# CS4760 Operating Systems - Project No. 1

**Due Date Feb. 19, 2017 Max. Points : 100**

**Deliver your project using Mygateway to 23.59 on the due date.**

Important: Please do all assignments on hoare

## Unix System Calls and Library Functions

The goal of this homework is to become familiar with the environment in hoare while practicing system calls. I'll like to see the use of perror and getopt in this submission. This is Exercise 2.13 (p. 55-56) in your text by Robbins/Robbins. I encourage you to use code from the textbook when you think it applies.

**Problem:** Implement the logging utility whose template is provided in program 2.13 (p. 56) in the text. Use the specifications given in the text. You will then demonstrate this logging utility by using it as described in the next few paragraphs.

Create a .cpp file and .h file containing all the code for the logging utility. In your main.cpp, have code that then uses that logging utility. All functions in the logging utility should be used at least twice, with a special message dependent on a command line argument discussed later. Try and come up with creative logging messages that would indicate some error that needed to be Logged (for amusement value). Note that it might be a good idea to write extra functions from the minimum in Exercise 2.13 to make things easier later on. So, if you find yourself repeating some common task to build an error message, write a function for it.

I do want a standard format for logged messages. The format for logged messages should be:

buggy_code: time: Error: Detailed error message

where buggy_code is actually the name of the executable (argv[0]) that you are trying to debug. It should be appropriately modified if the name of executable is changed without a need to recompile the source. time should be nanoseconds since epoch.

An example of a log file in practice might be something like:

*****Begin Log*****

./debugger: 1485211525309381617: Error: nValue = 37 - No dogs detected
./debugger: 1485211525309394302: Error: nValue = 37 - Division by zero
./debugger: 1485211525309400907: Error: nValue = 37 - P == NP

and so on...

Your main executable should use command line arguments. You must implement at least 3 command line arguments using getopt:

-h
-n x
-l filename

If given the command line arguments -h, a useful message describing the function of all the command line arguments should be displayed. The -n x is a command line parameter that should set a variable in your program to the integer x. The value of this variable should be logged every time your program is run using the logging utility above. The default value of this variable should be 37. That is, if the command line argument -n is not used, it should log a value of 37. The -l filename command will set the name of the log file. The default value for the name of the logfile should be logfile.txt.

**What to handin**

Handin an electronic copy of all the sources, README, Makefile(s), and results using Mygateway. Do not forget Makefile (with suffix or pattern rules), and README for the assignment. It will be nice when you will use some system for version control. This README should tell me how to run your project, and any problems or issues that it has running. Omission of a Makefile will result in a loss of 10 points, while README will cost you 5 points. Make sure that there is no shared memory left after your process finishes (normal or interrupted termination). Do not forgot include in the README file your name and the date when your project was submitted. I do not like to see any extensions on Makefile and README files.

I cannot stress enough how important it is that you become comfortable with the things in this project. You will need to be very comfortable with C programming, Makefile and command line arguments before we move on to more complicated projects. If you have any issues or confusion on what I want you to accomplish in this project, please come by my office or send me an email. Do not wait until the last minute.