Codility_

Candidate Report: trainingGRSJJQ-HHC

Check out Codility training tasks

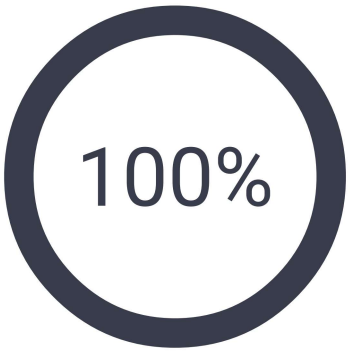
Test Name:

Summary Review (0) Timeline

Tasks summary

Task	Time spent	Score
CountFactors Java 8	26 min	100%

Total score



Tasks Details

Easy

1. CountFactors

Count factors of given number n.

Task Score	Correctness	Performance
100%	100%	100%

Task description

A positive integer D is a *factor* of a positive integer N if there exists an integer M such that N = D * M.

For example, 6 is a factor of 24, because M = 4 satisfies the above condition (24 = 6 * 4).

Write a function:

```
class Solution { public int solution(int N); }
```

that, given a positive integer N, returns the number of its factors.

For example, given N = 24, the function should return 8, because 24 has 8 factors, namely 1, 2, 3, 4, 6, 8, 12, 24. There are no other factors of 24.

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

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

Solution

Programming language used:	Java 8
Total time used:	26 minutes
Effective time used:	26 minutes
Notes:	*not defined yet*

Task timeline

08:32:44 08:58:10

Code: 08:58:10 UTC, java, final, score: **100** show code in pop-up

```
1  // you can also use imports, for example:
2  // import java.util.*;
3
4  // you can write to stdout for debugging purposes, e.g.
5  // System.out.println("this is a debug message");
6  import java.util.List;
7  import java.util.ArrayList;
8  class Solution {
9      public int solution(int N) {
```

```
10          List<Integer> list = new ArrayList<>();
11          if(N<1)
12              return 0;
13          for (int i = 1; i <= Math.sqrt(N); i++) {
14              if (N % i == 0) {
15                  if (N / i == i) {
16                      list.add(i);
17                  }
18                  else {
19                      list.add(i);
20                      list.add(N/i);
21                  }
22              }
23          }
24          return list.size();
25      }
26  }
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: **O(sqrt(N))**

expand all	Example tests	
▶	example1 example test (N=24=4!)	✓ OK
expand all	**Correctness tests**	
▶	squares N=16, N=36	✓ OK
▶	tiny N <= 10	✓ OK
▶	simple1 N=41(prime), N=42	✓ OK
▶	simple2 N=69, N=64, N=120=5!	✓ OK
▶	simple3 N=720=6!, N=1111	✓ OK
▶	simple4 N=5,040=7!, N=12,345	✓ OK
▶	simple5 N=34,879, N=40,320=8!	✓ OK
▶	extreme_one N=1	✓ OK
expand all	**Performance tests**	
▶	medium1 N=362,880=9!, N=1,948,102	✓ OK
▶	medium2 N=3,628,800=10!, N=5,621,892, N=4,999,696	✓ OK
▶	big1 N=27,043,111, N=39,916,800=11!, N = 39,992,976	✓ OK
▶	big2 N=97,093,212, N=2^28	✓ OK
▶	big3 N=479,001,600=12!, N=780291637(prime), N=449,991,369	✓ OK
▶	extreme_maxint N=1,000,000,000, N=MAX_INT, N=2147,395,600	✓ OK