**Data Structures (EECS 2080C) -- Lab 04**

***Topics covered: Classes***

**Objective:**

The objective of this Lab is to recursion/stacks and Exceptions

Note Skeleton Lab code doesn’t compile, as this was intentional. It is expected that you will provide the missing details to get it compiling. Place all code in the Recursion.cpp\h

**Task 1:** Implement computing if a std::string is a palidrome different ways.

1. Recursion
2. Stack (you may use std::stack or std::list (and use the stack-like methods) – google it for example if need be)

Provide Screen shot of a positive and negative case for palindrome test.

Provide Unit tests for each implementation. Your unit tests should deal with at least, empty string, simple string, complicated string, and at least one negative tests (min 4 tests per methodology, for a total of 8 tests).

I have provided a sample implementation for calculating factorial for comparison. Feel free to modify this code to calculate the palindromes.

**Task 2:** Recursion\Stacks

Using the recursive implementation for Queensolver. Switch it to use a stack.

You will need to provide screen shots of just one valid solution of a queen placed on a 8X8 board. **You MUST use a stack.**

You can stop after getting a single valid solution.

**Task 3:** Exceptions

* Implement CallSimpleException to return a string of the status of the call to make the unit tests pass.
* Additionally using try\catch blocks catch the exceptions to report the error string correct
* Free the memory of the MyFakeClass appropriately.
* Add in MyException3

**Lab Submission:**

1. Write a lab report including the following information ( in addition to what is listed on Piazza):
   1. Screenshots requested from Task 1, 2.
   2. A short essay about Exception handling. Specifically, what advantages do exceptions have over traditional error codes.
2. A tar of your total solution. Please do a Clean build before submitting (this will make the size be smaller). Include all source code from all tasks, input and output files (if any) from Task 1, Task2, and Task3.

**Lab Grading:**

1. 20% - Lab attendance
2. 20% - Task 1 has been correctly implemented and meets all requirements.
3. 30% - Task 2 has been correctly implemented and meets all requirements.
4. 10% - Task 3 has been correctly implemented and meets all requirements.
5. 20% - Lab report contains all required information and is well written.

Attach RecursionList.h, RecursionList.cpp, and your UnitTest file that contains your new test.