Part 1: Setting Up Your Environment

For the first part of lab we are going to log into our accounts and write a 'Hello World' C program. This will ensure everyone has access to their accounts and all of the tools needed for the course.

# Part A: Writing the code

* + If you haven’t already, create a 'cs580' folder in your home folder. Inside CS580, create a ‘labs’ folder. Next create a 'lab1' folder inside the ‘labs’ folder.
  + As shown in class, create a file in the text editor named "lab1.c", and save it to the 'cs580/labs/lab1' folder you just created.
  + Open Terminal. Using the 'cd ~/cs580/labs/lab1' command (the '~' is a stand-in for your home directory), change the directory to your lab directory you just created
  + Create the lab file using the following command: ‘touch lab1.c’
  + You can edit the text file you created in 2 different ways:
    - Using command 'nano lab1.c'. (nano tutorial)
    - Open the file in the GUI text editor (gEdit).
  + Create a hello world program using the following line to print out ‘Hello World’
    - printf(“Hello World!\n”);
  + Save and exit.

# Part B: Compiling and Executing

* + Like you did in the previous step, navigate to your lab folder in the shell.
  + Type in 'ls' to list the directory's files. Ensure that 'lab1.c' is in the directory.
  + To compile, we will be using the program called 'make'. Do not confuse this with the 'makefile' which we will be creating. 'make' is the program that reads makefiles.
  + As described in class, create a makefile to compile your lab1.c source code to an executable called ‘lab1’.
  + The source should compile to an executable. You can run the executable with the command, './<executable>. In this case, './lab1'
  + Save your code.

Part 2 - Using Variables and Arithmetic

For the second part of the lab we will be using variables and math. Helpful link:

math.h <https://en.wikipedia.org/wiki/C_mathematical_functions>

# Part A

* + In your main, write code to evaluate the following expressions shown here:
    - 3x3 - 5x2 + 6 for x = 2.55.
    - (3.31 × 10-8 × 2.01 × 10-7) / (7.16 × 10-6 + 2.01 × 10-8)

# Part B

* + To round off an integer i to the next largest even multiple of another integer j, the following formula can be used:
  + int next\_multiple = i + j - i % j
    - For example, to round off 256 days to the next largest number of days evenly divisible by a week, values of i = 256 and j = 7 can be substituted into the preceding formula as follows:  
      *int next\_multiple = 256 + 7 - 256 % 7*
  + Write a program to find the next largest even multiple for the following values of i and j:
    - **i j**  
      365 7  
      12258 28  
      996 4
  + All of your code should go into the main function (do not use separate functions). Print your output for each exercise to the console in a nicely formatted way (as shown in class) to differentiate the answers. Make sure to save your answer from each exercise in a separate variable.

# Part C

* + print out each answer to the console again, this time typecast to an
    - int
    - double
    - char
  + Again, format the output nicely using format specifiers, and be aware that some results will be junk

Part 3 – Submission

Create a tar archive with the command “tar -czvf lab1.tar.gz .”, and then email your archive to bu580u2017@gmail.com and cc your TA dmu1@binghamton.edu before the submission deadline. Make sure you do not include the executable in your archive (make clean before creating the archive). Late assignments will not be accepted under any circumstances. Plan to turn in your assignments early.

Demo your lab before the demo deadline (after the submission deadline) by downloading your submission from class Gmail and extracting your archive with the command “tar -xvf lab1.tar.gz”. Then compile (with your makefile), and run your code, show your source to your TA, and answer any questions your TA may have.

Grading Guidelines

## Part 1:

* + Compiles and outputs hello world when run: 2 points

## Part 2:

* + Part A: 2 points
  + Part B: 3 points
  + Part C: 2 points

## Style Guidelines - 1 point

* + Uses whitespace in source to clearly identify code blocks
  + Clear variable names
  + No single letter names (except 'i')
  + Clearly formatted output

Note: don’t forget to include your honest statement in your code (see course syllabus). You can create a readme file for it and put it in your archive.

Submission Deadline: 11:59pm 9/5/2017 EDT

Demo Deadline: 2:00pm 9/8/2017 EDT