Subject: ECE 1111: Exam No. 1
From: Joseph Picone <joseph.picone@gmail.com>
Date: 2/13/19, 10:57 AM
To: "ECE 1111: Facebook" <temple.engineering.ece1111@groups.facebook.com>, "ECE 1111: Google" <temple-engineering-ece1111@googlegroups.com>

```
To submit your solution to Exam No. 1, reply to this email, change the addressee to the class
instructor - joseph.picone@gmail.com - and attach three files:

 (1) p01.cc: Your solution to problem no. 1
 (2) p02.txt: Your solution to problem no. 2
 (3) p03.cc: Your solution to problem no. 3

Any deviations from this will result in a grade of 0.

=======================
(40 pts) Problem No. 1:

Consider the mathematical definition of a sum of sine functions:

 x(t) = A1 * sin(2*pi*f1*t) + A2 * sin(2*pi*f2*t)

where f1 = 10 Hz, A1 = 100, f2 = 20 Hz, A2 = 200. Write a program that prints out values of
this function at specific points in time. Your interface should be:

 p01.exe duration step_size

where t starts at zero and is incremented by (i * step_size) until t exceeds the duration.
For example,

 p01.exe 10.0 1.0

would print values of the above function for t = 0, 1, ..., 10.

Be careful to compute t using (i * step_size) rather than "t += step_size" to avoid roundoff
errors. Your loop should terminate when t exceeds the duration.

Be sure to define constants using #define and to read the duration and step_size from the
commandline using atof().

Print values to stdout using the following format:

 A1 = 100.0000
 f1 = 10.0000
 A2 = 200.0000
 f2 = 20.0000

 time = 0.0000  value = 0.0000
 time = 1.0000  value = 275.1234
 ...

Place your code in a single file, p01.cc, that can be compiled using this command:

 gcc -lm -o p01.exe p01.cc

=======================
```

(40 pts) Problem No. 2: Write a one-line Unix command that produces the 5 largest executable files in a directory tree. Your command should ideally work on any directory that is specified from the commandline (the root node) and recurse through all subdirectories below the root node. By "largest executable" I mean that the file has execute permission and its size in bytes is used as a measure of its size.

The 5 largest programs for /usr/bin on nedc_999 are:

```
-rwxr-xr-x   1 root root    68082016 Sep 18 13:49 dockerd
-rwxr-xr-x   1 root root    32794584 Sep 18 13:49 docker
-rwxr-xr-x   1 root root    26674752 Sep 18 13:49 docker-containerd
-rwxr-xr-x   1 root root    15509032 Aug  3  2018 amazon-ssm-agent
-rwxr-xr-x   1 root root    14953344 Sep 18 13:49 docker-ctr
```

Place your command in a text file, p02.txt.

=======================

(20 pts) Problem No. 3:

Write a C program that reads a stream of short integers from stdin and prints out the most significant byte of each value. For example, consider this file:

```
nedc_999_[1]: ls -l /data/courses/ece_1111/2019_spring/exams/exam_01/x0.dat
-rw-rw-r-- 1 picone ece_1111 4 Feb 13 09:31 /data/courses/ece_1111/2019_spring/exams/exam_01
/x0.dat

nedc_999_[1]: od -s x0.dat
0000000      1    128
0000004
```

x0.dat is 4 bytes long and contains two short integers.

Your C program should function this way:

```
cat x0.dat | p03.exe
```

and should print out the corresponding values of the first byte of each of these short integers as unsigned values. Your program should work for a binary file of any size.

A second test file is available:

```
nedc_999_[1]: ls -l x1.dat
-rw-rw-r-- 1 picone ece_1111 12 Feb 13 10:02 x1.dat
nedc_999_[1]: od -s x1.dat
0000000      1    128      1    128      1    128
0000014
```

Use the od command to verify that your program is producing the proper values.

Place your code in a single file, p03.cc, that can be compiled using this command:

```
 gcc -lm -o p03.exe p03.cc
```