C++ Coding Style Guidelines

1. File Headers
1.1 The Main Module
1.1.1 The main module file must begin with a block of comments that contains the following information:
1.1.1.1 the filename
1.1.1.2 the name of the programmer
1.1.1.3 the date
1.1.1.4 the name of the class (e.g., CSCI 123)
1.1.1.5 the time at which the class meets
1.1.1.6 the name of the instructor
1.1.1.7 the name of the project
1.1.1.8 a description of the program, followed by a description of the module's contents

1.2 Other Files
1.2.1 All other files should begin with a block containing:
1.2.1.1 the filename
1.2.1.2 the name of the programmer
1.2.1.3 the date
1.2.1.4 a description of the file's contents; it's not necessary to provide a general description of the entire program here

2. Function Headers
2.1 Every function must begin with a block of comments that contains the following information:
2.1.1 the name of the function
2.1.2 a description of what the function does; describe any assumptions you are making and any other helpful information
2.1.3 a description of the function's input; each formal parameter should be named and a description of purpose should follow
2.1.3.1 the mode of each parameter should be specified as being [IN], [OUT], or [IN/OUT]
2.1.4 a description of the function's return value and any overall side effects (e.g., modifications to global data or reference parameters)

3. Identifiers
3.1 All variables are to have meaningful names. One-letter variable names or terse abbreviations are unacceptable.
3.2 Variable identifiers are to begin with a lowercase letter and use mixed-case for multiple words.
3.3 All member variables should begin with a "m_" (e.g., m_szAddress, m_dwBitFlags, etc.).
3.4 All global variables should begin with a "g_" (e.g., g_totalCount, g_currentColor, etc.)
3.5 All class tag names should begin with an uppercase "C" immediately followed by the class name which also begins with an uppercase letter and is of mixed case. For example, "CDialogBox".
3.6 All function identifiers are to have meaningful names as well. They should be verb-oriented (i.e., begin with a verb followed by a noun) and describe what the function does.
3.7 Function identifiers begin with an uppercase letter (unlike variable names).

4. Whitespace
4.1 Vertical alignment should take place along a four-space tab.
4.2 An open curly brace for a block should appear on its own line, and indented one tab position from the line above it; the close curly brace should be justified with its corresponding open curly brace. The only exception is function bodies, whose curly braces are positioned at column zero.
4.3 Code contained within a block should be left justified with the open and close curly braces (which are already indented).
4.4 There should be no whitespace between a function identifier and its open parentheses.
4.5 There should be three blank lines between the closing brace of a function and the function header following it.
4.6 A blank space should be placed on either side of operators (e.g., **sum = val1 + val2**, not **sum=val1+val2**).

5. Functions
5.1 The vertical alignment for code contained within a function body should begin at the first tab position.
5.2 The closing curly brace of a function body should be followed by a comment indicating the name of the function.

6. Comments
6.1 Comments should appear on their own lines, immediately above and left justified with the code you are describing.
6.2 A section of comments should be preceded by a blank line.
6.3 Comments should describe the primary steps of your algorithm; they shouldn't be so particular so as to describe the details of each individual statement, nor should they be so vague so they generate more questions than they answer.

7. Miscellaneous
7.1 Don't let your code drift off beyond the 80th column.
7.2 Curly braces are to be used for all if/else statements, as well as for all loop statements, even if a statement does not require them.
7.3 Each variable is to be declared on its own line, don't provide a comma-separated list.
7.4 As mentioned above, there should be no whitespace between a function identifier and its open parentheses. An open parenthesis in any other context should be preceded by at least one blank space ("if" statements, "while" statements, "for" statements, etc.).