QN Link : <https://leetcode.com/problems/count-array-pairs-divisible-by-k/>

Observation :

Scenario 1 :

* If a pair is divisible of K , then it have a multiple of K.

Scenario 2 :

* It is a divisor of K , other than 1

Example :

[2 , 3 ,4 , 5] k= 6

* (2 , 3)
* (3 , 4)

These are the 2 pairs which are divisible by K , but either of the number is multiple of 6.But how ?

* GCD (6 , 2) and GCD (3 , 6) are 2 , 3
* If a \* b is divisible by K , then [ GCD (a, K) and GCD (b , K) ] is also divisible by K.

So now we re reducing the search space from k to gcd(a , k) \* gcd (b , k)

So instead of putting the number , put the GCD ( X , K) in the map

Step 1 : Create an empty hashmap and an variable count

Step 2 : For each iteration , find the GCD of the number with K

Step 3 : Iterate over the key of the Map & cheack if the key of the map \* current element is divisible by K.

Step 4 : If so , add the occurrence to the map.

Step 5 : Put the GCD value into the map.

class Solution {

    public long countPairs(int[] nums, int k) {

        long result = 0;

        Map<Integer, Integer> gcdMap = new HashMap<>();

        for (int i = 0; i < nums.length; i++) {

            int gcd = findGcd(nums[i], k);

            for (int num : gcdMap.keySet()) {

                if ((long) gcd \* num % k == 0) {

                    result += gcdMap.get(num);

                }

            }

            gcdMap.put(gcd, gcdMap.getOrDefault(gcd, 0) + 1);

        }

        return result;

    }

    private int findGcd(int x, int y) {

        if (x < y) {

            return findGcd(y, x);

        }

        return y == 0 ? x : findGcd(y, x % y);

    }

}