CS 210 - Fundamentals of Programming I
Spring 2019 - Practical Exam 1 Review Sheet

Reminders and announcements:

- For various reasons, Programming Practical Exam 1 has been moved to Tuesday, February 19
- This exam is worth 5% of your final grade in this course

Programming Practical Exam 1 will be on Tuesday, February 19, during the regularly scheduled class period. Half the morning class (with laptops) will be taking the exam in KC-255 so everyone has room to spread out notes.

For this exam, you will be expected to write programs in C using CodeBlocks to the specifications given in the exam. **The exam is open textbook (Hanley and Koffman), and open class notes and assignments (both your own and those on the course website, but not anyone else's notes or assignments). You are allowed to read the code from your own assignments, but you may NOT copy and paste from them.** I.e., All code submitted must the result of typing during the exam period. These are the only aids you may use in completing the exam. Make sure you know how to create a new console project and how to create an empty project and add an existing source file to it.

This exam will consist of one (1) problem. You may be given a file with some code already written that you will be asked to modify or enhance or you may be asked to write an entire program. The exam will be cumulative and comprehensive with respect to basic programming constructs in the sense that you are expected to be able to read and write code using concepts such as input/output, assignment and expressions, selection, repetition, and functions (both predefined in libraries and written by you). Material covered in lecture and assignments given through Thursday, February 7, will be tested. Switch statements, nested loops, and function as parameters will not be on the exam. The exam will be similar to the in-class exercises and programming projects.

The following is a list of topics that will be emphasized, but it is in no way to be construed as an exclusive list.

1. Declaration and use of constants and variables, use of arithmetic, relational, and logical operators, assignment statements.
2. Use of input and output statements (i.e., scanf and printf functions).
3. Use of selection constructs: if-statements, if-else statements, and multi-branch if-statements.
4. Use of repetition constructs: pre-test loops, post-test loops, counting loops and their implementations (while-loop, do-while-loop, for-loop, respectively).
5. Implementation of functions, including how to define and call them, (actual) arguments vs. (formal) parameters, received parameters, returned objects, and the use of the `void` type.