Lab 1: Hello World

Introduction

For many of you, this is your first exposure to Linux, so this first lab holds your hand and takes you step by step. You will be logging into icarus using an SSH client and creating a one line script that prints "Hello world!" on the console. You will then execute your first-ever script, then run a special testing program to verify that your script gives you 100% credit towards this assignment. Warning: you will be working at the Linux command-line in this course.

Steps

Follow the steps below to install an SSH client onto your Windows workstation.

Clone your private repo on github.com

In this class, we use a source code versioning technology called `git` and a website called `github.com`. You will create a private repo on github.com, create a local copy of that repo, write and test your script, then upload your updated repo to github.com for grading. As long as you do as instructed, you don't have to know anything about git. If you are curious, I have placed links to more information about git and github.com on the Resources page in Canvas. I strongly suggest you learn more about `git` and `github.com`

In the assignment module in Canvas, you should find a link to a Canvas page entitled "Clone your Lab 1 Repo". Find it in Module 1 for this lab. Click on it to accept this assignment in Github Classroom and create your private Github repository. You will do this once for each assignment. You will be shown your private Github repository (or repo) name. Remember this name for future `git` commands.

Each time you create a private repo in Github Classroom, an *experimental* continuous integration server called Jenkins is running in your instructor's basement that attempts to grade your assignment and send you an email. Because you haven't even begun to work on the assignment, all of the tests will fail. You may disregard this first email. Each time you perform an git push, another email will be sent containing a test report. Pay attention to those subsequent emails. More will be said about Jenkins as we move forward in the class.

Login to icarus

If you are a Mac or Linux user, or you know how to enable the SSH client in Windows 10, you do not need (nor can you run) PuTTY - simply open a terminal window and type:

```
ssh USERNAME@icarus.cs.weber.edu
```

Skip to the "Login to icarus" step to continue.

See comments in the section below entitled “Login to icarus” about the format of your username and password.

Configure PuTTY and login to the Linux server icarus

If PuTTY is not already installed on your workstation, go to https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html and download it. Run the installer to install PuTTY.

Start putty.exe and create a stored session called cs3030:

- Under Category on the left, select Session
- Host Name is `icarus.cs.weber.edu`
- Protocol is `SSH`
- Saved Sessions name is `cs3030` and then click the SAVE button
- Under Category on the left, select Window→Colours
- Default Foreground color should be Red-Green-Blue values of 0-0-160 (dark blue)
- Default Bold Foreground color should be 0-0-255 (royal blue)
- Default Background color should be 255-255-255 (white)
- Default Bold Background color should be 85-85-85 (dark grey)
- Cursor Text should be 0-0-0 (black)
- Cursor Color should be 0-255-0 (lime green)
- Under Category on the left, select Session, enter `cs3030` in the Saved Sessions box and click the SAVE button
- Click LOAD and PuTTY should begin the login process to icarus. You should get a PuTTY security alert about the security key. Click YES.

Login to icarus:

- icarus should prompt you for a username. Your username should be `flnnnnn`, where `fl` are your initials in lower case and `nnnnn` is the last 5 digits of your W number. Your password will be `Firstcs!`, where `First` is your first legal name with the first letter capitalized and `cs!` is just that.

Here is an example username and password for a fictitious student named David Johnson, with W number W12345678:

username	`dj45678`
password	`Davidcs!`

If you did all of this and it didn't work, please consult the icarus account information at https://icarus.cs.weber.edu/Technical_Support.html.

Choose an editor

There are a cornucopia of text editors available for UNIX/Linux. They all basically accomplish the same thing, editing text, so it doesn't much matter which one you choose, as long as it's one that you're comfortable with.

Here's a (rather short) list of editors available on icarus:

- vi - Real UNIX/Linux people use vi. (on Icarus we are actually using `vim` which is short for "vi iMproved".)
- pico/nano - pico (alias nano) is a fairly easy-to-use and intuitive editor. This may be the editor that some of you will end up using.

I will help you with `vi`. You're on your own for the others and at this time only `vi` and `pico/nano` are installed on icarus. If you wish to download and compile another editor of your choice, I commend you for your ambition but you are on your own.

Secure your account

Linux users have a profile script in their home directory that executes every time you login. Here, we will edit this script and change a value that will prevent other users from seeing files you create. It will also give you a chance to see how other shell scripts are written.

Type `vi ~/.profile` and look down about eight lines from the top for a line like this (if your .profile is empty or does not exist, exit vi with `ESC`, `:q!` and `vi ~/.bash_profile` and add the umask command on line 2, which should be blank):

```
#umask 022
```

Remove the pound sign by hitting ESCape to return to command mode, placing the cursor over the pound sign with the arrow keys and hitting 'x' once. Then type 'i' to get into edit mode and change the 022 (zero two two) to 077 (zero seven seven). Use backspace to delete in vi. It should now look like this:

```
umask 077
```

Hit ESCape and save the file with `:wq`. To test your changes, type

```bash
. ~/.profile
```

(or `. ~/.bash_profile` if you didn't have a .profile)

If you receive any strange error messages, review your work.

Prepare to use `git`

The `git` program is installed on icarus and is ready for you to use. There are two git commands you should enter the first time you wish to use `git` on icarus, and they only need to be performed once. Type these commands, performing the required substitutions:

```bash
git config --global user.email "YOUR_WEBER_EMAIL_ADDRESS"
git config --global user.name "YOUR_FIRST_AND_LAST_NAME"
```

For example, if your name was "David Johnson", you would enter the above commands thus:

```bash
git config user.email "davidjohnson@mail.weber.edu"
git config user.name "David Johnson"
```

Use `git` to clone (download) a copy of your private repo to your local machine

To obtain a copy of your private repo on your local machine, enter this command:

```
git clone https://github.com/cowancs3030<SEMESTER>/lab1-YOURGITHUBUSERNAME
```

The http: address is the Github repo name you were given when you cloned your Lab 1 repo in Canvas. Change `YOURGITHUBUSERNAME` to your Github username. <SEMESTER> will be something like "spring20". A folder will be created in your home directory named `lab1-YOURGITHUBUSERNAME`. Change into the repo's root folder by typing:

```
cd lab1-YOURGITHUBUSERNAME
```

Once in your repo's root folder, execute `ls -al` to see what files and folders were cloned from your private repo. You will be adding a file to the root folder named `hello`.

There are several files and folders included in your repo, which you may inspect **but you should not modify them or their contents in any way**, including:

`Jenkinsfile`	a file used to test your code on our server.
`features`	a folder containing test scripts available for you to test
`tmp and bin`	folders of files needed by the test scripts
`Gemfile`	a file needed by the test scripts

Personalize your environment

Change your password by entering `passwd` and following the prompts.

Now you are ready to create your first script, `hello`.

Write your "Hello World" script

Create a new file in `vi` by typing `vi hello` and hit `i` to insert. Then type in these lines (the lines beginning with # are comments, except the very first one):

```bash
#!/bin/bash (1)
# (Your name) (2)
# Lab1 - Hello World (3)
# CS 3030 - Scripting Languages (4)

echo "Hello, world!" (5)
```

1	The "shebang" that identifies this file as a bash script
2	Your first and last name
3	The name of this assignment
4	The name of this class
5	The line of code that actually outputs the words "Hello, world!"

Hit the `ESCape` key to get out of insert mode, then `:wq` RETURN to save the file and exit `vi`. Type

```
ls -al hello
```

(the Linux equivalent of the Windows `dir` command) to list the hello file in your subdirectory and to verify that your script file is named correctly. Linux is case-sensitive, so if you called it `Hello` or `HELLO`, you need to either rename it or delete it and begin again.

Should you make a mistake, you can use one of the following LINUX commands (these are examples):

Rename a file or folder	`mv oldfile newfile`
Delete a file	`rm file`
Remove an empty folder	`rmdir folder`

There is no such thing as UNDO on the commandline, so be very careful when using these commands.

The comment lines (the ones with the # in column 1) are good programming practice; place at the top of every assignment you turn in. I like to call it "documentation".

Make your script executable and run your script

Type the following command at the command prompt to make your script executable by the shell:

```
chmod 700 hello
```

The chmod command changes the permissions of the file and makes it executable. In Lab 2 we will discuss tools like chmod and how we use them. The `/bin/bash` in the line 1 tells the operating system what shell interpreter should execute this particular file. Execute the `ls` command again:

```
ls -al hello
```

You should see something like this:

```
tedcowan@icarus:~/lab1-tcowan$ ls -al hello
-rwx------ 1 tedcowan tedcowan 85 Jan  3 19:27 hello
tedcowan@icarus:~/lab1-tcowan$
```

If you see `-rwx` in the permissions at the far left on the line with the file hello, it means you the owner have Read, Write and eXecute permissions on this file. If you see anything else, or other `r`, `w` or `x` letters beyond the first three, re-execute your `chmod` command as stated above.

Notice that in the above example, the folder is named after my repo for Lab 1, with my Github username of "tcowan". Yours will display with your Github username.

Type `./hello` at the command prompt to run your script. Please note that you must type the `./` before the hello or the system will not be able to find your script. This is because the current working directory is not included in your PATH. (Type `echo $PATH` if you want to see what it contains)

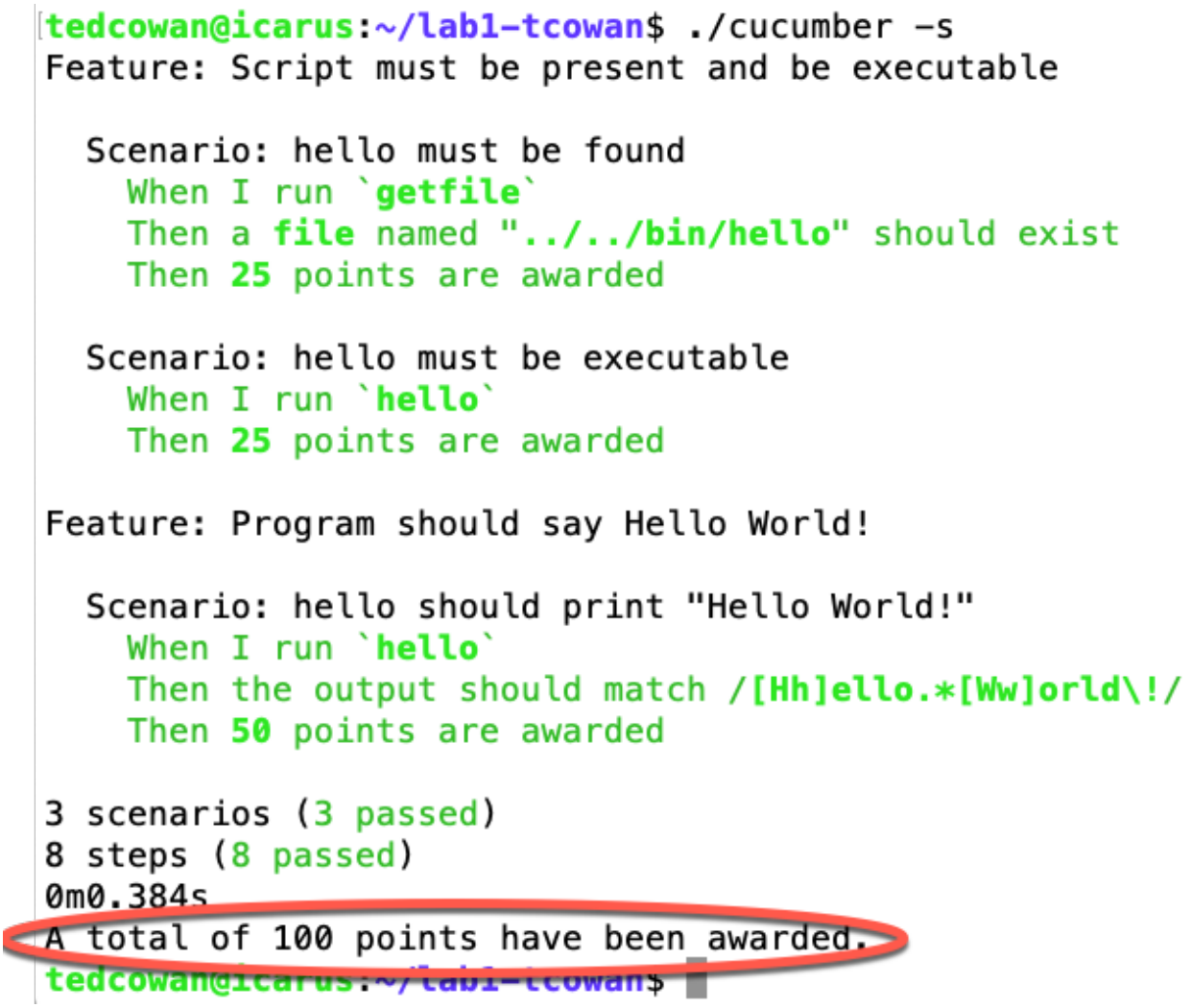
If your script prints "Hello, world!", you are successful. However, if it doesn't, review steps 4-5 and make sure your script is called hello and is executable.

Test your program using Cucumber

Cucumber is a testing tool that is used to verify program output. We are using it in CS 3030 to test the output of all of your lab assignments this semester. More will be said about cucumber as we move forward in the semester. Cucumber is installed on icarus and the configuration fies are included in your private repo. This command must be executed in the same folder as your hello script.

```
./cucumber -s
```

The cucumber report should look something like this:



The last line of the cucumber report is the one that matters to you. In this case, this assignment is worth 100 points (see Canvas for the point value of each assignment) and you have received all 100 points if you see this message. If you see less than full credit, the scenarios and steps messages will tell you which tests passed and which failed. Failed steps are printed in red in the cucumber report. Passed steps will be printed in green. Skipped steps (steps that could not run because previous steps failed) are printed in a blue-green color.

Submit your assignment code for grading

You must push your script in your private repo to github.com to receive any points, for all assignments in this course.

First, add your hello file to the local `git` repo with these commands (Change the COMMIT MESSAGE to something descriptive, such as "added hello script file"):

```bash
git add hello (1)
git commit -m"COMMIT MESSAGE" (2)
git push origin master (3)
```

(1) stages your new file `hello` for the commit.

(2) commits the staged file.

(3) Uploads your entire repo back up to github.com for grading.

I will grade your assignment by rerunning the cucumber program and using its output to grade your work. You may run the cucumber program at any time as you are developing your scripts to test your work.

Sometimes Cucumber will hide or obscure errors in your scripts as you work out the bugs. I recommend first running the script manually without cucumber to ensure compliance with the requirements, then run cucumber to verify that your program is correct and that you receive the desired number of points.

Files created for this lab: `hello`

I request that you to finish this lab as soon as humanly possible so that your access to icarus is verified.

Grading

Here is how you earn points for this assignment:

FEATURES	POINTS
Must-Have Features	
Script is named correctly and found in its proper place in your repo	25
Script is executable	25
Required Features	
Script prints “Hello, world!”	50
Grand Total	100