COMSC-200 Lab 4
Applying const

GOOD PROGRAMMING PRACTICES *show / hide*

ABOUT THIS ASSIGNMENT
In this lab session, you will further modify the **GeometryHomework** solution from labs 1-3 so that it uses the `const` keyword C++ classes. You will also practice programming with classes by doing exercises from our textbook.

As you complete this assignment, post the required files to the COMSC server. You may post them all at once, or one-at-a-time as you complete them, as you prefer. The individual labs are graded in order, starting with "a", which must be fully correct before "b" is graded, and so on. This assignment is worth 50-points, to be awarded after all labs are successfully completed. Use the "Submit your SP2015 work" link on the class website to post your file for this lab, using the **lab4** folder provided.

Lab 4a [`Time.h` and `Time.cpp`]
Write the class as presented in figures 10.1 and 10.2 of our Deitel textbook, with modifications per the universal requirements listed in lab 1a, and the grading policy. Name the files **Time.h** and **Time.cpp**. Name the functions exactly as Dietel does.

Properly apply the principle of least privilege. In your class files, flag all getters as getters, using the trailing `const`. That means fixing the "print standard time" getter written in the textbook, so that it really is a getter.

Make sure that your fully test your Time class by writing your own test CPP as you did in lab 3a. Do *not* submit your test CPP(s) -- the instructor has his own!

*Post **Time.h** and **Time.cpp** to the COMSC server for credit.*

Lab 4b Using the `const` Keyword [`GeometryHomework4.cpp`]
In this small change to the GeometryHomework program from lab 3, associate the `const` keyword with all data member declarations (that is, make the objects "immutable"), *non-constructor* member functions (that is, make them all "getters"), local variables (including arrays), and function parameter variables (that is, make them `const` references, and constant read-only pointers) as is possible. Basically, if something can be made `const`, make it `const`. That means *at least* the following:

- Your constructor parameters should be constant arrays of constant read-only pointers.
- Your class data members should all be immutable.
- None of your class member functions should be setters.
- The void* array should be a constant array of variable read-only pointers.
- The token array should be a constant array of variable read-only pointers.

It is NOT necessary to use `const` in expressions like `new Square(token)` (that means to *not* write "new const Square(...)"), or function return types of `void` or `int` -- in fact, just don't do it. Such uses of `const` are meaningless, and therefore misleading. If you have any uncertainty about where to use `const` and where not to use `const`, use the online discussion group to explore.

Note that when the data members are declared with `const`, the statements in the constructors that initialize them will no longer compile. Fix this by using initializer lists. Since there is logic involved in setting these values (using default values of zero for missing dimensions, instead of using `atof`), you will need to use *conditional* expressions or calls to *private setter functions* in the initializer list.

Do *not* use more than one source file -- that means no **.h** files, and just a single **.cpp**. *Post* ***GeometryHomework4.cpp*** *to the COMSC server for credit.* Save this file, because modifications will be made to this program in labs 11-13. The GeometryHomework series is the subject of seven (7) lab assignments.

How to pause a console program: *show / hide*

GRADING POLICY *show/hide*
