CS 271 Computer Architecture and Assembly Language
Programming Assignment #1

Objectives:

1. Introduction to *MASM* assembly language
2. Defining variables (integer and string)
3. Using library procedures for I/O
4. Integer arithmetic

Description:

Write and test a MASM program to perform the following tasks:

1. Display your name and program title on the output screen.
2. Display instructions for the user.
3. Prompt the user to enter three numbers (A, B, C) in descending order.
4. Calculate and display the sum and differences: (A+B, A-B, A+C, A-C, B+C, B-C, A+B+C).
5. Display a terminating message.

Requirements:

1. The main procedure must be divided into sections:
    - introduction
    - get the data
    - calculate the required values
    - display the results
    - say goodbye
2. The results of calculations **must** be stored in named variables before being displayed.
3. 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.
4. Submit your text code file (*.asm*) to Canvas by the due date.

Notes:

1. A program shell (template) is available on the course website.
2. You are not required to handle negative input or negative results.
3. Check the Syllabus for late submission guidelines.
4. To create, assemble, run, debug, and modify your program, follow the instructions on the course Syllabus page's Tools tab.
5. Find the assembly language instruction syntax and help on the Irvine lib functions in the textbook.

Example execution (user input is in ***italics***):

```
         Elementary Arithmetic     by Wile E. Coyote

Enter 3 numbers A > B > C, and I'll show you the sums and differences.

First number: 20
Second number: 10
Third number: 5

20 + 10 = 30
20 - 10 = 10
20 + 5 = 25
20 - 5 = 15
10 + 5 = 15
10 - 5 = 5
20 + 10 + 5 = 35

Impressed?  Bye!
```

Extra-credit options (original definition must be fulfilled):

1. Repeat until the user chooses to quit. (1pt)
2. Checks if numbers are in non-descending order. (1pt)
3. Handles negative results and computes B-A, C-A, C-B, C-B-A. (2pt)
4. Calculate and display the quotients A/B, A/C, B/C as floating-point numbers, rounded to the nearest .001. (2pt)

To ensure you receive credit for any extra credit options you did, 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—
```

For example, for extra credit option #2:

```
         Elementary Arithmetic     by Wile E. Coyote
**EC: Program verifies the numbers are in descending order.

Enter 3 numbers A > B > C, and I'll show you the sums and differences.

First number: 20
Second number: 25
ERROR: The numbers are not in descending order!

Impressed?  Bye!
```