Assignment #1

Topic(s):	I/O & Functions in C
Date assigned:	Friday, September 4, 2015
Date due:	Monday, September 14, 2015
Points:	20

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

A sample Eclipse project ExamStats exists in /home/CS300Public/2015. The project is in the form of a zip file. Download the file to your machine and extract it in your eclipse workspace. The project contains a source file called examstats.c that contains the code we discussed in class.

You are to modify the program so that it 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 mean, median, and mode. 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: 6

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

3cd) Output your results as follows:

```
Exam Summary Output

Exam Scores

  50  60  70  80  90
 100

Mean is 75.00
Median is 75.00
Mode is 50
```

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 function to order the scores after all scores have been entered. Instead, each score the user enters is to be placed in the proper positon within the array shifting previous array values when necessary.
2. We will use the coding standards for C Version 6.3 found on the CS300 home page.
3. The mode is not necessarily a unique value, so for the above example, any score would be the correct mode.
4. Your code is to be written in C using Eclipse 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.
5. 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.
6. 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.
7. 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.
8. 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. Change into the directory that has the Makefile and type:

```
shereen@zeus:~> make
```

Remember: Your program is to compile without any warnings!!!

How to run your project from the command line. Change into the directory that has the executeable (bin) and type:

```
shereen@zeus:~> ./examstats
```

How to submit your project

To produce a compressed tar file: Change into the directory that has the project ExamStats and type:

```
shereen@ralph:~> tar czf cs300_1_PUNetID.tar.gz ExamStats
```

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

```
shereen@ralph:~> scp cs300_1_PUNetID.tar.gz  shereen@zeus:
```

To extract your compressed tar file on zeus for testing:

```
shereen@zeus:~> tar xzf cs300_1_PUNetID.tar.gz
shereen@zeus:~> cd ExamStats
shereen@zeus:~> make
shereen@zeus:~> cd bin
shereen@zeus:~> ./examstats
```

To use the submit script the command is:

```
submit classname filename
```

EXAMPLE

```
shereen@zeus:~> submit cs300f15 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

```
shereen@zeus:~> checkReceipt cs300_1_PUNetID.tar.gz cs300f15

     cs300_1_PUNetID.tar.gz.cs300f15.receipt
```

will produce the result:

```
HASH>>>k�ℸ���������`U�y�h□
HASH>>>k�ℸ���������`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.