CSC 211 Lab 2: Command-line arguments and Echo

Fall 2018

About this lab

Last lab, you explored the Linux `bash` command-line environment. Several of the commands you used, such as `cd` and `cp`, take command-line arguments, which are very much like the arguments to a function, except they apply to the execution of a whole program. Another command that takes command-line arguments is `echo`, which simply repeats back all its command-line arguments, separated by spaces.

Here, you're going to write a program, not in the Mimir editor, but using the CS50IDE environment.

Find a partner

You should do this lab with a partner. You should be able to submit as a **group** in Mimir when you are done.

Get into the CS50 IDE

You should know how to do this by now.

In the Terminal window, play with the `echo` command a little bit. Type `echo hello` followed by enter. Play around with some other arguments to echo, including multiple arguments (with spaces in between) such as `echo hello there`.

Download the lab framework

In the terminal, type `git clone https://github.com/csc211/lab2` (followed by enter, of course, which you should assume by now).

You'll see some informational output. Type `ls` and note that you'll see a `lab2` directory. Change directory into there by typing `cd lab2`. Type `ls` and see what's there. Now, go to the file browser and open up the `lab2` directory within your workspace directory. Note that there are a few files in there. Two of them are these lab instructions (one as a `.md` file, which is a simple formatting language known as markdown, and the other as a PDF). One is called `compile` and it's a script that will let you compile your code. The other is an empty file called `echo.cpp` which you need to write.

Start working on `echo.cpp`

- You need to write a `main()` function. For this lab, you probably won't need any helper functions, though.
- Your program must print out each of its command-line arguments, with one space in between each argument. After this, it must print a newline character.
- If your program is given no command-line arguments, it should simply print a newline character.
- This behavior is **identical** to the command `echo` you just tried out. You're re-implementing the Linux `echo` program.

Things to remember

- You will compile your program by typing `./compile` (followed by enter, of course).
  - This will produce an **executable binary** called `echo`
  - Don't try to write the whole thing and then compile it.
    - Instead, compile whenever you think the program *might* compile.
    - This will save you a huge amount of time.
    - Remember to look at the top-most compiler errors, not the bottom-most ones.
- You can then run your program by typing `./echo` (followed by any arguments you wish to provide to it).
- You will need the type signature for `main()` that allows command-line arguments:
  - `int main(int argc, char *argv[])`
- Remember that `argc` contains the number of elements in `argv`
  - But also remember that `argv[0]` is the name of the program, which you want to skip
- You need to print out each element of `argv` except for `argv[0]`

- You need to print a space after each one **except** the last one.
- After the last one, you need to print out a **newline character**
  - You can print out a newline character with `std::cout << endl`
- Given a `char *` named `fred`, you can print it out with:
  - `std::cout << fred`
  - But, in order to be able to use `std::cout` you will need to put `#include <iostream>` at the very top of `echo.cpp`
- Think about what you might need to do differently with the last entry in `argv`. Remember that you are printing a space **between** each entry, but not after the last one. After the last one, you're printing a newline. So, think carefully about whatever **programming construct** you use to solve this problem.

Submitting the lab

Go to Mimir, and go into Lab 2. **Create a group** for you and your partner, as you will submit one solution for both of you. Then, upload your `echo.cpp` file to Lab 2. You don't need to upload any other files.