

CMSC 341 Homework 1 – All Versions

Programming C++ at UMBC.

Name:
Section:
HW #:
Version:
Username:

This will be a review for some and new for others. For veterans, this should take less than 30 minutes, for rookies, this will be a while, please read all of it first and then complete. Plan appropriately. There will be many CMSC 341 students who have transferred from another college that were taught Java instead of C++. This HW will make sure everyone is successful. Make sure to complete each problem below **IN ORDER**. If you get stuck, visit a TA or instructor since emailing will not remedy any issues. You will be submitting code and screen captures to prove you correctly did the problems below. Please be careful on how the screen capture files are to be named.

#1 (GL) Linux Basics – 10 pts

Simply put, create directories within your home directory of:

CMSC341/HW1/src (src is a sub-directory of HW1)

If you are unfamiliar with UMBC Linux Server called GL, watch the first 4 videos in this [playlist](#).

To prove you did this correctly, while in the CMSC341/HW1/src directory, type “pwd” (present working directory). Your output should look like below, but your username instead:

```
[slupoli@linux1 src]$ pwd
/afs/umbc.edu/users/s/l/slupoli/home/CMSC341/HW1/src
[slupoli@linux1 src]$
```

Take a screen capture (help with screen capture below) of your success and name it **directory.jpg**. (This image should ONLY show the pwd result) If you know how to transfer files to another server like GL using a file transfer program, go ahead. If not, help is below.

[How to screen capture with Windows](#)

[Using a file transfer program](#)

#2 Copying given code – 10 pts

This is the ONLY time you are allowed to copy code from anyone! Period. There will be some projects where the instructors may provide code. Sometimes there will be a URL link, sometimes it will be on GL. For this homework, some C++ **syntactically correct** code is given, but only on GL. You will be copying the code from an instructor's public location on GL to your HW1 directory. While logged into GL, move your current directory to:

CMSC341/HW1/

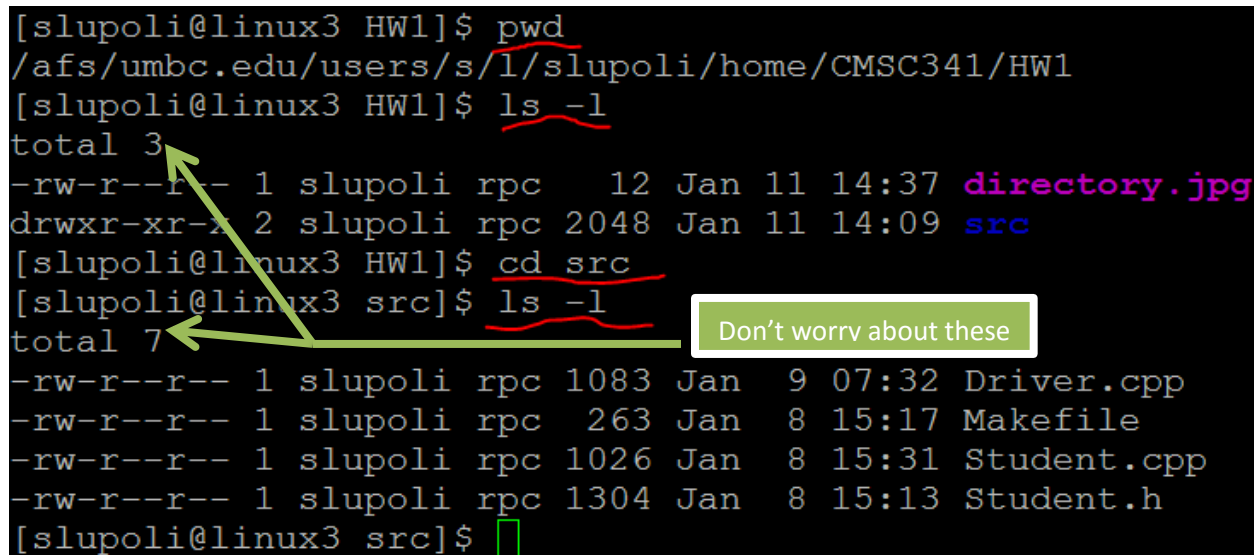
(If new to GL, review the playlist above)

Then copy the code from an instructor's directory to your present directory by typing:

```
cp -r /afs/umbc.edu/users/s/l/slupoli/pub/labCode341/HW1/* .
```

(Yes, space then the dot. The dot is telling us to copy all and sub directories (-r) to our current directory)

To prove you did this correctly, notice and type the commands typed below (ls -l, cd .., etc..) . Your output should look like below, but your username instead:



```
[slupoli@linux3 HW1]$ pwd
/afs/umbc.edu/users/s/l/slupoli/home/CMSC341/HW1
[slupoli@linux3 HW1]$ ls -l
total 3
-rw-r--r-- 1 slupoli rpc 12 Jan 11 14:37 directory.jpg
drwxr-xr-x 2 slupoli rpc 2048 Jan 11 14:09 src
[slupoli@linux3 HW1]$ cd src
[slupoli@linux3 src]$ ls -l
total 7
-rw-r--r-- 1 slupoli rpc 1083 Jan 9 07:32 Driver.cpp
-rw-r--r-- 1 slupoli rpc 263 Jan 8 15:17 Makefile
-rw-r--r-- 1 slupoli rpc 1026 Jan 8 15:31 Student.cpp
-rw-r--r-- 1 slupoli rpc 1304 Jan 8 15:13 Student.h
[slupoli@linux3 src]$
```

Take a screen capture of your success and name it **copied.jpg** and place in CMSC341/HW1/

(This image should ONLY show the results)

#3 Compiling and showing output – 15 pts

(Veterans, be careful here. I am asking for something specific here.) Compiling is the same idea for either Java or C++. On GL, it uses a g++ compiler which won't mean much for some of you now. Remember, compiling is just that. It will not show output. That is a separate step. The code is already ready to go, so let's compile and view the result.

From the directory:

CMSC341/HW1/src/

to compile, type and hit enter after each line below:

```
g++ -Wall Driver.cpp -c
```

```
g++ -Wall Student.cpp -c
```

```
g++ -Wall Driver.o Student.o -o output.out
```

Notice we compiled the .cpp files. But why not the .h?? That, you will learn in class. (Just believe us for now.)

Showing the output is the final step. The last line you typed gave us the file to run the program.

To prove you did this correctly, notice and type the commands typed below (ls -l, cd .., etc..) . Your output should look like below, but your username instead:

```
[slupoli@linux1 src]$ ls -l
total 107
-rw-r--r-- 1 slupoli rpc 1083 Jan 8 15:13 Driver.cpp
-rw-r--r-- 1 slupoli rpc 59648 Jan 8 15:31 Driver.o
-rw-r--r-- 1 slupoli rpc 263 Jan 8 15:17 Makefile
-rwxr-xr-x 1 slupoli rpc 34619 Jan 8 15:31 output.out
-rw-r--r-- 1 slupoli rpc 1026 Jan 8 15:31 Student.cpp
-rw-r--r-- 1 slupoli rpc 1304 Jan 8 15:13 Student.h
-rw-r--r-- 1 slupoli rpc 6784 Jan 8 15:31 Student.o
[slupoli@linux1 src]$ ./output.out
Walter
Henry
100

Jonathan
Laney
75

[slupoli@linux1 src]$
```

Take a screen capture of your success and name it **ran.jpg** and place in CMSC341/HW1/

(This image should ONLY show the results)

#4 Makefile and Cleaning up after yourself (for a change) – 15 pts

That was a lot of work to compile and run. And notice in the last screen some extra files appeared from nowhere! Programmers are notoriously lazy, we don't want to do all that work. (Cry me a river.) A Makefile is a file **we create** in order to reduce the redundant steps we will have to complete while working on a project. More will be covered in class. Let's take a look at the file Makefile.

From the directory:

CMSC341/HW1/src/

Display the Makefile (very small file) to the screen type and hit enter after each line below:

```
clear
```

```
cat Makefile
```

To demonstrate the "clean" target (or block) in the Makefile type and hit enter after each line below:

```
ls
```

```
make clean
```

```
ls
```

Notice after the second "ls", many of the object files that we don't edit were removed. BEFORE YOU SUBMIT ANY CODE, you are to "clean" up, then submit. Take another look at the Makefile and figure out why "clean" works. Be careful, the command below would be devastating in that block:

```
rm *
```

Compiling is also redundant, since we will be compiling many times during a project to see if it works. Another look at the Makefile shows those same lines you typed in by hand will now work with typing:

```
make
```

That command compiled it all for you. Notice though, it did not run the project. This file will change slightly for your other projects since the file names are different and may require command line arguments. To prove you can do this correctly, notice and type the commands typed below (ls -l, cd .., etc..) . Your output should look like below, but your username instead:

```
[slupoli@linux1 src]$ ls -l
total 107
-rw-r--r-- 1 slupoli rpc 1083 Jan 8 15:13 Driver.cpp
-rw-r--r-- 1 slupoli rpc 59648 Jan 8 15:31 Driver.o
-rw-r--r-- 1 slupoli rpc 263 Jan 8 15:17 Makefile
-rwxr-xr-x 1 slupoli rpc 34619 Jan 8 15:31 output.out
-rw-r--r-- 1 slupoli rpc 1026 Jan 8 15:31 Student.cpp
-rw-r--r-- 1 slupoli rpc 1304 Jan 8 15:13 Student.h
-rw-r--r-- 1 slupoli rpc 6784 Jan 8 15:31 Student.o
[slupoli@linux1 src]$ make clean
rm -rf *.o
rm -f *.out
rm -f *~ *.h.gch *#
[slupoli@linux1 src]$ ls -l
total 7
-rw-r--r-- 1 slupoli rpc 1083 Jan 8 15:13 Driver.cpp
-rw-r--r-- 1 slupoli rpc 263 Jan 8 15:17 Makefile
-rw-r--r-- 1 slupoli rpc 1026 Jan 8 15:31 Student.cpp
-rw-r--r-- 1 slupoli rpc 1304 Jan 8 15:13 Student.h
[slupoli@linux1 src]$ █
```

Take a screen capture of your success and name it **cleanedUp.jpg**. (This image should ONLY show the results)

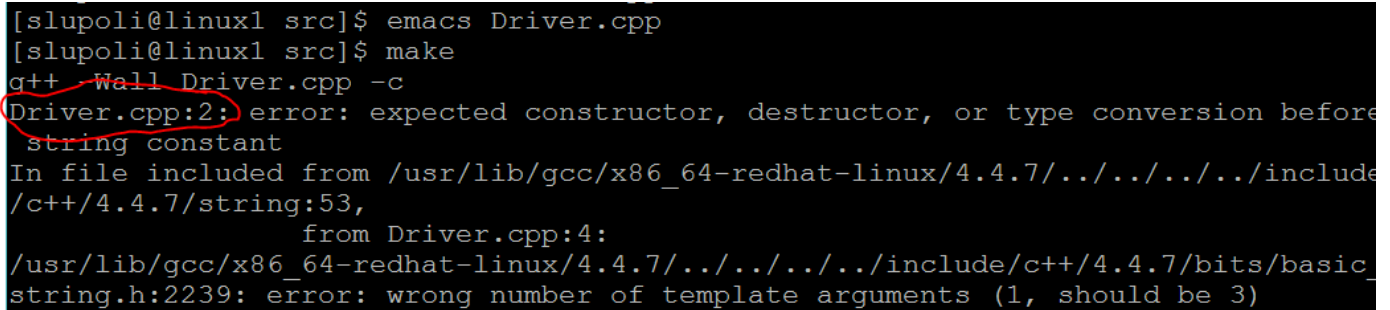
#5 Reading g++ compiling errors and why GL is important – 15 pts

Many of you will not like creating your programs on GL. An alternative will be shown later. But... **EVERY** project needs to be able to run on GL since that is the platform the grading will be done on. So no matter where you create the code for your project, **it needs to be able to compile and run on GL for full credit.**

Let's create an error just to see how g++ would handle it. In "Driver.cpp" remove the first "#" from the top.

If you are new to Emacs, take a look at the video [here](#).

Go ahead and compile. While the error message is massive, in reality, we start from the top, or first issue. As shown below, it tells us line 2, exactly where and which file we removed the "#".



```
[slupoli@linux1 src]$ emacs Driver.cpp
[slupoli@linux1 src]$ make
g++ -Wall Driver.cpp -c
Driver.cpp:2: error: expected constructor, destructor, or type conversion before
string constant
In file included from /usr/lib/gcc/x86_64-redhat-linux/4.4.7/../../../../include
/c++/4.4.7/string:53,
from Driver.cpp:4:
/usr/lib/gcc/x86_64-redhat-linux/4.4.7/../../../../include/c++/4.4.7/bits/basic
string.h:2239: error: wrong number of template arguments (1, should be 3)
```

To prove you did this correctly, display your output which should look like above, but your username instead. Take a screen capture of your success and name it **error.jpg**. (This image should ONLY show the results)

Fix the file "Driver.cpp" and recompile.

Home based programming – 0 pts

Many choose to program on their own desktops or laptops. Any C++ IDE ([Eclipse](#), [Visual Studio](#), etc...) is fine. Just remember, when complete, it MUST run on GL and have a Makefile.

For those that are new to this, a set of videos was created for this situation. Please take a look [here](#).

And creating and transferring programs is covered [here](#).

Since this would be tough to prove, there is nothing required for this exercise.

#6 Submitting Part 1 – 15 pts

Finally, submitting. Remember, you can only submit from GL. All of your material/code/etc... must be in a directory on GL first. You also need to be in the current directory with the material you want to submit. While logged into GL, move your current directory to:

CMSC341/HW1/

Then copy all of the code and subdirectories (-r) from your directory to the lead instructor's directory (submitting) by typing:

```
cp -r . /afs/umbc.edu/users/s/l/slupoli/pub/cs341/username/HW1/ (make sure to replace it with your username)
```

If you have an error here, it is most likely you added the class late. Email the lead instructor with a screen capture of the error message.

That was submitting!! Let's check to see if you were successful.

Type and hit enter after each line below:

```
ls /afs/umbc.edu/users/s/l/slupoli/pub/cs341/username/HW1/
```

```
ls /afs/umbc.edu/users/s/l/slupoli/pub/cs341/username/HW1/src/
```

this will show what files have been "submitted" to the lead instructors. (worth 8 pts)

To prove you can do this correctly, notice and type the commands typed below (ls -l, cd .., etc..) . Your output should look like below, but your username instead:

```
[slupoli@linux3 HW1]$ cp -r . /afs/umbc.edu/users/s/l/slupoli/pub/cs341/slupoli/HW1/
[slupoli@linux3 HW1]$ ls /afs/umbc.edu/users/s/l/slupoli/pub/cs341/slupoli/HW1
cleanedUp.jpg copied.jpg directory.jpg error.jpg ran.jpg src
[slupoli@linux3 HW1]$ ls /afs/umbc.edu/users/s/l/slupoli/pub/cs341/slupoli/HW1/src/
Driver.cpp Makefile Student.cpp Student.h
[slupoli@linux3 HW1]$
```

Take a screen capture of your success and name it **submit.jpg**. (This image should ONLY show the results, (worth 7 pts)

#7 Submitting Part 2 – 10 pts

Since we all like being as “efficient” as possible, instead of typing the long /afs/... we can create a symbolic link. We will create this and submit all of the code and jpgs created for this homework.

While logged into GL, move your current directory to:

CMSC341/HW1/

type and hit enter after each line below:

```
ln -s /afs/umbc.edu/users/s/l/slupoli/pub/cs341/username ~/cs341class
```

```
cp -r . ~/cs341class/HW1/
```

Your complete submission will be proof you completed this. What we expect to see in your submitted directory is below:

```
[slupoli@linux3 HW1]$ cp -r . ~/cs341class/HW1/
[slupoli@linux3 HW1]$ ls /afs/umbc.edu/users/s/l/slupoli/pub/cs341/slupoli/HW1
cleanedUp.jpg copied.jpg directory.jpg error.jpg ran.jpg src submit.jpg
[slupoli@linux3 HW1]$ ls /afs/umbc.edu/users/s/l/slupoli/pub/cs341/slupoli/HW1/src/
Driver.cpp Makefile Student.cpp Student.h
[slupoli@linux3 HW1]$
```

#8 Discussion Board – 10 pts

Remember, this homework is to be completed in order. Complete this problem only when you have finished all of the previous problems. In CMSC 341, we use Piazza instead of email. This discussion board is broken down by assignment, is monitored by TAs, and is meant for constructive questions and answers to other student questions. In Blackboard, there will be a link to this Piazza discussion board, but you should also get an invite. You are to add one posting to the HW1 board with only your name and section.