CSE 3100 Systems Programming
Lab #0

Welcome to Lab 0! At the start of each lab, you will receive a document like this detailing the tasks you are to complete. Read it carefully. Try your best to finish labs during the lab session. However, labs are not due right after each lab period. **Normally, lab assignments are due in two days. Lab 0, however, is due in one week.**

This lab is intended to be interactive. Feel free to discuss the assignment with other students. However, please recall that all of your code should be your own! If you have trouble, feel free to raise your hand and ask your TA for guidance!

After lab 0, you should be able to access your account on Coding Rooms, use basic commands in a shell (e.g. to copy/remove/rename files), edit text files like C source files, compile and run simple C code, and submit your work.

1 Coding Rooms

You will use Coding Rooms to do our work for the first week or two of the course. We will learn how to use an IDE, use shell in a terminal, work on C code, and submit your work.

- The first step is to create an account on Coding Rooms and join the CSE 3100 course. You can use the following link. Choose the 'Learner' option when creating the account. If asked for a join code, use the code `C-Lkzatk3`. Link: https://app.codingrooms.com/app?joinCode=C-Lkzatk3
- You should now see the course. On the left side, there are folders 'Code for thought' and 'labs'. Clicking on a folder shows the assignments/projects inside it.
- In Coding Rooms, in the 'labs' folder, select the project lab0. Scroll down, and you should see an editor (window with starter code) and console. This provides an IDE in which you can edit and run your code. In the top right corner of the IDE, you can click the arrows to expand the IDE window.
- In the top left corner of the IDE, you can click on the file icon to see the files you are working on, including text files (such as C files) and executable files.
- You can type commands in the console (which gives you a virtual terminal). Try the `pwd` command (to print the working directory) and the `ls` command (to list the contents of the directory).
- After you have used the console or run your code using the 'Run' button, you should also see an 'Open Desktop' button appear in the bottom left above the console. This is an additional feature, and you may not need to use it. However, you can use it to see a visual desktop, either as a small window on the page, or in another tab (to open in another tab, hover over the button and click the Open in a New Tab icon). Here, you can e.g. open multiple terminal windows.

2 Shell

In the shell (most likely bash), try the following commands (and explore a bit!):

```
pwd                     Show the current directory (where you are).
ls                      List the files and directories in a directory.
ls -l                   The -l flag tells ls to show more details.
mkdir adir              Create a directory. Try different names.
ls -l adir              Lists the files in the directory 'adir'.
cd adir                 Move into the directory 'adir'.
ls -l ..                List the files in the parent directory.
cd                      Go back to your home directory.
cd adir                 Go to 'adir' again.
echo "hi!"              Print 'hi!' to the standard output (stdout).
echo "hi!" > file       Save 'hi!' to a new file called 'file'
cat file                Print the contents of a file to stdout.
grep "hi" file          Search the file for the string 'hi'.
less file               View the file interactively (q to quit).
rm file                 Permanently deletes the file.
cat                     Read from standard input (stdin) and dump to stdout. Ctrl-D to exit.
history                 Show history of commands.
man ls                  Show the manual for the 'ls' command.
man cat                 Show the manual for the 'cat' command.
man scanf               Show the manual for the 'scanf' function.
exit                    Exit from the shell.
```

3 Text editors and your first C program!

To work on a project, you need to go to the corresponding directory in the virtual terminal. Normally Coding Rooms goes to the correct directory when you start a virtual terminal. Under the project directory, you can modify existing files and/or create new files (.c files, .h files, readme's, make files, etc.).

You can use the web editor, or an editor in the virtual terminal, to edit text files, which include your C source code. If you decide to learn an editor in the terminal, there are three primary options: `vim`, `emacs`, and `nano`. It is highly recommended to learn one of them, in addition to the web editor.

Use your favorite editor to open `lab0.c`. If you use the editor in the Coding Rooms IDE, you can click on the Files icon (top left) and you can use use the buttons. Otherwise, here are different ways to do it in a terminal. The following three editors are available in many systems.

```
vi lab0.c
emacs lab0.c
nano lab0.c
```

Add the following C code in `lab0.c`. If you are using the editor in the Coding Rooms IDE, changes should save automatically. Otherwise, remember to save the file after editing.

```
#include <stdio.h>

int main(void)
{
     printf("Hello, World!\n");
     return 0;
}
```

This is your first C program! Before you can run it, you must compile it. You can do so by typing the following command in shell/console, or by clicking the 'Run' button in the Coding Rooms IDE. Note that you can use console/shell at the bottom of the Coding Rooms IDE. If you would like, you can also use the tips in section 1 to open a visual desktop, where you can open multiple terminal windows.

```
cc lab0.c
```

This will compile your C code in `lab0.c` and create an executable called `a.out`. If the process fails, read the error messages, fix the error in your C file (using the text editor), and try again. If there is no error, you will see `a.out` in your current directory.

You can optionally specify the name of the executable that is generated by the compiler.

```
cc -o lab0 lab0.c
```

To run your program, type

```
./lab0
```

4 Something new

Revise `lab0.c`. Use a `while` loop to calculate the sum of positive even integers below 200 and print the result to the standard output. This is very similar to the sum example we have studied. Think about how much you should increase the loop control variable to get to the next even number.

Your program should output the following text:

```
Hello, World!
9900
```

5 Submission

If you think your code works, you should submit it on **gradescope**. You can download the files you have worked on in the file directory interface to the left of the IDE. There is a link to gradescope in the HuskyCT, you should already be added to the course with your UConn email.

Please remember to submit your work before the deadline for each assignment and exam.

Enjoy!

Exit from an editor in terminal

There are many tutorials on nano, vi/vim, and emacs. Feel free to explore and share with other students. In case you are stuck in an editor, here are the commands you can exit from the program.

nano

nano is very user friendly. You can see the help at the bottom of the screen.

```
nano a.txt          Start nano to edit file a.txt
Ctrl-x              Exit from nano. See the prompt at the bottom of the screen
```

There are many tutorials on the Internet, for example, here.

emacs

Commands in emacs are control characters or prefixed a set of characters.

```
emacs a.txt         Start emacs to edit file a.txt
Ctrl-x Ctrl-s       Save file
Ctrl-x Ctrl-c       Exit from emacs
```

Here and here are examples of cheat sheets and tutorials on emacs.

vi/vim

vi has three modes. Normally, you are in the normal mode when vi just starts. Press `i` to enter the insert mode, where you can insert/type text. Press `Esc` to exit from the insert mode and get back to the normal mode. If you get lost, press `Esc` many times to get back to the normal mode.

```
vi a.txt            Start vi/vim to edit file a.txt
:wq                 Save the file and then exit from vi. Work in the normal mode
:q!                 Exit from vi without saving file. Work in the normal mode
```

Here is an example cheat sheet.