Course Book Updates

C/C++ Programming I & II

C and C++ are evolving languages and as new versions of the standards are released features are sometimes added, changed, or removed. Although most of the material provided in the "Beginning and Advanced C/C++ Notes" course book and accompanying audio lessons is not affected by these changes, there are a few differences worth noting, as explained below.

I recommend making appropriate notations in the book or simply attaching a copy of this sheet to the inside the cover as a reminder. Please contact me if you need clarification or find anything else in the course materials that needs updating.

`return` Statement in `main` (Note 1.3)
The `return` statement that was once required in the `main` function is now optional and a `0` is automatically returned if `main`'s closing brace is reached before a `return` statement is reached. However, using an explicit `return` statement in `main` is still recommended by many programmers.

Binary Integer Literals (C++ only) – Add to note 2.2A
In C++ only, integer literal values can now be written in binary if desired. Such literals must start with `0b` or `0B` and contain only `1`s and `0`s, optionally followed by a suffix that indicates their data type.

Hexadecimal Floating Point Literals – Add to note 2.4
Floating literal values can now be written in hexadecimal if desired. Such literals must start with `0x` or `0X` and be written in scientific notation with a mandatory exponent part that begins with `p` or `P` rather than with `e` or `E`. The optional suffix that forces the data type to `float` or `long double` is still permitted.

Integer and Modulo Division (Note 2.8)
Note 2.8 states that integer division with one or both operands negative can yield two possible values. While this was true in older compilers, modern compilers simply discard the fractional part. This behavior is illustrated in the "Implementation X" examples for both integer and modulo division and should always be assumed unless using a legacy compiler.

Placement of Variable Declarations (Note 3.5)
In C++ it has always been legal to intermix variable declarations with other code and to declare variables in the "initial expression" of a "for" statement. Modern C compilers also support this but older versions required declarations to be placed before other code and not placed in a "for" loop's initial expression. The best practice is to place declarations in C as you would in C++ unless using a legacy C compiler.

Function Overloading – Correction to note 5.8
`Compute(12.86, 23.8F);` on line 35 should instead be `Compute(12.86, 23.8);`

Declaration and Cast Syntax - Attributes (Note 6.1)
In the discussion of the Right-Left rule in note 6.1 and elsewhere, the symbols *, &, [], and () are sometimes referred to as "attributes" when used in declarations or casts. However, this is merely a term I have used for convenience because there is no "official" name for them other than simply "punctuators". Officially, the term "attribute" refers to an unrelated concept that is not covered in these courses.

The "null pointer"
In both C and C++ a literal `0` or the `NULL` macro can be used to represent the null pointer. However, in C++ the keyword `nullptr` should be used instead.