COMSC-200 Lab 7
The `string` Class

GOOD PROGRAMMING PRACTICES *show / hide*

ABOUT THIS ASSIGNMENT
In this lab session, you begin solving the **ShortestRoute** problem -- a problem that will be completed in the final project. 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 **lab7** folder provided.

Lab 7a [`MyStrings.cpp`]
Write the **CPP** program as presented in figure 18.6 of our Deitel textbook, with modifications per the universal requirements listed in lab 1a, and the grading policy. Name the file **MyStrings.cpp**. But embellish it with one or more of your own tests, modeled after the ones in the code listing, added at the end of main.

You may customize the program as you wish, using various string features that you may find interesting, but do NOT prompt the user for input. Also, note that the output sample in the textbook is *not* correct! Figure out what the output *should* look like, and go by that in your testing. You may use the online discussion group to share what you think the output should be.

*Post **MyStrings.cpp** to the COMSC server for credit.*

Lab 7b Using the `const` Keyword and `strings` [`ShortestRoute3.cpp`]
In small change to lab 6's shortest route classes, you will apply the `const` keyword and use the `string` class. Copy **ShortestRoute2.cpp** from your **lab6** folder into your **lab7** folder, and rename it as **ShortestRoute3.cpp**. If it was not fully correct in lab 6, make the required corrections. Make sure that the `Leg` and `Route` classes are fully correct before proceeding. Here are the changes:

1. ROUTE DISTANCE: Add the capability to track the total distance for a route. Use a private member variable that is set in the constructors and accessed via friend relationships. It will be needed in a function of the yet-to-be-written **ShortestRoute** class, to compare distances of all possible routes for getting from city "A" to city "Z".

You should also change the `outputRoute` function, so that it uses the total distance as determined above, instead of any value computed inside the function.

2. COMPARE: Write a boolean compare function, that tests the host object against another object, and returns `true` if the host object's total route distance is greater than that of the other object. Here is the prototype: `bool isGreaterThan(const Route&) const;`.

Also, modify the `main` function to test the new compare function, by comparing at least 3 pairs of routes, and outputting some result indicating which route is longer and which is shorter.

3. CONST: Associate the `const` keyword with all member data variables in the `Leg` and `Route` classes. Declare the `Leg**` data member as a *constant pointer* to an array of variable read-only pointers.

Once the data members are declared with `const`, the statements in the constructors that initialize them may no longer compile -- you'll need to find a way around this...

4. STRINGS: Change the `char*` and `char` array variables to `strings` wherever they appear. (This is not really an improvement in our application, but it is an opportunity to practice using the `string` class. Also, it frees the `Leg` class from depending on the cities persisting throughout the program -- it stores copies instead of pointers.)

5. EXCEPTIONS: Modify the `Route(const Route&, const Leg&)` constructor to throw a simple exception if the start city of the `Leg` object does not match the end city of the `Route`. Throw either a `const char*` or a `string`. Test it by adding like this to `main`:

```
try
{
  Route(Route(Leg("a", "b", 0)), Leg("c", "d", 0));
}
catch (const char* ex)
{
  cout << "ERROR DETECTED: " << ex << endl;
}
```

Guard against memory leaks!

Do *not* use more than one source file -- that means no **.h** files, and just a single **.cpp**. *Post **ShortestRoute3.cpp** to the COMSC server for credit.* Save this file, because modifications will be made to this program in the final project. The ShortestRoute series is the subject of three (3) lab assignments, and the final project.

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

GRADING POLICY *show/hide*