**Lab 3 - Preprocessor, functions, and arrays**

*Due Date: 5:00 p.m., October 9, 2015*

Part 1 -- Setting Up Your #defines and #macros

# Part A

* + Set up the following #defines
    - TRUE
      * should evaluate to a boolean true
    - FALSE
      * should evaluate to a boolean false
    - EQUAL
      * should be able to use in place of ==
  + Set up a macro called ‘TRACE’ that prints the current line number using the global [\_\_LINE\_\_](http://www.cprogramming.com/reference/preprocessor/__LINE__.html) . Use [fprintf](http://www.cplusplus.com/reference/cstdio/fprintf/)(stderr, “%d”, [\_\_LINE\_\_](http://www.cprogramming.com/reference/preprocessor/__LINE__.html)) with instead of printf().
  + Setup a macro called ‘DEBUG’ that allows you to print a string and the current line number. Once again, use fprintf with stderr, instead of printf.
  + Wrap your TRACE and DEBUG macros in a preprocessor #if/else statement that you can turn on and off with a 1 or 0. The else statements should have empty versions of the TRACE and DEBUG macros.
  + Anytime you test for equality in the lab, use the EQUAL define

Part 2 -- String Manipulations

# Part A

* + For this part of the lab you will implement your own version of the library function strstr(), with the following function interface:   
    *int myStrStr (char haystack[], char needle[], char buffer[]);*
    - Your function will take 3 strings, a 'haystack' string, a 'needle' string, and a buffer string. You will search the haystack string for a sequence matching the needle string, and copy the found result from the haystack string into the buffer (do not copy the needle string). You should return a 1 if the matching sequence in the haystack is found and a 0 if the needle is not found.
  + Test your function with the following strings.
    - haystack="apple", needle="app"
    - haystack="orange", needle="ge"
    - haystack="blueberry", needle="ueber"
    - haystack="strawberry", needle="strawberry"
    - haystack="grapefruit", needle="terrible"
  + From within your main, after your function call, neatly print the haystack, needle, buffer from the myStrStr function.

Part 3 - Submission

* Create a tar archive with the command ”tar -czvf lab3.tar.gz .”, and then upload the archive to Blackboard before the deadline. Make sure you do not include the executable in your archive (make clean before creating the archive).
* Demo your lab at the beginning of lab by downloading from blackboard and extracting your archive with the command "tar -xvf lab3.tar.gz". Then compile (with your makefile), and run your code, show your source to the TA, and answer any questions she or he may have.

Grading Guidelines

## Part 1: 3 points

* + Implements all required defines and macros
  + Uses fprintf with stderr

## Part 2: 6 points

* + myStrStr function works with all test cases

## Style Guidelines - 1 point

* + Clear, readable code submitted to blackboard as tar and gzipped archive