Starting Out with Hello!

Due Tuesday, January 28 at 8 a.m.

Revision 0

CSE 1325 - Spring 2020 - Homework #1 - 1

Assignment Overview

Way back in 1972, Brian Kernighan wrote the *very first program* to ever print "hello, world" to the console, as part of the Basic Combined Programming Language (BCPL) documentation. From BCPL, he derived a language called B. From B, he and Dennis Richie derived a new systems language called C, used to write a new operating system called Unix (Hi, Mac fans!). From C, Bjorne Stroustrup created C++, with which we'll soon be *very* familiar.

And the first program in *each* of those languages also printed "hello, world".

So now that you're learning C++, tradition clearly dictates that we say hello to the world! We already showed you versions of this program in Lecture 01 for several languages - C, C++, Java, and Python. **Compiling and running this program is a great way to ensure your environment is operating correctly, and to start exploring its features.** You'll find a *lot* of additional hints after the instructions on this page.

Full Credit

Create a GitHub account, and post its name as your solution to Assignment P1 in Canvas. Under Ubuntu Linux 18.04, create and clone a GitHub repository **named cse1325** (capitalization matters!), then create a P01/full_credit subdirectory. In it, create and run the C++ “Hello, World!” program using cout, **replacing “World” with your name**. Take a screenshot of your program running, and name it hello.png.

Add hello.cpp and hello.png to git (**git add hello.cpp hello.png**), commit them with an informative commit message (**git commit -m "*your message here* **), and then push that commit to GitHub (**git push**).

Bonus

Instead of hard-coding your name, *ask the user for their name using cin*. Then print “Hello, [name]”, where [name] is whatever name they enter. Add a bonus subdirectory to P01 (**cse1325/P01/bonus**) with your source file named hello.cpp and one or more screenshots named hello1.png, hello2.png, etc. showing your program in action with *at least* 3 distinct inputs, e.g., multiple word names, Unicode characters, and a zero-length string.

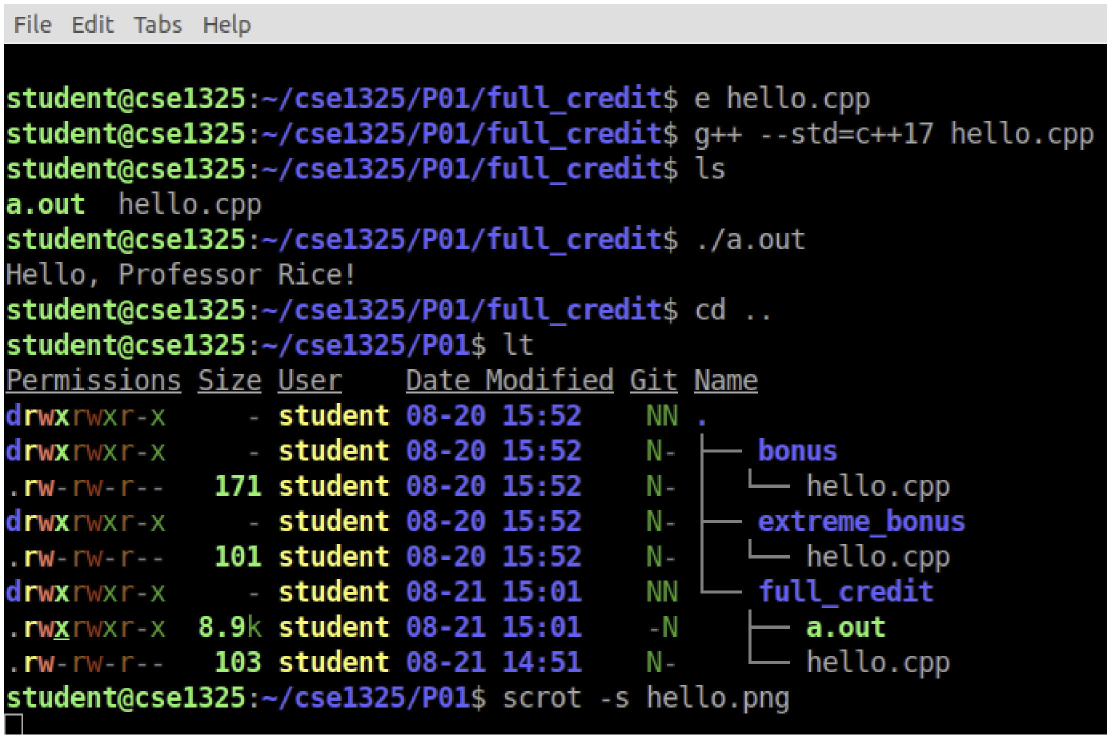
Add, commit, and push all files to GitHub.

Extreme Bonus

Instead of asking the user for their name, **automatically identify the current user** and print “Hello, [name]!” without any input or hard-coded names in the source code, where “[name]” is something that uniquely identifies the user, such as a user name, proper name, etc. Add an “extreme_bonus” subdirectory to P01 (**cse1325/P01/extreme_bonus**) with your source file named hello.cpp and one or more screenshots named hello1.png, hello2.png, etc. showing your program in action. For maximum credit, test the program *in at least two user accounts* and *on at least two operating systems*.

Add, commit, and push all files. Additional information that you may find helpful follows.

Hint: How to Take a Screenshot



```
student@cse1325:~/cse1325/P01/full_credit$ e hello.cpp
student@cse1325:~/cse1325/P01/full_credit$ g++ --std=c++17 hello.cpp
student@cse1325:~/cse1325/P01/full_credit$ ls
a.out  hello.cpp
student@cse1325:~/cse1325/P01/full_credit$ ./a.out
Hello, Professor Rice!
student@cse1325:~/cse1325/P01/full_credit$ cd ..
student@cse1325:~/cse1325/P01$ lt
Permissions Size User    Date Modified Git Name
drwxrwxr-x     - student 08-20 15:52    NN .
drwxrwxr-x     - student 08-20 15:52    N- ├── bonus
.rw-rw-r--   171 student 08-20 15:52    N- │  └── hello.cpp
drwxrwxr-x     - student 08-20 15:52    N- ├── extreme_bonus
.rw-rw-r--   101 student 08-20 15:52    N- │  └── hello.cpp
drwxrwxr-x     - student 08-21 15:01    NN └── full_credit
.rwxrwxr-x  8.9k student 08-21 15:01    -N    ├── a.out
.rw-rw-r--   103 student 08-21 14:51    N-    └── hello.cpp
student@cse1325:~/cse1325/P01$ scrot -s hello.png
```

Lubuntu 18.04 (part of the VM appliance) includes a program called scrot (for screenshot). Type `man scrot` for the manual, using PgUp and PgDn or your mouse wheel to explore, and "q" to quit.

- **To capture the entire screen** to hello.png, simply type “scrot hello.png”. Done and done.
- **To capture a single window** to a file named (ahem) hello.png, e.g., your terminal window, type `scrot -s hello.png`, then click the terminal window with your mouse. The “-s” means “select” what to capture. You can view the screenshot using “e hello.png”.
- **To capture a rectangular area** from the screen to file hello1.png, type `scrot -s hello1.png` as well, but move the mouse to the upper left corner of the rectangle you want to capture, hold down the primary mouse button, drag the mouse to the lower right corner, and release.

If you installed your own Linux operating system and scrot is missing, simply install scrot using `sudo apt install scrot`. Of course!

You may use any screenshot utility you prefer, though, as long as your screenshots are delivered in PNG format.

Hint: Working With GitHub

Professional software developers use version control, so we will, too. Don’t panic – it’s not as hard as you may think.

For the first homework only

1. As we demonstrated in the first lecture, and as discussed in the slides and in "GitHub in 5 Pages" (*did we document this enough?*), create an account at github.com.
2. Using this account, create a **private** repository named cse1325 (capitalization is important - *not* CSE1325, and *not* cse_1325!).
3. Add the professor and TAs as collaborators.
4. **Submit the URL as your solution in Canvas**. If you don't do this, we can't grade your work and you'll get a 0!
5. Clone your cse1325 repository to your computer.

This effort should make all future assignments very simple to deliver for grading.

Full Credit

1. In bash, change to your cloned repository. Something like `cd cse1325` may work, depending on where you cloned your repository. Or find your repository in the file manager (next to Start), right-click and select “Open in Terminal”.
2. Type `mkcd  P01/full_credit` to create and change to the directory for your full credit solution. (If you created your own Linux environment, ask about our custom .bash_aliases!)
3. Create your solution, e.g., `e hello.cpp` to create and edit the source file.
4. To compile hello.cpp, use `g++ --std=c++17 hello.cpp`. If you see errors, fix them in the text editor. If not, when you type `ls` you should see an executable file (in green text) named a.out.
5. Run the program using `./a.out`.
6. Take the required screenshot by typing `scrot -s hello.png` and clicking the terminal window.
7. Add the source file and screenshot to git using `git add hello.cpp hello.png`.
8. Commit your added files to the local repository using `git commit -m “P01 fc First Commit”` (or whatever comment you like).
9. Upload your commit to GitHub using `git push`.

At this point, your full credit work is done! If you’d like to try the bonus (and you should), keep going!

Bonus

1. Create and change to the bonus directory using `mkcd ../bonus`.
2. Repeat steps 4-10 above to develop, add, commit, and push your bonus solution.

At this point, your bonus work is done! If you’d like to try the extreme bonus (and you should), *keep going*!

Extreme Bonus

1. Create and change to the bonus directory using `mkcd ../extreme_bonus`.
2. Repeat steps 4-10 above to develop, add, commit, and push your bonus solution.

You’re done!

More on This (and Future) Homework

Don’t worry, it gets MUCH more interesting (and challenging). This homework is primarily designed to help you acclimate to the new environment.

All homework must be turned in via GitHub by the deadline. **Late submissions are never graded** without an exceptionally good excuse, but the grader will grade the latest submission prior to the deadline unless you specify otherwise.

Start and finish early. It's a great habit to develop for life.

You may push as many times as you like, and we will only grade your latest commit on GitHub that is prior to the deadline. (We won't go snooping around your earlier commits looking for something to criticize - first, we don't have time, and second, you're *expected* to make beginner mistakes when you're a beginner! :-D) Don’t accept a 0 after completing 90% of the work! **Commit and push early and often.**

It is possible – indeed, rather easy – to earn more than a 100 on homework assignments. Going beyond the Full Credit requirements will help you to grow your skill, and also result in bonus points. If you have more than a 100 average on your homework at midterm or the end of the class, the additional credit WILL be included in calculating your final average. This is a GREAT way to recover from the occasional missed exam question.

Please do not wait until the end of the semester to seek “extra credit work” to pass the class or improve your GPA. **Extra credit work is right here! Start today and avoid the rush.**

The Suggested Solution

Don't get *too* excited - you won't get the suggested solution until *after* the due date and time! :-D

The professor for this class provides example code, homework resources, and (after the due date) suggested solutions via his cse1325-prof GitHub repository.

- If you haven't already, `cd` and clone the professor's cse1325-prof repository with ``git clone https://github.com/prof-rice/cse1325-prof.git``to create a new directory named "cse1325-prof". This will reflect the GitHub repository contents.
- The cse1325-prof directory doesn't automatically update. To update it with the professor's latest code at any time, change to the cse1325-prof directory (`cd ~/cse1325-prof`) and pull (`git pull`). `~` (the tilde) means "my home directory" in bash.

Since cse1325-prof generally contains the directory structure you need in advance, before you start the homework, you *may* be able to copy it directly using e.g., `cd cse1325 ; cp -r ../cse1325-prof/P01 .` (for the first assignment). The `-r` on the copy (`cp`) command means "recursively", that is, copy the entire directory structure and its contents, not just the first level. This might save you some time and typos.