Name: ____________________________________________

| Problem | Points | Score |
|---|---|---|
| 1 | 40 | |
| 2 | 40 | |
| 3 | 20 | |
| 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) Email the instructor your solution as a pdf file following the naming convention described in class. The code for this exam is "ex01". Use "ECE 3822: Exam No. 1" in the subject line. Points will be deducted if you get these things wrong.

(3) Provide your code in your pdf document in a textbox. I must be able to cut and paste your code into a terminal window (so do not include code as images). Provide a convincing demonstration that your code works.

**Problem No. 1**: Recall the format of the EEG text database:

/data/isip/data/software_tools/eeg_reports/v1.0/data/027/0027987/s02_2013_04_17/00006326_s02.txt

We have previously explained the structure of this data. The first three digits after /data/ are simply an index. The next portion of the pathname contains a 7-digit number that is the patient number. The next segment is the session information that includes the session number and the date (YYYY_MM_DD). The last segment we refer to as the EEG report filename and contains a text file with a filename containing a number followed by the session number. This text file contains an EEG report.

Find all EEG reports that meet these constraints:

- The patient number contains a 7 followed by any two numbers followed by a number between 7 and 9 (this means 7, 8 or 9);
- The EEG report filename contains the sequence "123".
- The session was from the year 2011.
- The session was from a month that has an odd number in the last position (e.g., January (01), March (03), ..., November (11).
- The session was on the 29th day of the month.
- The three-digit number after "/data/" (027 in the example above) was "757".

Show your command and the resulting filenames. Demonstrate that the names match your search criteria.

**Problem 2:** Write a shellscript that finds all the processes being run by root on the machine you are using and adds the process numbers. Your script should produce the following output:

```
myscript.sh:
  The machine name is nedc_000
  root owns 16 processes
  The sum of the process IDs for these is 9898999
```

Note that "nedc_000", "16" and "9898999" are just examples. Your script must compute and display the correct values. Nothing can be hardcoded.

Your script must run on any Linux machine or Mac without any problems, so stick to the bash language and simple standard Unix commands.

**Problem No. 3:** Your machine has $2^{16}$ bytes of physical memory. You need to declare an array of 100,000 floating point numbers in a C program. Can you do this? Explain how this might be possible by describing features of the operating system that might enable this.