UCF Local Contest — September 3, 2016

Count the Dividing Pairs
filename: `divide`
(*Difficulty Level:* `Medium`)

Number Theory provides many fascinating properties. You have most likely written programs dealing with different groups of numbers such as Prime, Perfect, Amicable, Happy, Powerful, and Untouchable numbers, just to name a few. In this problem, you'll attack yet another fascinating property of numbers, one dealing with pairs of numbers.

An integer D is said to be a proper divisor of an integer N if $D \neq N$ and there exist an integer Q such that N = Q * D. For example, 4 is a proper divisor of 8 and 5 is a proper divisor of 15, but 9 is not a proper divisor of 9 and 6 is not a proper divisor of 8. Note that zero is not a proper divisor of any number but all numbers (except zero) are proper divisors of zero.

We will call (D, N) as defined above "proper dividing pairs".

The Problem:

Given a list of integers A = {A_1, A_2, …, A_p}, you are to determine (count) the number of proper dividing pairs (A_i, A_j), where $1 \leq i, j \leq p$.

The Input:

The first input line contains a positive integer, *n*, indicating the number of test cases to process. Each test case starts with an integer, *p* $(2 \leq p \leq 10^6)$, indicating the number of integers in the list. The following input line will provide *p* integers, A_i $(0 \leq A_i \leq 10^7)$.

The Output:

For each test case, print "`Test case #`*t*`:` *m*", where *t* indicates the case number starting with 1 and *m* indicates the number of proper dividing pairs. Leave a blank line after the output for each test case.

Note that, as illustrated in Sample Input/Output, duplicate values in the input list are considered as different elements in the list and they each contribute to the total count (proper dividing pairs).

Sample Input:

```
5
3
1 2 3
4
1 2 3 1
2
7 5
3
29 0 17
10
32 16 8 4 2 2 4 8 16 32
```

Sample Output:

```
Test case #1: 2

Test case #2: 4

Test case #3: 0

Test case #4: 2

Test case #5: 40
```