Average Revisited

Project Explanation

Last week, you used a loop and an accumulation pattern to calculate the average of a sequence of numbers provided by a user. You asked the user to tell you how many numbers would be entered. In this program, you will collect the numbers and compute the average without knowing how many numbers to expect. As we move forward in the semester, you will find that there are many ways to accomplish the specified goal. In solving this problem, you may need to consider multiple strategies and choose the best one. The program you submit must be Pythonic.

```
Program to compute the average of the numbers provided.

Enter each number followed by <enter>.
When you are done, just hit <enter> in response to the prompt.

Enter a number: 10
Enter a number: 15
Enter a number: 5
Enter a number: 10
Enter a number:

You entered 4 numbers.
The average is 10.0.
```

HINTS

1. When a user hits `<enter>` without typing any text, `input()` returns an empty string (called a null string). If checked as a boolean, a null string will be interpreted as `False`. A string containing characters will be interpreted as `True`. You can use this property to control a `while` loop.
2. As we gain knowledge, we also gain the ability to evaluate different solution methods. Here, you could keep a running total as the user enters each number OR you could accumulate the numbers in a list and process them after exiting the loop. You could accumulate the response strings through concatenation and convert that to a list using `split()` or you could accumulate the response strings in a list or you could convert the response strings to floats and append the floats to a list. Which results in the most readable code? What circumstances would prefer one solution over the other?

CHALLENGE

My solution is 6 lines of code (with announcement and reporting but without the comments). You may need to use the walrus operator (`:=`) to get your code that small. Without the walrus operator, add two more lines (8 total).