Codility_

CodeCheck Report: trainingKWXR24-B23

Test Name:

Check out Codility training tasks

Summary   Timeline

Tasks summary

Task		Time spent	Score
CountSemiprimes C#	⚠	4 min	100%

Total score

100%

Tasks Details

Medium

1. CountSemiprimes

Count the semiprime numbers in the given range [a..b]

Task Score	Correctness	Performance
100%	100%	100%

Task description

A *prime* is a positive integer X that has exactly two distinct divisors: 1 and X. The first few prime integers are 2, 3, 5, 7, 11 and 13.

A *semiprime* is a natural number that is the product of two (not necessarily distinct) prime numbers. The first few semiprimes are 4, 6, 9, 10, 14, 15, 21, 22, 25, 26.

You are given two non-empty arrays P and Q, each consisting of M integers. These arrays represent queries about the number of semiprimes within specified ranges.

Query K requires you to find the number of semiprimes within the range (P[K], Q[K]), where 1 ≤ P[K] ≤ Q[K] ≤ N.

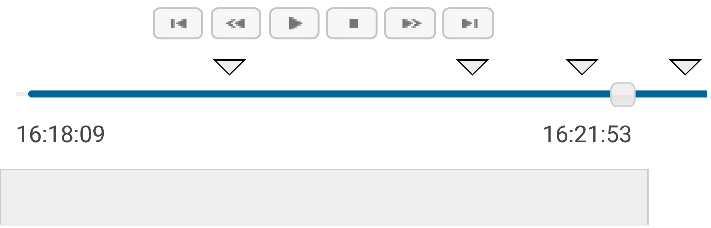
For example, consider an integer N = 26 and arrays P, Q such that:

```
    P[0] = 1    Q[0] = 26
    P[1] = 4    Q[1] = 10
    P[2] = 16   Q[2] = 20
```

Solution

Programming language used:	C#
Total time used:	4 minutes
Effective time used:	4 minutes
Notes:	*not defined yet*

Task timeline



The number of semiprimes within each of these ranges is as follows:

- (1, 26) is 10,
- (4, 10) is 4,
- (16, 20) is 0.

Write a function:

```
class Solution { public int[] solution(int N,
int[] P, int[] Q); }
```

that, given an integer N and two non-empty arrays P and Q consisting of M integers, returns an array consisting of M elements specifying the consecutive answers to all the queries.

For example, given an integer N = 26 and arrays P, Q such that:

```
P[0] = 1    Q[0] = 26
P[1] = 4    Q[1] = 10
P[2] = 16   Q[2] = 20
```

the function should return the values [10, 4, 0], as explained above.

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

- N is an integer within the range [1..50,000];
- M is an integer within the range [1..30,000];
- each element of arrays P and Q is an integer within the range [1..N];
- P[i] ≤ Q[i].

Code: 16:21:52 UTC, cs, final, score: 100

show code in pop-up

```
using System;
using System.Linq;
using System.Collections.Generic;

/**
 * 11.2 - Count Semi Primes
 * Paulo Santos
 * 19.Dec.2022
 */
class Solution {
    public int[] solution(int N, int[] P, int[] Q)

        /*
         * Calculate all the primes up to (N / 2 +
         */
        var primes = new List<int>();
        var sieve = new bool[N / 2 + 2];
        sieve[0] = sieve[1] = true;
        for(var i = 2; i < sieve.Length; i++) {
            for(var j = i * i; j < sieve.Length; j
                sieve[j] = true;
            }
        }
        for(var i = 0; i < sieve.Length; i++)
            if (!sieve[i])
                primes.Add(i);


        /*
         * Calculate all the semi-primes
         */
        var semi = new int[N + 1];
        for (var i = 0; i < primes.Count; i++)
            for (var j = 0; j < primes.Count; j++)
                var sp = primes[i] * primes[j];
                if (sp <= N)
                    semi[sp] = 1;
            }
        var aux1 = new int[N + 1];
        for (var i = 1; i < N + 1; i++)
            aux1[i] = semi[i] + aux1[i - 1];

        /*
         * Count the semi-primes
         */
        var res = new int[P.Length];
        for(var i = 0; i < res.Length; i++)
            res[i] = aux1[Q[i]] - aux1[P[i] - 1];

        return res;
    }
}
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: O(N * log(log(N)) + M)

expand all	Example tests
▶ example example test	✓ OK

expand all	Correctness tests
▶ extreme_one small N = 1	✓ OK
▶ extreme_four small N = 4	✓ OK
▶ small_functional small functional	✓ OK
▶ small_random small random, length = ~40	✓ OK

expand all	Performance tests
▶ medium_random small random, length = ~300	✓ OK
▶ large_small_slices large with very small slices, length = ~30,000	✓ OK
▶ large_random1 large random, length = ~30,000	✓ OK
▶ large_random2 large random, length = ~30,000	✓ OK
▶ extreme_large all max ranges	✓ OK