CpSc 101, 102, and 111
General Programming Assignment Formatting Requirements (for my sections)

1. For each program, you must work individually unless instructed otherwise. You may discuss the problem in a general sense with classmates, but at no time should you discuss your code in any form. You may not show another student your code, share your file with another student, look at another student's code, or tell another student what to type. ***Any evidence of cheating will result in a grade of zero for all students involved****. If you have questions, you should check with your lecture instructor or a lab instructor.*

2. Your program must adhere to the ***problem statement requirements*** **AND** ***coding standards*** below. Violations will lead to deductions.
   a. A **header comment** must be included at the top of ***each*** source file that you create. A header comment should include your name, username, the course number and section, the date, the program assignment number, and a brief description of the program along with any other comments or instructions about your program.

   b. **Functions**:
      - A brief description of each function (if the function's name is not self-explanatory), when applicable, should be placed in comments directly above the function header at the function definition, not at the prototype.
      - *In general*, no function body should be longer than 30 lines of code, excluding whitespace and comments. Sometimes is not possible to adhere to this rule, but if you have a block of code that performs some kind of task that could be separated out into a function - that's probably what you should do, especially if it's a block of code that will be, or might be, performed more than once. This will help keep each function in your program shorter and cleaner.

   c. **Statements**:
      - No more than one statement may be written on a single line.
      - The following may ***NOT*** be used: continue, goto, and break not in a switch structure. For example, ***do not use a break or a return statement to prematurely exit a control structure***. Control structures, if written properly, will provide the breaking capability on their own. The only place a break should be used is within a switch statement and nowhere else.
      - Lines of code should not extend beyond column 80 (i.e. no more than 80 characters per line, which includes the leading indentation spaces).
      - Diagnostic / debug prints should be disabled/deleted in the final submission.

   d. **Variables**:
      - Use reasonably descriptive names for variables and functions using naming standards discussed in class.
      - Global variables are **NOT** allowed; global constants are.
      - All variables in any given function should be declared at the top of the function, not scattered throughout.

   e. **Basic blocks (code enclosed in braces)**:
      - Blocks should use braces using methods demonstrated in class: the beginning brace either can be at the end of the opening line or on the next line.
      - Statements in the block should be indented consistent with logical nesting: 3 - 4 spaces (pick either 3 or 4).
      - With indentation, use **either** the spacebar **OR** the tab key, do **NOT** mix the two.

   f. **Comments**:
      - Comments should be used anywhere to help explain ***non-obvious*** details, algorithm, variables, or whatever would help explain to someone else who might not understand what you are doing. Do not write comments that re-state the code (e.g. `// an if statement`).
      - Do not include serious or repeated spelling or grammatical errors.
      - The comments should appear right above the line(s) and indented at the same level as the line(s) to which it belongs.
      - Keep in mind, however, that there is such a thing as too many comments.

   g. **Blank Lines**:
      - Declarations, functions, and code within basic blocks e.g. before and after control structures or logical lines of code that can be grouped together, should be separated by a blank line. *Do not double space everything*, but any time you can separate logical groups of code by a blank line to make the code more readable, do so.