ECS 122B Environment Setup Project 1

1. Overview
    1.1. Development cycle
        1.1.1. Add a test
        1.1.2. Run all tests
        1.1.3. Write the code
        1.1.4. Run tests
        1.1.5. Modify code
    1.2. Unit test
2. Review
    2.1. Environment variables
        2.1.1. Set up for current shell: export LANG=UTF_8
        2.1.2. Configure to effect (source)
        2.1.3. $PATH and $LD_LIBRARY_PATH
    2.2. Symbolic links (interpreted as a path to another file or directory)
        2.2.1. `In -s target_path link_path`
    2.3. Linking libraries
        2.3.1. use the gcc command line options -L for the path to the library files and -l to link in a library (a .so or a .a): -L{path to file containing library} -l${library name}
        2.3.2. You may also need to specify and include path so the compiler can find the library header file: -I /home/newhall/include
        2.3.3. export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/ubuntu/workspace/lib
    2.4. Static library archives (command ar)
        2.4.1. create and update static library files that the link editor or linker uses and for generating
        2.4.2. **Example:** `ar -rv libclass.a class1.o class2.o class3.o`
        2.4.3. `cc main.c libclass.a` **same as** `cc main.c class1.o class2.o class3.o`
    2.5. Cmake (how to use)
        2.5.1. CMakelist.txt
        2.5.2. http://derekmolloy.ie/hello-world-introductions-to-cmake/
3. Setting up developing environment
    3.1. About Cloud9 (you can skip this if you are comfortable in your own linux environment or wish to use something like VirtualBox. Cloud9 is a simple way to get started but will sometimes hang)
        3.1.1. Create Cloud9 account https://c9.io
        3.1.2. Create new workspace
        3.1.3. Click "Private: This is a workspace for your eyes only"
        3.1.4. Choose C++ Template
        3.1.5. Click Create Workspace
    3.2. Update gcc
        3.2.1. Add repository: sudo add-apt-repository ppa:ubuntu-toolchain-r/test

- 3.2.2. sudo apt install gcc-6
- 3.2.3. sudo apt install g++-6
- 3.2.4. Make gcc-6/g++-6 your default gcc/g++ compiler by executing "sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-6 60 --slave /usr/bin/g++ g++ /usr/bin/g++-6" (ln -s /usr/bin/gcc-6 /usr/bin/gcc)
- 3.2.5. g++ -v to check or (gcc --version)

3.3. Textbook code

- 3.3.1. Create /home/ubuntu/workspace/langr-book on your Cloud9 IDE
- 3.3.2. In the above directory, download the source with the terminal command "curl -o lotdd-code.tgz http://media.pragprog.com/titles/lotdd/code/lotdd-code.tgz" (or wget)
- 3.3.3. Unpack the tgz file using gunzip and tar. The source should be in /home/ubuntu/workspace/langr-book/code (tar -xf lotdd-code.tgz)

3.4. Google test

- 3.4.1. In /home/ubuntu/workspace/testing-frameworks, use git to clone the repository at https://github.com/google/googletest
- 3.4.2. Create /home/ubuntu/workspace/testing-frameworks/googletest/mybuild
- 3.4.3. Use "cmake .." in /home/ubuntu/workspace/testing-frameworks/googletest/mybuild
  - . means current folder
  - .. upper level folder
- 3.4.4. Run make in /home/ubuntu/workspace/testing-frameworks/googletest/mybuild
- 3.4.5. Create or modify the file ~/.bash_profile, add the following line, then refresh environment variables (source ~/.bash_profile): export GTEST_DIR="/home/ubuntu/workspace/testing-frameworks/googletest/googletest", then refresh environment variables.
- 3.4.6. Verify Google Test install with: "cd ${GTEST_DIR}/make", "make", then "./sample1_unittest"
- 3.4.7. Add the line "export PATH=$PATH:"$GTEST_DIR/include" to your ~/.bash_profile

3.5. Google mock

- 3.5.1. Add the following line to ~/.bash_profile: export GMOCK_DIR="/home/ubuntu/workspace/testing-frameworks/googletest/googlemock"
- 3.5.2. Add the following line to ~/.bash_profile, then refresh environment variables (source ~/.bash_profile): export GMOCK_HOME=$GMOCK_DIR
- 3.5.3. Verify Google Mock install with: "cd ${GMOCK_DIR}/make", "make", then "./gmock_test"
- 3.5.4. In $GMOCK_DIR, create a symbolic link to googletest with the command "ln -s /home/ubuntu/workspace/testing-frameworks/googletest/googletest/ gtest"

        3.5.5. In $GMOCK_DIR, create a symbolic link with "ln -s make mybuild"
        3.5.6. In $GMOCK_DIR/mybuild execute the command "ar -rv libgmock.a gtest-all.o gmock-all.o"
        3.5.7. Create $GMOCK_DIR/gtest/mybuild. Do “cmake ..”, Then “make”
        3.5.8. Verify install: create "/home/ubuntu/workspace/langr-book/code/c2/40/build", change to that directory, then execute "cmake .." followed by "make". Run "test" and 26 tests should pass.
        3.5.9. Add the line "export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:"$GMOCK_DIR/mybuild" to your ~/.bash_profile

3.6. Cpputest

- 3.6.1. move to directory "/home/ubuntu/workspace/testing-frameworks"
- 3.6.2. Use git to clone the cpputest repository at git://github.com/cpputest/cpputest.git
- 3.6.3. Add the line "export CPPUTEST_HOME=”/home/ubuntu/workspace/testing-frameworks/cpputest" to your ~/.bash_profile. Refresh your environment variable
- 3.6.4. Change to directory “cd $GMOCK_HOME”, do “cmake .”, then “make”
- 3.6.5. Build from source (cd $CPPUTEST_HOME ./autogen.sh ./configure make)   (noted: cmake method is not working for me)
- 3.6.6. Verify CppUtest install. See README.md
    - 3.6.6.1.
```
#include "CppUTest/CommandLineTestRunner.h"
int main(int ac, char** av)
{
  return RUN_ALL_TESTS(ac, av);
}
```
    - 3.6.6.2.
```
g++ main.cpp -I$CPPUTEST_HOME/include
-L$CPPUTEST_HOME/lib -lCppUTest -lCppUTestExt
```
    - 3.6.6.3. Use “./a.out” to verify.

3.7. LibCURL

- 3.7.1. Download and unpack https://curl.haxx.se/download/curl-7.53.1.tar.gz to /home/ubuntu/workspace/lib
- 3.7.2. Create a CURL_HOME environment variable in ~/.bash_profile
- 3.7.3. Cd $CURL_HOME mkdir build cd build cmake .. make

3.8. Boost

- 3.8.1. Download the boost: wget https://superb-dca2.dl.sourceforge.net/project/boost/boost/1.63.0/boost_1_63_0.tar.gz   (Don’t use github version)
- 3.8.2. Build from source
- 3.8.3. Set up BOOST_ROOT like before . cd $BOOST_ROOT
- 3.8.4. ./bootstrap.sh --with-libraries=filesystem,system

3.8.5. ./b2

4. Verify Setup
    4.1. /home/ubuntu/workspace/langr-book/code/c2/40
    4.2. /home/ubuntu/workspace/langr-book/code/c3/18
    4.3. /home/ubuntu/workspace/langr-book/code/c6/19 (modified the code to pass:
    add #include <numeric>
    Replace accumulate with std::accumulate)

```
Mkdir build
Cd build
Cmake ..
Make
./test
```