Name: _____

| Problem | Points | Score |
|---|---|---|
| 1 | 50 | |
| 2 | 50 | |
| Total | 100 | |

Notes:

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

/data/courses/ece_1111/current/exams/ex_01

Your directory should be your last name all lowercase, followed by an underscore, following by your first name (e.g, "picone_joseph"). Set the permissions using "chmod u+rwx,g-rwx,o-rwx <lastname>" so only you have read and write permission to this directory. Create subdirectories within this directory: p01, p02, … You will use these for problems 1 and 2, … respectively. Put ALL your code in these directories. Do not touch your files after the exam is over.

(2) For this exam you are allowed to open a terminal window on your computer, you are allowed to web surf with Google and use online chatbot services. However, if you turn in code that is moe advanced than what we have discussed in class, you will be penalized. Your code must only use things we have discussed. For example, we HAVE NOT discussed things like a static declaration, or C++ syntax. Make sure your code is built from concepts we have discussed at this point in the course.

(3) Do not submit code you do not understand. Use these tools as a learning tool, but not a substitute for learning how to write code. If I see particularly complicated code, I will ask you to come to my office and explain it and write a simpler version of it. These kinds of exercises don't usually go well. So, make sure you understand the code you are submitting.

**(50 pts) (p01) Problem No. 1**:

The resources directory contains files with filenames of the form:

```
/data/courses/ece_1111/resources/data/eeg/tusz/v2.0.5/edf/dev/aabbbccc/s001_2012/01_tcp_ar/aabbbccc_s001_t010.csv_b
i
```

Write a shell script that takes a path as the first argument (e.g., *p01.sh <path>*), a file extension as the second argument (e.g., *"csv_bi"*), a session ID (e.g, *"s001"* in the above pathname), and a match string as the last argument (e.g, *"version"*). Find all the files in the directory tree that end in the second argument (e.g., *"*.csv_bi"*), contain the session ID in their filename (only the filename, not the directory pathname), and contain the match string somewhere in file. Sum up the sizes of these files and print the total size in bytes to stdout.

Develop your code on a small set of files so you can easily test it. I will evaluate your code on the entire data set in */data/eeg*.

**(50 pts) (p02) Problem No. 2:** C Programming

Write a C program using our standard templates (Makefile, header file, driver file, etc.) named *p02.cc* (for the driver file) that implements the following task.

In this problem, we are going to manually implement a simple random number generator. Building on what you have done in the labs and homework, define an approximate value of $\pi$ as 3.14 in a double precision variable named "*joe*." Take the sin of this value using the math library sin function and store it in a double precision variable called *sum*. Normally this result would be zero, but due to the approximation we are using for $\pi$, it is not. Add a value of "1.0" to *joe* and repeat this process, adding the new sin value to the variable sum. Sum accumulates the result of all these operations. Repeat this process 1,000 times.

Scale the result so that it is a number in the range [-1000,1000]. Round this number to an integer and take the modulus of it with the number 27. The result is your random number.

Run your random number generator 10,000 times, and compute the product of each value with the previous value. Sum up these products. Mathematically, this means you are computing:

$$\sigma^2 = \sum_{n=0}^{N-2} x[n]x[n+1]$$

This is called a correlation coefficient. If it is close to zero, you have done a good job ☺

Display each random number generated to stdout, and then display the computed correlation coefficient:

```
p02.exe
1234.1235
27.999
-35.123
...
the correlation coefficient = 9.9999
```