GOAL: Learn the procedure to enter a program, compile and build it, and get it to work. Most of this will be needed ALL semester for almost every assignment.

Step 1. Make sure your VPN software, GlobalProtect is connected **if** you are connecting from a non-campus computer and not on the campus network. Log in using your Sac State credentials to a Coding computer (or your own Linux computer).

Step 2. LOGGING ON TO a Linux computer:

Use one on the Plan documents to help you log in.

Plan A Putty and File Transfer. Windows.docx
Plan B Mac Alternatives.docx
Plan C Mobaxterm. Windows.docx

STEP 3: GETTING SET UP TO START

You should now be logged into a Linux computer, and in your home directory.
Type the following instructions:

mkdir csc60	Create a directory (folder) for this class
cd csc60	Move to the new directory
mkdir lab2	Create a directory for this assignment
cd lab2	Move to the new directory

STEP 4: START ENTERING YOUR PROGRAM.

At the prompt ">" or "$", type: **vim lab2.c**

Go into insert mode by typing: **i**

Start typing in your program in this window. (Remember, no mouse)
For this Lab2, enter in the program that appears below ***inside the box*** below.
(PS: You **don't** have to create the box.)
Type your own full name (both first and last) in all the places needed.
Start typing at the left edge of the screen.
Use the indentation style as shown.

Attribution for the quote is required. You may print the quote & attribution in either one or two *printf*s.
Pay attention to your punctuation and spelling. Presentation is important and your instructor is picky about it.

→ **more on next page**

***Teacher Comments**:*

```
/*------------------------*/
/* Your Name Here   */
/* Lab 2                */

#include <stdio.h>
#include <stdlib.h>

int main (void)
{
   printf("\nLab 2. \n\n");
   printf("Hello World.\n");
   printf("Hi, Your Name. \n\n");
   printf("Quote. \n\n");

   return EXIT_SUCCESS;
}
/*------------------------*/
```

Don't count the dashes. Approximate.
Put Your-Name here, both first & last.

{Preprocessing directives for
{the compiler.

Line required in each program.

Be sure to indent for clarity
Put your First & Last name here!
***Put your quote here** instead of the word "Quote"*
Attribution for the quote is **required**.
Capitalize EXIT_SUCCESS

5: SAVE YOUR WORK, COMPILE IT, AND SEE THE RESULTS.

Get out of insert mode by typing: **Esc** (the escape key)
To save your work and quit, type: **:wq**
The shell prompt returns.
Type: **gcc lab2.c**
(This compiles the program, and if successful,
the executable output is sent to a file called **a.out**)
If you have compile errors, they will appear, and they will need to be fixed.
The prompt returns.

If you have no errors, type **a.out** and the output of your program will display.
If typing **a.out** does NOT work, try: **./a.out**

The results of my program appear on **next page**.

```
[bielr@ecs-pa-coding1 lab2]$ ./a.out

Lab 2.

Hello World.
Hi, Ruthann Biel.

Be yourself; everyone else is already taken.
- Oscar Wilde

[bielr@ecs-pa-coding1 lab2]$
```

NOTICE: the attribution (WHO created the quote) is required.

Comments:

- Your instructor is very picky about presentation.
- There should be an empty line between *a.out* and *Lab 1,* and again at the end before the prompt returns.
- I expect your output to have empty lines where they show above.
- Use proper English and proper punctuation in your quote.
- Save your creativity for the quote.

Standardize your indentation:

If you are not sure about your indentation,
get out of Insert Mode then…type: **Esc**
type: **:1** which takes you to line 1
type: **=G** which will standardize your indentation

If you have Errors:

If you have errors, it is OK, a normal course of events. Examine the Error Message list. Sometimes the second or third message makes more sense than the first error message. One code error can cause SEVERAL error messages.

Fix your errors and save your changes. Go back to the top of STEP 5.
Repeat until you have NO ERRORS.

MAJOR REMINDER.

Every time you change the code, you must **redo** the COMPILE (which is the **gcc** line) before you run the program, or you will NOT see any changes when you run the program (a.out). !!!

STEP 6. PREPARE YOUR FILE FOR GRADING.

When all is well and correct, and you are still on our Linux machine…

- type: **script StudentName_lab2.txt** Script will keep a log of your session.
  Please use Your name instead of "StudentName"
  Use both your first and last name.
- At the prompt, type: **gcc lab2.c** To compile the program.
- At the prompt, type: **a.out** or **./a.out** as needed to run the program
- After the program run is complete, type: **exit** To leave the script session.

NOTE: If you forget to type **exit** to leave *script*, your script file will be empty!!!
(How can you tell if the script file is empty?
Type: **ls -l**
If the script file shows a length of zero, it is empty.)

STEP 7: Move your files to be accessible to your browser.

Use one on the Plan documents to help you move files.
Plan A Putty and File Transfer. Windows.docx
Plan B Mac Alternatives.docx
Plan C Mobaxterm. Windows.docx

STEP 8: Turn in your work. 18 points

Go to Canvas and turn in two files:

1. **lab2.c** …the code file
2. **StudentName_lab2.txt** …the script file

This assignment is due by the end of 2/25 for a chance at full points (18 points).
If turned in before the end of 3/4, you lose 2 points. (16 points)
If turned in before the end of 3/11, you lose 2 more points. (14 points)
This assignment will not be accepted for any points after 3/11.

STEP 9: LOG OFF EVERYTHING.

Type "**exit**" when you are ready to leave the Linux computer.
Close as much software and hardware as necessary for safety, depending on your location.