Codility_

CodeCheck Report: trainingBSP2GN-B97

Test Name:

Check out Codility training tasks

Summary Timeline

Tasks summary

Task	Time spent	Score
FrogJmp Java 8 ⚠	19 min	100%

Total score

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:

```
class Solution { public int solution(int X, int
Y, int D); }
```

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

For example, given:

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

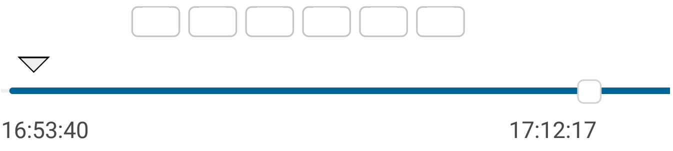
the function should return 3, because the frog will be positioned as follows:

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

Solution

Programming language used:	Java 8
Total time used:	19 minutes ?
Effective time used:	19 minutes ?
Notes:	*not defined yet*

Task timeline ?



Code: 17:12:16 UTC, java, final, score: **100**

show code in pop-up

```
// you can also use imports, for example:
// import java.util.*;

// you can write to stdout for debugging purposes,
```

- 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.

```
// System.out.println("this is a debug message");

class Solution {
 public int solution(int X, int Y, int D) {
 int distance = Y-X;
 double d=D;
 int jump = (int)Math.ceil(distance/(d));
 return jump;
 }
}
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: O(1)

expand all	Example tests
▶ example example test	✓ OK

expand all	Correctness tests
▶ simple1 simple test	✓ OK
▶ simple2	✓ OK
▶ extreme_position no jump needed	✓ OK
▶ small_extreme_jump one big jump	✓ OK

expand all	Performance tests
▶ many_jump1 many jumps, D = 2	✓ OK
▶ many_jump2 many jumps, D = 99	✓ OK
▶ many_jump3 many jumps, D = 1283	✓ OK
▶ big_extreme_jump maximal number of jumps	✓ OK
▶ small_jumps many small jumps	✓ OK