Assignment #1

Topic(s):	I/O & Functions in C
Date assigned:	Wednesday, August 31, 2016
Date due:	Friday, September 9, 2016
Points:	25

For this first assignment, I would like to introduce you to the differences of I/O in C versus C++ while reinforcing the use of functions when writing code.

Specifically, using the C lecture code for Wednesday, you are to create a program called ***punetid*****ExamStats.c** (for me that's ***ryandj*****ExamStats.c**) that will allow a user the ability to enter between 0 and 50 (inclusive) integer exam scores. Your program must be fully documented and follow the C coding standards. You are to print each integer exam score right justified in a field of 4, five per line, followed by the:

a) arithmetic mean: the average of all exam scores

b) harmonic mean: $H = \frac{n}{\sum_{i=1}^{n}\frac{1}{x_i}}$

c) population standard deviation: $\sigma = \sqrt{\frac{\sum(x-\mu)^2}{N}}$

d) sample standard deviation: $s = \sqrt{\frac{\sum(x-\mu)^2}{N-1}}$

You can clear the screen using **system (“clear”).**

Here is exactly how I want your program to work:

1) Clear the screen

2) Enter exam scores exactly as shown below. If the user enters an invalid data value, simply repeat the line until the user enters a valid value. User input below is in bold.

```
Exam Stats

Enter Number of Exams: -1
Enter Number of Exams: 6

Exam #1: 100
Exam #2: 90
Exam #3: 800
Exam #3: 80
Exam #4: 70
Exam #5: 60
Exam #6: 50
```

3) Sample output for 5 scores and 0 scores respectively

```
Exam Summary Output

Exam Scores

  50  60  70  80  90
 100

Arithmetic Mean is 75.00
Harmonic Mean is 70.95
Population Standard Deviation is 17.08
Sample Standard Deviation is 18.71
Program Complete.
```

```
No Exam Data Exists
Program Complete.
```

Notes:

1. Your output is to look and work EXACTLY the same as the sample output above. Notice the exam scores are output from smallest to largest. Do not use a sorting routine to place the scores in order; instead, insert each score into its proper position within the array as it is entered by the user.
2. All scores must be positive (> 0) for the harmonic mean. If the scores entered by the user are not all positive, then output N/A for the harmonic mean.
3. The sample standard deviation shows N-1 as the divisor. If only one score is entered, then output N/A for the sample standard deviation.
4. We will use the coding standards for C Version 6.3 found on the CS300 home page.
5. Your code for this first assignment is to be written in C using Geany and tested on zeus. Programs written in other environments will not be graded. We will be using a submit script for submitting programs. I will talk about using the submit script in class, and we will submit a sample program in class.
6. Function documentation can be found in the coding standards document. Make sure you follow the documentation exactly. All functions including main are to be well documented following the coding standards. If you would like me to take a look at your code prior to submission, please stop by.
7. All code is to be original. You can use any of my functions from the C Topics lecture with proper siting. If you look stuff up on the Web or in the textbook, make the code original.
8. You must provide a hard copy (color, double-sided, stapled) as well as an electronic copy by 9:15am on the date in which the assignment is due.
9. Keep a record of the time you spent on this assignment. Place a comment in your file header documentation indicating how long you worked on this assignment. For example: **`//Hours worked = 5.5`**.

For testing on Zeus:

How to compile your project from the command line on zeus. Change into the directory that has the source code:

```
ryand@zeus:~> gcc –Wall –o ryandjExamStats ryandjExamStats.c -lm
```

Remember: Your program is to compile without any warnings!!! You will lose points if your program compiles with warnings.

How to run your project from the command line.

```
ryand@zeus:~> ./ryandjExamStats
```

How to submit your project

To produce a compressed tar file: Change into the directory that has the source code and type:

```
ryand@ralph:~> tar czf cs300_1_punetid.tar.gz punetidExamStats.c
```

To copy your compressed tar file to your home directory on zeus type:

```
ryand@ralph:~> scp cs300_1_punetid.tar.gz  ryand@zeus:
```

To extract your compressed tar file on zeus for testing:

```
ryand@zeus:~> tar xzf cs300_1_punetid.tar.gz
ryand@zeus:~> gcc –Wall –o punetidExamStats punetidExamStats.c
ryand@zeus:~> ./punetidExamStats
```

To use the submit script the command is:

```
submit classname filename
```

EXAMPLE

```
ryand@zeus:~> submit cs300f16 cs300_1_punetid.tar.gz
```

which will submit the zipped up tar file `cs300_1_punetid.tar.gz`

Once you have successfully submitted a file you will get a receipt which is a file that will end in **`.receipt`**. You can make sure your file was submitted correctly by typing the command:

```
checkReceipt submittedfile classname receiptfile
```

EXAMPLE

```
ryand@zeus:~> checkReceipt cs300_1_punetid.tar.gz cs300f16

     cs300_1_punetid.tar.gz.cs300f16.receipt
```

will produce the result:

```
HASH>>>k�˥���������`U�y�h
HASH>>>k�˥���������`U�y�h�
SUCCESS! Your receipt is valid
```

DO NOT MOVE OR MODIFY THE SUBMITTED FILE **OR** THE RECEIPT FILE OR THE CHECK RECEIPT COMMAND WILL NOT BE SUCCESSFUL.