2183. Count Array Pairs Divisible by K

Given a **0-indexed** integer array `nums` of length `n` and an integer `k`, return *the* ***number of pairs*** `(i, j)` *such that:*

- `0 <= i < j <= n - 1` *and*
- `nums[i] * nums[j]` *is divisible by* `k`.

Example 1:

```
Input: nums = [1,2,3,4,5], k = 2
Output: 7
Explanation: 
The 7 pairs of indices whose corresponding products are divisible by 2 are
(0, 1), (0, 3), (1, 2), (1, 3), (1, 4), (2, 3), and (3, 4).
Their products are 2, 4, 6, 8, 10, 12, and 20 respectively.
Other pairs such as (0, 2) and (2, 4) have products 3 and 15 respectively, which
are not divisible by 2.
```

Example 2:

```
Input: nums = [1,2,3,4], k = 5
Output: 0
Explanation: There does not exist any pair of indices whose corresponding product
is divisible by 5.
```

Constraints:

- `1 <= nums.length <=` 10^5
- `1 <= nums[i], k <=` 10^5

2183. Count Array Pairs Divisible by K

```
/*
    Brute force
    Time complexity: O(n^2)
    Space complexity: O(1)
*/
typedef std::vector<int> vi;
typedef long long ll;
class Solution {
    public:
        ll countPairs(vi& nums, int k) {
            int n=nums.size();
            ll ans=0;
            for(int i=0;i<n-1;++i){
                for(int j=i+1;j<n;++j){
                    if(nums[i]*1ll*nums[j]*1ll%k==0) ans++;
                }
            }
            return ans;
        }
};
```

2183. Count Array Pairs Divisible by K

```
/*
    Hashing+GCD
    Time complexity: $O(n.\sqrt{(k)})$
    Space complexity: $O(\sqrt{(k)})$
*/
typedef std::vector<int> vi;
typedef long long ll;
typedef std::vector<ll> vll;
class Solution {
    public:
        ll countPairs(vi& nums, int k) {
            std::unordered_map<ll,ll> gcd_freq;
            ll ans=0;
            for(auto& a: nums){
                ll gcd_ak=std::gcd(a,k);
                for(auto& [g,f]: gcd_freq) ans+=(gcd_ak*g%k==0)?f:0;
                gcd_freq[gcd_ak]++;
            }
            return ans;
        }
};
```