼	example3	V	OK
	example test n=32=100000_2		
1.	0.004 s <b>OK</b>		
olla	pse all Correctness te	sts	;
▼	extremes	~	ОК
	n=1, n=5=101_2 and		
	n=2147483647=2**31-1		
1.	0.004 s <b>OK</b>		
2.	0.004 s <b>OK</b>		
3.	0.004 s <b>OK</b>		
_	Augilian managa	_	OK
•	trailing_zeroes		OK
	n=6=110_2 and n=328=101001000_2		
1.	0.004 s <b>OK</b>		
2.	0.004 s <b>OK</b>		
•	power_of_2	V	OK
	n=5=101_2, n=16=2**4 and		
	n=1024=2**10		
1.	0.004 s <b>OK</b>		
2.	0.008 s <b>OK</b>		
3.	0.008 s <b>OK</b>		

Fest re	esults - Codility		
<b>V</b>	simple1 n=9=1001_2 and n=11=1011_2	<b>'</b>	OK
1.	0.004 s <b>OK</b>		
2.	0.004 s <b>OK</b>		
•	simple2 n=19=10011 and n=42=101010_2	~	ок
1.	0.004 s <b>OK</b>		
2.	0.008 s <b>OK</b>		
•	simple3 n=1162=10010001010_2 and n=5=101_2	~	OK
1.	0.004 s <b>OK</b>		
2.	0.004 s <b>OK</b>		
•	medium1 n=51712=110010100000000_2 and n=20=10100_2	~	OK
1.	0.004 s <b>OK</b>		
2.	0.004 s <b>OK</b>		
•	medium2 n=561892=100010010010111100100_2 and n=9=1001_2	~	ок
1.	0.008 s <b>OK</b>		
2.	0.004 s <b>OK</b>		
•	medium3 n=66561=1000001000000001_2	~	ОК
1.	0.004 s <b>OK</b>		
•	large1 n=6291457=1100000000000000000000000000000000000	~	ОК
1.	0.008 s <b>OK</b>		
•	large2 n=74901729=1000111011011101000 11100001	<b>✓</b>	OK
1.	0.004 s <b>OK</b>		
•	large3 n=805306373=110000000000000000000000000000000000	<b>✓</b>	ОК
1.	0.004 s <b>OK</b>		
▼	large4 n=1376796946=10100100001000001 00000100010010_2	✓	ОК
1.	0.004 s <b>OK</b>		
•	large5 n=1073741825=1000000000000000000000000000000000000	✓	OK
1.	0.004 s <b>OK</b>		

$\blacksquare$	large6	✓ OK
	n=1610612737=110000000000000000	
	0000000000001_2	
1.	0.008 s <b>OK</b>	