



CodeCheck Report: training8ZYNGC-KM6

[Check out Codility training tasks](#)

Test Name:

Summary

Timeline

Tasks summary

Task	Time spent	Score
BinaryGap Java 8	2 min	100%

Total score



Tasks Details

	1. BinaryGap	Task Score	Correctness	Performance
Easy	Find longest sequence of zeros in binary representation of an integer.		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:

```
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:

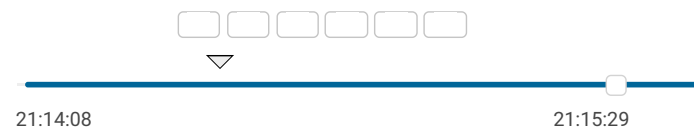
- N is an integer within the range [1..2,147,483,647].

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

Solution

Programming language used:	Java 8
Total time used:	2 minutes
Effective time used:	2 minutes
Notes:	not defined yet

Task timeline



Code: 21:15:28 UTC, java, final, score: 100 [show code in pop-up](#)

```
1 class Solution {
2     public int solution(int N) {
3         int maxLength = 0;
4         int end;
5         for (int start = nextSetBit(N, 0); start
6             maxLength = Integer.max(maxLength, end - start);
7         }
8         return maxLength;
9     }
10
11     /**
12      * Find the first bit "1" in a given number "N"
13      *
14      * @param N
15      * @param start
16      * @return
17      */
```

```
18         private static int nextSetBit(int N, int start)
19             for (; start < 32; start++) {
20                 if ((N & (1 << start)) != 0) {
21                     return start;
22                 }
23             }
24         return -1;
25     }
26 }
```

Analysis summary

The solution obtained perfect score.

Analysis

collapse all		Example tests	
▼	example1	✓ OK	
example test n=1041=10000010001_2			
1.	0.004 s	OK	
▼	example2	✓ OK	
example test n=15=1111_2			
1.	0.004 s	OK	
▼	example3	✓ OK	
example test n=32=100000_2			
1.	0.008 s	OK	
collapse all		Correctness tests	
▼	extremes	✓ OK	
n=1, n=5=101_2 and n=2147483647=2**31-1			
1.	0.008 s	OK	
2.	0.004 s	OK	
3.	0.008 s	OK	
▼	trailing_zeroes	✓ OK	
n=6=110_2 and n=328=101001000_2			
1.	0.008 s	OK	
2.	0.004 s	OK	
▼	power_of_2	✓ OK	
n=5=101_2, n=16=2**4 and n=1024=2**10			
1.	0.008 s	OK	
2.	0.008 s	OK	
3.	0.004 s	OK	
▼	simple1	✓ OK	
n=9=1001_2 and n=11=1011_2			
1.	0.004 s	OK	
2.	0.004 s	OK	
▼	simple2	✓ OK	
n=19=10011 and n=42=101010_2			
1.	0.004 s	OK	
2.	0.008 s	OK	
▼	simple3	✓ OK	
n=1162=10010001010_2 and n=5=101_2			
1.	0.008 s	OK	
2.	0.004 s	OK	
▼	medium1	✓ OK	
n=51712=110010100000000_2 and			

n=20=10100_2		
1.	0.008 s	OK
2.	0.008 s	OK
▼	medium2	✓ OK
n=561892=10001001001011100100_2 and n=9=1001_2		
1.	0.008 s	OK
2.	0.008 s	OK
▼	medium3	✓ OK
n=66561=10000010000000001_2		
1.	0.004 s	OK
▼	large1	✓ OK
n=6291457=110000000000000000001_2		
1.	0.008 s	OK
▼	large2	✓ OK
n=74901729=1000111011011101000111000 01		
1.	0.008 s	OK
▼	large3	✓ OK
n=805306373=1100000000000000000000 000101_2		
1.	0.008 s	OK
▼	large4	✓ OK
n=1376796946=10100100001000001000001 00010010_2		
1.	0.008 s	OK
▼	large5	✓ OK
n=1073741825=1000000000000000000000 00000001_2		
1.	0.004 s	OK
▼	large6	✓ OK
n=1610612737=1100000000000000000000 00000001_2		
1.	0.004 s	OK