Codility_

Candidate Report: trainingYN9HFF-H4V

Check out Codility training tasks

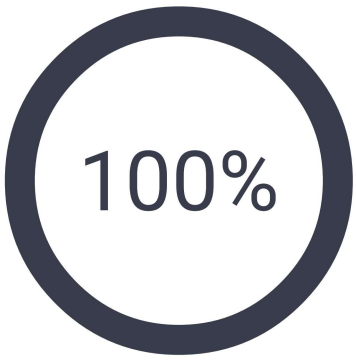
Test Name:

Summary Timeline Feedback

Tasks summary

Task	Time spent	Score
FrogJmp C#	1 min	100%

Total score



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:	C#
Total time used:	1 minutes
Effective time used:	1 minutes
Notes:	*not defined yet*

Task timeline

18:37:57 18:38:20

Code: 18:38:20 UTC, cs, final, score: **100**

show code in pop-up

- 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.


```
1  using System;
2  using System.Linq;
3  // you can also use other imports, for example:
4  // using System.Collections.Generic;
5
6  // you can write to stdout for debugging purposes, e.g.
7  // Console.WriteLine("this is a debug message");
8
9  class Solution {
10     public int solution(int X, int Y, int D)
11         {
12             if (X == Y)
13                 return 0;
14             var distance =(double)( Y - X );
15             double d = Math.Ceiling( distance / (double)D);
16             return System.Convert.ToInt32(d);
17         }
18 }
```

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