4. Given a 0-indexed integer array nums of length n and an integer k, return *the number of pairs* (i, j) *where* 0 <= i < j < n, *such that* nums[i] == nums[j] *and* (i * j) *is divisible by* k.

Example 1:
Input: nums = [3,1,2,2,2,1,3], k = 2
Output: 4
Explanation:
There are 4 pairs that meet all the requirements:
- nums[0] == nums[6], and 0 * 6 == 0, which is divisible by 2.
- nums[2] == nums[3], and 2 * 3 == 6, which is divisible by 2.
- nums[2] == nums[4], and 2 * 4 == 8, which is divisible by 2.
- nums[3] == nums[4], and 3 * 4 == 12, which is divisible by 2.

Example 2:
Input: nums = [1,2,3,4], k = 1
Output: 0
Explanation: Since no value in nums is repeated, there are no pairs (i,j) that meet all the requirements.

A**. Program**:

```
def countPairs(nums, k):
    n = len(nums)
    count = 0
    frequency_map = {}

    # Step 1: Build frequency map of elements in nums
    for num in nums:
        if num in frequency_map:
            frequency_map[num] += 1
        else:
            frequency_map[num] = 1

    for i in range(n):
        for j in range(i + 1, n):
            if nums[i] == nums[j]:
                if (i * j) % k == 0:
                    count += 1

    return count
nums = [3, 1, 2, 2, 2, 1, 3]
k = 2
print(countPairs(nums, k))  # Output: 4
```

Output:

```
4

=== Code Execution Successful ===
```

Timecomplexity: O(n^2)