Candidate Report: Anonymous

Test Name:

Summary   Timeline

Test Score

100 out of 100 points

100%

Tasks in Test

	Time Spent ⓘ	Task Score
FrogJmp
Submitted in: JavaS…	3 min	100%

TASKS DETAILS

EASY

1.
FrogJmp
Count minimal number of jumps from position X to Y.

Task Score	Correctness	Performance
100%	100%	100%

Task description

A small frog wants to get to the other side of the road. The frog is currently located at position X and wants to get to a position greater than or equal to Y. The small frog always jumps a fixed distance, D.

Count the minimal number of jumps that the small frog must perform to reach its target.

Write a function:

```
function solution(X, Y, D);
```

that, given three integers X, Y and D, returns the minimal number of jumps from position X to a position equal to or

Solution

Programming language used:	JavaScript
Total time used:	3 minutes ❓
Effective time used:	3 minutes ❓
Notes:	*not defined yet*

greater than Y.

For example, given:

```
X = 10
Y = 85
D = 30
```

the function should return 3, because the frog will be positioned as follows:

- after the first jump, at position 10 + 30 = 40
- after the second jump, at position 10 + 30 + 30 = 70
- after the third jump, at position 10 + 30 + 30 + 30 = 100

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

- X, Y and D are integers within the range [1..1,000,000,000];
- X ≤ Y.

Task timeline

16:05:21   16:07:45

Code: 16:07:45 UTC, js,   show code in pop-up
final, score: **100**

```js
1   // you can write to stdout for debugging purpo
2   // console.log('this is a debug message');
3
4   function solution(X, Y, D) {
5       if (X > Y || D === 0) {
6           return 0;
7       }
8
9       const result = (Y - X) / D;
10      return Math.ceil(result);
11  }
```

Analysis summary

The solution obtained perfect score.

Analysis ❓

Detected time complexity: O(1)

	Example tests	
► example		✓ OK
example test		
	Correctness tests	
► simple1		✓ OK
simple test		
► simple2		✓ OK
► extreme_position		✓ OK
no jump needed		
► small_extreme_jump		✓ OK
one big jump		
	Performance tests	
► many_jump1		✓ OK
many jumps, D = 2		
► many_jump2		✓ OK
many jumps, D = 99		

►	many_jump3 many jumps, D = 1283	✓ OK
►	big_extreme_jump maximal number of jumps	✓ OK
►	small_jumps many small jumps	✓ OK