Name: _____

Problem	Points	Score
1	50	
2	50	
3 (BONUS)	50	
Total	100	

Notes:

(1) For this exam you are allowed to open a terminal window on your computer, you are allowed to web surf with Google, but you cannot use online chat or other interactive services.

(2) Your code and results should be placed in directories p01, p02 and p03.

The first step in this exam is to create a workspace in the following directory:

/data/courses/ece_1111/current/exams/exam_02/lastname_firstname

Set the permissions using "chmod -R u+rwx,g-rwx,o-rwx <lastname_firstname>" so only you have read and write permission to this directory. Create three subdirectories within this directory: p01, p02 and p03. You will use these for problems 1, 2 and 3 respectively. Put ALL your code in these directories. Do not touch your files after the exam is over.

You must use a make file, a header file and a main program file named p01.cc (or p02.cc). All other code needs to go into an implementation file called p01_00.cc (or p02_00.cc). Failure to follow these instructions will result in a grade of 0.

Problem No. 1: The standard deviation of a sequence of data points, or signal, is defined as:

$$\sigma = \sqrt{\frac{\sum_{n=0}^{N-1}(x[n] - \bar{x})^2}{N}}$$

where N is the total number of points and $\bar{x}$ is the average of the data points. The data points are held in the array $x[]$.

Using the program you previously developed that generates and writes a sinewave to a BINARY file as a SHORT INTEGER (16 bits or two bytes), generate a sinewave of amplitude 10,000 to a file. Use a frequency of 100 Hz and a duration of 10 seconds. You will have 1,000 points in this file. The signal will range from [-10,000,10,000]. The file will be 2,000 bytes long. Call this file **p01.dat**.

Write a program, p01.cc, that reads the data from this file and computes the standard deviation. To make things easy, you can read the data into memory first. Then you must design and implement a function that takes a signal as an argument and computes the standard deviation. Your interface requirements for the program are:

```
nedc_999[1]: p01.exe <filename>
file: %s
number of samples = %d
standard deviation = %15.4f
```

For example:

```
nedc_999[1]: p01.exe foo.dat
file: foo.dat
number of samples = 1000
standard deviation = 10000.0000
```

The function interface can be whatever you want as long as you write all of the code yourself, and you implement the above equation inside the function. Your code must be well commented. You must explain what you are doing in the comments in your code. It must be clear from these comments what algorithm you have used to implement this calculation.

Your program, **p01.cc**, must work on any binary file containing short integers. Use the file you generated as a test case. I will provide a previously unseen file for testing when I grade the exam. Its length and contents will be arbitrary. Demonstrate your program works by running it on your example file and putting the result in a comment at the top of your main program.

YOU CAN ONLY READ THE INPUT FILE ONCE – you are not allowed to do multiple passes over the input file.

Problem No. 2: In this section of the class, we learned how to compile and link programs using the math libraries. We also learned about functions and how to use the math library functions.

Using the random number generator function rand() (see "man -s3 rand), write a program that generates random numbers ranging from [-27,27] and writes them to a binary file as 16-bit integers (short integers). Your program's name should be **p02.cc**, and the file you create should be named **p02.dat**. You should create a file of 10,000 samples, which will be 20,000 bytes in size.

The interface to your program should be:

nedc_999[1]: p02.exe <filename> <number_of_samples>

Be careful with how you handle round off errors and arithmetic precision. You don't want to bias your random number generator so that its output is less random.

Problem No. 3 (BONUS): This can only be completed after you solve problem no. 1 and 2.

Run your code from problem 1 on the data generated in problem 2, and convince me that the result makes sense. Put your explanation in a file **p03.txt** in the subdirectory **p03**.