CS1410 PROGRAMMING ASSIGNMENT 7

This is a group assignment. Go to Canvas and find out is part of your team for this assignment.

This assignment will help you understand C-Strings. and well as the use of smart pointers. This is also our first use of dynamically allocated memory through the use of smart pointers. Begin the assignment by downloading the sample code for your class. Once again, we will use a Test-Driven-Development (TDD) technique to develop our code.

The requirements for your class are defined in the attached **refman.pdf** file. You need to get use to read technical documentation and develop your software based on it.

Note: this project needs to be created with **C++14 standard**.

PART 1 (30 POINTS)

Build your class **constructor** and **destructor** according to the **refman.pdf** specifications. Also you need to develop a **getInternalCString() Method** for the string.

This part should pass **Test 1** and **Test 2** from **testMyString.cpp**

PART 2 (25 POINTS)

Develop a **compareStr() Method** for your class according to the **refman.pdf** specifications.

This part should pass **Test 3** from **testMyString.cpp**

PART 3 (25 POINTS)

Develop a **reverseit() Method** for your class according to the **refman.pdf** specifications.

This part should pass **Test 4** from **testMyString.cpp**

PART 4 (10 POINTS)

Develop a **<< operator overload** for your class according to the **refman.pdf** specifications.

This part should pass **Test 5** from **testMyString.cpp**

PART 5 (10 POINTS)

Documentation. You are required to create your own **Doxygen** documentation file.

To generate a Doxygen file configuration run the following command on the terminal inside your project:

$ doxygen –g

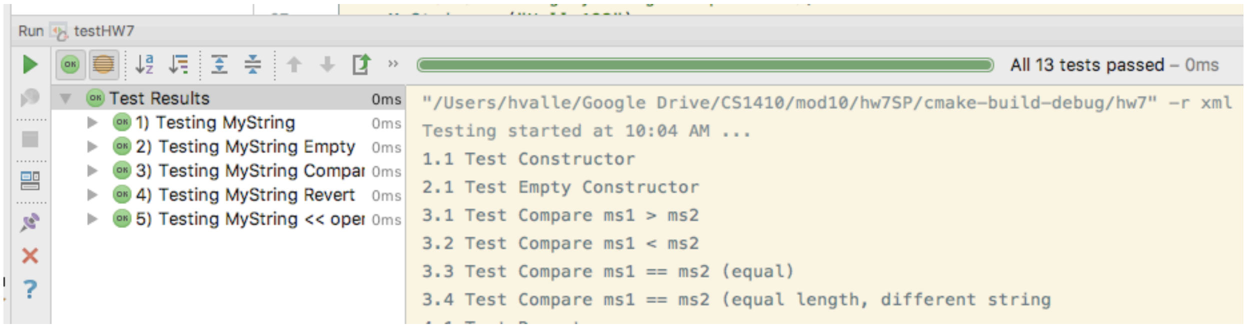
The above command generates your Doxyfile default configuration file. Now, generate your documentation:

$ doxygen Doxyfile

This will produce an html and latex folder with your documentation.

Finally, you need to provide a **test results html** file (as discussed in class) for your test scenarios. Remember the file is generated by CLion (see attachment sample)

SAMPLE OUTPUT:



```
Test Results                      0ms
  1) Testing MyString             0ms
  2) Testing MyString Empty       0ms
  3) Testing MyString Compar      0ms
  4) Testing MyString Revert      0ms
  5) Testing MyString << oper     0ms
```

```
"/Users/hvalle/Google Drive/CS1410/mod10/hw7SP/cmake-build-debug/hw7" -r xml
Testing started at 10:04 AM ...
1.1 Test Constructor
2.1 Test Empty Constructor
3.1 Test Compare ms1 > ms2
3.2 Test Compare ms1 < ms2
3.3 Test Compare ms1 == ms2 (equal)
3.4 Test Compare ms1 == ms2 (equal length, different string
```

When you are done, zip your project folder and upload it to Canvas.
Make sure you add your project to github.com and you send me an invitation via my user name (hugo-valle).

Use the .zip file given to you (see link in Canvas) as reference.

NAMING CONVENTION:

Your scripts will be submitted as a zip file through canvas and they must follow this naming convention:

TeamName_Hw#.zip

For example if your name is John Perez, and you need to submit hw#1, your zip fle name will be: ***protons_hw1.zip***

Note: Make sure your zip file includes your **three source files** and your **Makefile** as well.

HOW TO SUBMIT:

At the end of this assignment, upload your zip file to Canvas.

SUBMISSION CHECKLIST

1. Does it follow the naming convention?
2. Did you submit the text entry in canvas? This is your zip file name.

---------------- PROGRAM ESSENTIAL HELP ------------------

Make sure you understand the following for smart pointers:

http://en.cppreference.com/w/cpp/memory/unique_ptr