RESPECTABLE

CommonPrimeDivisors

START

Check whether two numbers have the same prime divisors.

Programming language: C++

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 prime D is called a *prime divisor* of a positive integer P if there exists a positive integer K such that D * K = P. For example, 2 and 5 are prime divisors of 20.

You are given two positive integers N and M. The goal is to check whether the sets of prime divisors of integers N and M are exactly the same.

For example, given:

- N = 15 and M = 75, the prime divisors are the same: {3, 5};
- N = 10 and M = 30, the prime divisors aren't the same: {2, 5} is not equal to {2, 3, 5};
- N = 9 and M = 5, the prime divisors aren't the same: {3} is not equal to {5}.

Write a function:

```
int solution(vector<int> &A, vector<int> &B);
```

that, given two non-empty zero-indexed arrays A and B of Z integers, returns the number of positions K for which the prime divisors of A[K] and B[K] are exactly the same.

For example, given:

```
A[0] = 15   B[0] = 75
A[1] = 10   B[1] = 30
A[2] = 3    B[2] = 5
```

the function should return 1, because only one pair (15, 75) has the same set of prime divisors.

Assume that:

- Z is an integer within the range [1..6,000];
- each element of arrays A, B is an integer within the range [1..2,147,483,647].

Complexity:

- expected worst-case time complexity is $O(Z*\log(\max(A)+\max(B))^2)$;
- expected worst-case space complexity is O(1), beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

Future training