# CSCE 3600: Systems Programming

## Minor Assignment 1 – ASCII Bits & Bytes

## Due: Monday, January 29, 2018 at 11:59pm

**PROGRAM DESCRIPTION:**

In this introductory minor assignment, you will write a complete C program that will prompt for and accept two 4-digit hexadecimal numbers and then combine them (i.e., not add them) into a single 32-bit number and print out the results.

1. Prompt for and read in two 4-digit hexadecimal numbers. You may assume that the user enters a hexadecimal number, though it may be out of range. A valid hexadecimal range is considered to be 0000 to FFFF. If out of range, you will continually repeat this process until two valid 4-digit hexadecimal numbers are entered.

   Note that you may read in these two 4-digit hexadecimal numbers any way you wish (i.e., a hexadecimal number or a C-string that you convert to a hexadecimal number), but you must ultimately treat these as individual hexadecimal numbers.

2. You will print out the decimal representation for each of these hexadecimal numbers.
3. You will concatenate the two 4-digit hexadecimal numbers as follows into a single 8-digit hexadecimal number, where the first hexadecimal number will be in the 4 least significant digits while the second hexadecimal number will be in the 4 most significant digits.

   Note that this operation to concatenate must be done numerically and not simply concatenated as a string. Ideally, you would use bitwise operators to do this, but either way, you must do so while treating these as hexadecimal numbers.

4. You will print out the concatenated hexadecimal number as well as its decimal representation.

As a hint, you may want to perform bitwise operations to accomplish this, but also pay attention to the data types used for these numbers.

Although not required, you may find Chapter 2 on Bits, Bytes, and Data Types in the *System Programming with C and Unix* optional reference textbook by Adam Hoover to be helpful.

**REQUIREMENTS:**

- Your code should be well documented in terms of comments. For example, good comments in general consist of a header (with your name, course section, date, and brief description), comments for each variable, and commented blocks of code.
- Your program should be named “`minor1.c`”, without the quotes.

- Your program will be graded based largely on whether it works correctly on the CSE machines (e.g., `cse01`, `cse02`, …, `cse06`), so you should make sure that your program compiles and runs on a CSE machine.
- <u>This is an individual programming assignment that must be the sole work of the individual student. Any instance of academic dishonesty will result in a grade of "F" for the course, along with a report filed into the Academic Integrity Database.</u>

**SAMPLE OUTPUT** (user input shown in **bold green**):

```
$ ./a.out
Enter a 4-digit hexadecimal number (e.g., "af43"): 1FFFF
1ffff is out of range. Please try again...
Enter a 4-digit hexadecimal number (e.g., "af43"): -123
fffffedd is out of range. Please try again...
Enter a 4-digit hexadecimal number (e.g., "af43"): 1234
Your number in decimal is: 4660
Enter another 4-digit hexadecimal number (e.g., "af43"): abcd
Your number in decimal is: 43981
Merging 0x1234 and 0xabcd results in 0xabcd1234 or 2882343476 in decimal
$ ./a.out
Enter a 4-digit hexadecimal number (e.g., "af43"): af43
Your number in decimal is: 44867
Enter another 4-digit hexadecimal number (e.g., "af43"): 9ffe
Your number in decimal is: 40958
Merging 0xaf43 and 0x9ffe results in 0x9ffeaf43 or 2684268355 in decimal
```

**SUBMISSION:**

- You will electronically submit your program to the **Minor 1** dropbox in Canvas by the due date and time.