Programming Documentation and Style Requirements

The following are required for each program that you submit in this class.

1. Header Comments

Place a header block at the top of every program. The header must have this format:

```
// CS 271
// Program Name: put program name here
// Author: put your name here
// Date: put today's date here
// Purpose: one or two sentences to describe what the program will do
```

2. Inline Comments

Place inline comments throughout the code to explain what is happening. All comments should be placed so that they do not interfere with readability of the code.

All comments must be single-line comments, beginning with //. Multiline comments (denoted by /* and */) are not acceptable for this course.

3. Formatting to Improve Readability

Leave blank lines between lines of code to improve readability (to separate parts of code).

Put a space on both sides of **all** binary operators, including both sides of an = . Examples:

```
x=y+2; Bad!

x = y + 2; Good!
```

4. Curly Brace Documentation

Comment **all** closing curly braces with a meaningful comment. Examples:

```
} // end if

} // end function main
```

5. **Choose Appropriate Identifiers**

Use meaningful variable names. Variable names should represent nouns from the problem statement. Function names should represent the purpose of the function.

6. Apply Consistent Indentation

Place the opening curly brace for a block at the end of the line of code that introduces the block. See examples below.

Place the closing curly brace at the beginning of a new line. Follow it with a comment (see #4 above).

Always indent the body (bodies) of a statement a uniform amount from the first character of the statement. Statements that have bodies include: loops, if-else statements, functions. Indent by the same amount for all statement bodies of at least 2 spaces and no more than 8 spaces (1 tab). Recommended indentation width is 3-4 spaces. Example:

```
// indentation of main - 0 spaces
int main (void) {
    // the first statement inside the main function is indented 4 spaces
    int addend1 = 3;
    int addend2 = 5;
    // indentation stays the same until a new control structure is reached
    // here we see an if statement (selection control structure)
    if (addend1 > addend2) {
        // indentation is increased by 4 spaces
        printf("%d is greater than %d\n", addend1, addend2);
    } // indentation of this closing curly brace matches the
      // indentation of the if
} // indentation of this curly brace matches the indentation of int
  // at the beginning of the main function header
```