

Program #3

Due Jan 30 by 11:59pm **Points** 50 **Submitting** a file upload **File Types** asm

Due: Week 4, Sunday 11:59 PM Pacific USA time zone

Objectives

1. Implementing data validation
2. Implementing an accumulator
3. Integer arithmetic
4. Defining variables (integer and string)
5. Using library procedures for I/O
6. Implementing control structures (decision, loop)

Problem Definition

Write a MASM program to perform the following tasks:



Display the program title and programmer's name.

Get the user's name, and greet the user.

- Display instructions for the user
- Repeatedly prompt the user to enter the number. Validate the user input to be in [-100, -1] (inclusive). Count and accumulate the valid user numbers until a non-negative number is entered (The non-negative number is discarded)
- Calculate the (rounded integer) average of the negative numbers
- Display:
 - the number of negative numbers entered (Note: if no negative numbers were entered, display a special message and display the goodbye message with the user's name at the end)
 - the sum of negative numbers entered
 - the average, rounded to the nearest integer (e.g., -20.5 rounds to -21)
 - a goodbye message that includes the user's name, and terminate the program.

Requirements

1. The programmer's name and the user's name must appear in the output.
2. The **main** procedure must be modularized into commented logical sections (procedures are not required this time).
3. Recursive solutions are not acceptable for this assignment. This one is about iteration.
4. The program must be fully documented. This includes a complete header block for identification, description, etc., and a comment outline to explain each section of code.
5. The lower limit must be defined and used as a constant.
6. The usual requirements regarding documentation, readability, user-friendliness, etc., apply.

Notes

1. There are no new concepts in this programming assignment. It is given for extra practice, to keep MASM fresh in your mind while we study internal/external data representation and error detection/correction.
2. This is an integer program. Even though it would make more sense to use floating-point computations, you are required to do this one with integers.

Example Program Operation

```
Welcome to the Integer Accumulator by Austin Miller
What's your name? Roger
Hello, Roger

▶ Please enter numbers in [-100, -1].
Enter a non-negative number when you are finished to see results.
Enter number: -15
Enter number: -100
Enter number: -36
Enter number: -200
Invalid number, please enter numbers in [-100, -1].
Enter number: -10
Enter number: 0
You entered 4 valid numbers.
The sum of your valid numbers is -161
The rounded average is -40
Thank you for playing Integer Accumulator!
Goodbye, Roger.
```

Extra Credit Option (original definition must be fulfilled)

- (1 pt) Calculate and display the average as a floating-point number, rounded to the nearest .001.

Remember, in order to ensure you receive credit for any extra credit work, you must add one print statement to your program output PER EXTRA CREDIT which describes the extra credit you chose to work on. You will not receive extra credit points unless you do this. The statement must be formatted as follows...

```
--Program Intro--  
**EC: DESCRIPTION  
--Program prompts, etc--
```


Please refer back to the documentation for Program 1 to see a sample of the extra credit format.

Program 3 Rubric



Criteria	Ratings		Pts
<p>Files Correctly Submitted</p> <p>Submitted file is correct assignment and is an individual .asm file.</p>	<p>1 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	<p>1 pts</p>
<p>Program Assembles & Links</p> <p>Submitted program assembles and links without need for clarifying work for TA and/or messages to the student.</p> <p>This assumes the program is actually an attempt at the assignment. Non-attempts which compile/link earn no points.</p>	<p>2 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	<p>2 pts</p>
<p>Documentation - Identification Block - Header</p> <p>Name, Date, Program number, etc as per syllabus are included in Identification Block</p>	<p>1 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	<p>1 pts</p>
<p>Documentation - Identification Block - Program Description</p> <p>Description of functionality and purpose of program is included in Identification block.</p>	<p>2 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	<p>2 pts</p>
<p>Documentation - Section Comments</p> <p>Code section headers describe functionality and implementation of program flow. Should mirror the style guide image.</p>	<p>4 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	<p>4 pts</p>

Criteria	Ratings		Pts
<p>Documentation - In-line Comments</p> <p>In-line comments contribute to understanding of program flow (from section comments) but are not line-by-line descriptions of moving memory to registers.</p>	<p>1 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	1 pts
<p>Verification - Program Executes</p> <p>Program executes and makes some attempt at the assigned functionality.</p>	<p>5 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	5 pts
<p>Completeness - Displays Programmer Name</p> <p>Program prints out the programmer's name.</p>	<p>1 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	1 pts
<p>Completeness - Gets / Uses User's name</p> <p>Receives input with ReadString. Saves input in a null-terminated BYTE array. Greets user (e.g. "Hello, Username")</p>	<p>2 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	2 pts
<div>▶</div> <p>Completeness - Displays Introduction</p> <p>Displays program introduction. Program introduction should describe functionality of program.</p>	<p>1 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	1 pts
<p>Completeness - Prompt for Input</p> <p>Prompts user to enter data, specifying bounds of acceptable inputs.</p>	<p>2 pts</p> <p>Full Marks</p>	<p>0 pts</p> <p>No Marks</p>	2 pts

Criteria	Ratings				Pts
Completeness - Gets data from user Utilizes ReadInt to receive user input. Saves values in appropriately-named identifiers for validation.	1 pts Full Marks		0 pts No Marks		1 pts
Completeness - Validates User Data Validates that user-entered values are within the advertised limits.	4 pts Full Marks	2 pts Partial validation Validates only one end or neglects to check edge cases.		0 pts No Marks No validation	4 pts
Completeness - Displays Results (number of valid numbers, sum, and rounded average) display a special message if no negative numbers were entered (-2 if missed)	4 pts Full Marks	2 pts Partial Display		0 pts No Marks	4 pts
Completeness - Displays Closing Message 	1 pts Full Marks		0 pts No Marks		1 pts
Correctness - Number of Valid Numbers Correct number of valid numbers is displayed.	2 pts Full Marks	1 pts Incorrect for small numbers Correct number of valid numbers for values greater than 2, but fails for one or more of the following values (0, 1, 2)		0 pts No Marks	2 pts

Criteria	Ratings			Pts
Correctness - Calculation of rounded average is correct	3 pts Full Marks	2 pts Calculates average but not rounded	0 pts No Marks	3 pts
Correctness - Calculation of Sum is correct	3 pts Full Marks	0 pts No Marks		3 pts
Requirements - Solution is non-recursive	1 pts Full Marks	0 pts No Marks		1 pts
Lower limit is defined and used as a constant	1 pts Full Marks	0 pts No Marks		1 pts
Requirements - Well-Modularized Program is divided into logical sections, separated by Section Comment ks.	4 pts Full Marks	0 pts No Marks		4 pts
Coding Style - Appropriately named identifiers Identifiers named so that a person reading the code can intuit the purpose of a variable, constant, or label just by reading its name.	2 pts Full Marks	1 pts Partial Some identifiers are named well, with others having no relevance to their functionality.	0 pts No Marks	2 pts

Criteria	Ratings			Pts
Coding Style - Readability Program uses readable white-space, indentation, and spacing as per the Indentation Style Guide. Logical sections are separated by white space.	2 pts Full Marks	1 pts Marginally Readable Program is marginally readable but lacks proper alignment and white space.	0 pts No Marks	2 pts
(1pt) Extra Credit for rounded average Rounded average is rounded to the nearest .001.	0 pts Full Marks		0 pts No Marks	0 pts
Late Penalty Remove points here for late assignments. (Enter negative point value, 15% of 'earned' points per day late)	0 pts Full Marks		0 pts No Marks	0 pts
Total Points: 50				

