CS201 Spring 2019 (Karavanic)
Homework #2 Due: Tuesday, April 16 before 12pm local time

Part A: Problems from B&O:

2.56, 2.57, 2.59, 2.61, 2.74

OPTIONAL (Does not count in your grade for HW2): 2.65

Part B: Hands on Exercises

These exercises are designed to familiarize you with C, gcc, and gdb.

Problem 1: Debugging basics

1. Copy the contents of the lab directory into your own home directory. (note: check the man page for cp and see the -r option).
2. Compile buggy.c with the command line option for gdb, into an executable named buggy.
3. Run buggy.
4. Oops! There is an error. Start gdb
   chinstrap:~/public/cs201_lab2> gdb buggy
   This GDB was configured as "x86_64-linux-gnu".
   Reading symbols from /u/karavan/public/cs201_lab2/buggy...done.
5. Now run buggy under the control of the debugger
   (gdb) run
6. The run stops at the error. Skip to the error line in the source file (if you don't know how to do this quickly in your editor, ask the tutor for help).
7. Examine the runtime error of p by entering at the gdb prompt: print p
8. Fix the error in buggy.c
9. Re-compile buggy, still using the gdb option
10. Re-run buggy to verify that the error is fixed.
11. Restart gdb with buggy
12. Run gdb
13. Use print to look at the value of x. Does it work?
14. Set a breakpoint at the start of main. (use gdb commands "list" and "break") This will stop the running program when it is about to execute the instruction that matches the source code line number.
15. Now run. The program will stop at the breakpoint. Now print out the value of x.

SUBMIT: There is nothing to submit for Problem 1.

Problem 2: unsigned vs. signed integers

1. Read over the source code file numbers.c
2. Compile this file with the debugging option and run it.
3. Change the code so that the values for NOSIGN and YESSIGN print out instead of just their names.
4. Compile this file with the debugging option and run it.
5. Check the format strings are correct for each type.

SUBMIT: your modified version of numbers.c

Problem 3: Shifter

Write a program shifter.c that demonstrates left shift and right shift of both an int and an unsigned. Use printf to show the values before and after each shift. Use some interesting values to demonstrate the differences between signed and unsigned shifts.

You can hardcode the particular values to be shifted into your program OR you can read in the values from the keyboard.

Redirect the output of your program to a file called shifter_out.txt and run.

SUBMIT: shifter.c. and shifter_out.txt

Turning in Homework

You will turn in your assignment ONLY electronically, following the instructions below. Submission is due at the beginning of class on the day the assignment is due.

- **Electronic submission:**
- **Email address:** Electronic submissions should be sent to karavan at pdx.edu.
- **Subject line:** Use the subject line with three words in the following format (without the angle brackets): ***HW<n> <Your CS login name> <Your last name>*** where <n> is the assignment number (1, 2, 3, etc.). For example, Subject: HW2 wuchang Feng where wuchang is my CS login name, and Feng is my last name.
- **The tar file:** Your electronic submission will be packaged as a tar file with your CS login name. For example, suppose your CS login name is "jane". Your tar file will be called "jane.tar". Once the TA untars the file, a directory "jane" will be created and your submitted material will be extracted to the directory "john". To create the tar file for each assignment, first you need to create a directory with your name and put your submission in that directory. See instruction for making the tar file. **Note if the TA cannot recover the directory with your login name or cannot compile your program, you are not considered to have turned in your project.** You may gzip your file: gzip mylogin.tar will create a file called mylogin.tar.gz that is compressed.

• **Directory content organization:** Once you've created the directory with your name described above, please organize your files in the directory as follows:

Your answers for all non-programming problems should be in a single plain text file. Clearly label the problem number in your file.

Each programming problem will have its own directory. Name each directory such that it's easy to know which problem it is for (e.g., P2 for problem 2). Place source code files, header files (if any), Makefile, and typescript in the directory.

Do not include any binaries or unnecessary files with your submissions.

• **Email attachment:** Send only your tar file in the *attachment* of the email. Note that you may submit your assignment multiple times if ncessary before it's due, although this is not encouraged. In the case of multiple submissions, we'll just use your last submission for grading.