codility

**Congratulations**

You have completed a Codility training test.

Tweet this!
I scored 100% in #java on @Codility!
https://codility.com/demo/take-sample-test/count_div/

Sign up for our newsletter!

Like us on Facebook!

# Training ticket

## Session

**ID:** training7U794F-AXD
**Time limit:** 120 min.

## Status: closed

**Created on:** 2016-06-22 17:51 UTC
**Started on:** 2016-06-22 17:51 UTC
**Finished on:** 2016-06-22 17:52 UTC

| Tasks in test | Correctness | Performance | Task score |
|---|---|---|---|
| 1 CountDiv<br>Submitted in: Java | 100% | 100% | 100% |

Test score

**100%**

100 out of 100 points

EASY

## 1. CountDiv

Compute number of integers divisible by k in range [a..b].

score: 100 of 100

### Task description

Write a function:

```
class Solution { public int solution(int A, int B,
int K); }
```

that, given three integers A, B and K, returns the number of integers within the range [A..B] that are divisible by K, i.e.:

$$\{ i : A \leq i \leq B, i \bmod K = 0 \}$$

For example, for A = 6, B = 11 and K = 2, your function should return 3, because there are three numbers divisible by 2 within the range [6..11], namely 6, 8 and 10.

Assume that:

- A and B are integers within the range [0..2,000,000,000];
- K is an integer within the range [1..2,000,000,000];
- A ≤ B.

Complexity:

- expected worst-case time complexity is O(1);
- expected worst-case space complexity is O(1).



### Solution

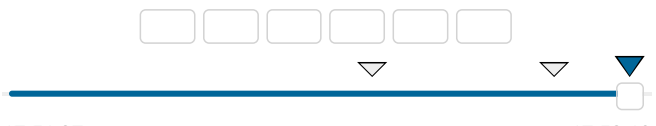
**Programming language used:** Java

**Total time used: 1 minutes**

**Effective time used: 1 minutes**

**Notes:** *not defined yet*

**Task timeline**



17:51:37 17:52:19

Code: 17:52:18 UTC, java, final, score: **100**

show code in pop-up

```
1  import java.util.function.Function;
2
3  class Solution {
4      public int solution(int A, int B, int K) {
5          int start = getStartValue(A, K);
6          int end = getEndValue(B, K);
7          return (end-start)/K+1;
```

```
8       }
9
10      protected int getStartValue(int start, int divisor)
11          return getDivisibleValue(start, divisor, (value
12      }
13
14      protected int getEndValue(int end, int divisor) {
15          return getDivisibleValue(end, divisor, (value)
16      }
17
18      private int getDivisibleValue(int value, int divis
19          while(value%divisor != 0){
20              value = func.apply(value);
21          }
22          return value;
23      }
24  }
```

### Analysis summary

The solution obtained perfect score.

### Analysis

Detected time complexity:
**O(1)**

| expand all | Example tests | |
|---|---|---|
| ▶ | example<br>A = 6, B = 11, K = 2 | ✔ OK |
| expand all | Correctness tests | |
| ▶ | simple<br>A = 11, B = 345, K = 17 | ✔ OK |
| ▶ | minimal<br>A = B in {0,1}, K = 11 | ✔ OK |
| ▶ | extreme_ifempty<br>A = 10, B = 10, K in {5,7,20} | ✔ OK |
| ▶ | extreme_endpoints<br>verify handling of range endpoints, multiple runs | ✔ OK |
| expand all | Performance tests | |
| ▶ | big_values<br>A = 100, B=123M+, K=2 | ✔ OK |
| ▶ | big_values2<br>A = 101, B = 123M+, K = 10K | ✔ OK |
| ▶ | big_values3<br>A = 0, B = MAXINT, K in {1,MAXINT} | ✔ OK |
| ▶ | big_values4<br>A, B, K in {1,MAXINT} | ✔ OK |

Training center