|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| LEARNING PROFILE FOR ComputeAverage | | | | | |
| *Name* | *:* | *Tyler Lucas* | *Due Date* | *:* | *N/A* |
| *Student ID* | *:* | *3305203* | *Submission Date* | *:* | *N/A* |

# Problem Statement

Textbook example program.

# Description of the Code

This program reads a sequence of positive integers input by the user, and it will print out the average of those integers. The user is prompted to enter one integer at time. The user must enter a 0 to mark the end of the data. (The zero is not counted as part of the data to be averaged.) The program does not check whether the user’s input is positive, so it will actually add up both positive and negative input values.

# Errors and Warnings

No errors.

# Sample Input and Output

[Version 1.1]

Enter integers (including 0 and negative integers) separated by a carriage return [enter key], and end with two carriage returns.

1

2

3

4

5

6

7

8

9

10

You entered 10 integers.

Their average is 5.500.

# Discussion

I had to look closely at the author’s input/output helper class *TextIO* to figure out which methods to use to both count sequential carriage returns and read integer values without missing any of them. His thorough documentation of each method, even many of the private ones, made this much easier than it would have been without, highlighting the importance of good documentation and commenting. I ended up using eoln() to check for carriage returns with getln() to look at the next character without skipping an integer, and getInt() to get the next integer without skipping a carriage return. It seems to work quite well, making the user interface and input easier and more natural, and I hope to use it again. I may need to learn how to implement it without *TextIO*, as the author has stated that it is primarily for instruction, not professional use.