# CS 15 Lab 0: System Setup

## 0.1 Introduction

Welcome to CS 15! This lab is intended to help students acclimate to the course work-flow. Be sure to ask questions if you get stuck.

## 0.2 System Setup

This section will help you do the following things:

- Open a command line terminal
- Access the CS servers via `ssh`
- Use some basic `UNIX` commands
- Ensure that VS Code and Remote-SSH are properly installed and configured
- Download code from the CS servers via `sftp`

Before moving on to the body of the lab, make sure to look at the **CS 15 Tech Guide** on Canvas, and follow the steps there to install tools like VS Code on your machine.

### 0.2.1 Working Remotely

Remote access to the server will allow you to, create, edit, and remove files on the Halligan homework server from your local system. It will also allow you to compile and run your work in a course-approved environment. Some of the following steps should be familiar to those of you who have taken CS 11 here at Tufts, but if you haven't, or it isn't ringing a bell anyhow, please don't hesitate to ask for help.

#### Terminal Access

In order to work remotely on your own computer, you'll need a program that lets you interact with the department server. To do this, you first need access to a command-line interface.

- **Windows**: If you are properly configured, you can use either the Command Prompt or PowerShell on Windows to perform basic terminal commands. There are other alternatives, of which we recommend `MobaXTerm`, but we recommend following our tech guide and then using the in-built Windows tools.
- **Mac/GNU/Linux**: Use the native Terminal applications.

#### Making an SSH Connection

After you have access to a terminal, you'll use `ssh` (Secure SHell) to connect to the Halligan homework server. The `ssh` command establishes a connection between your local machine and (in this case) the homework server, which allows you to run commands on (your personal folder in) the homework server from your machine.

To use `ssh`, open the terminal, and type:

```
ssh utln@homework.cs.tufts.edu
```

Where *utln* is your CS department-provided *utln*. You will have received communication from `staff@eecs.tufts.edu` with instructions about accessing this account and changing its password. If you haven't received anything from them, send an email. They are usually quick to respond during working hours. Once you have a *utln*, please return here to continue.

### Running Some Remote Commands

Now that you have accessed the department server, try running some commands.[1] Enter the following one at a time to see what they do.

```
hostname
who
date
cal
cal 2022
cal 5 2022
cal 5 22
```

Can you figure out how to use the `cal` command to see what day of the week you were born?

What you have been typing are commands and arguments:

1. **command**: the name of a program built in to the remote server system, like `cal`.
2. **arguments**: extra information you can give a command to control it further. For instance, typing in the number argument `2022` after the `cal` command tells that program to print the calendar for the year 2022. If you don't include any arguments, the program will perform its default behavior (some programs will refuse to run without some arguments).

A nice feature is that on most such systems, a *manual* explaining the built-in commands is also available, via another program, called `man`. To see the manual for `cal`, enter `man cal`. You can leave the manual by typing `q`, for quit. You can also find manuals on the internet these days.

### Running Some More Remote Commands

While working on stuff in this class, most of what you will be doing will be accessing and modifying files on your account on the CS servers. It will be useful to know how to manage files and folders.

In the Linux world, a folder is called a *directory*. Whenever you are using the command line, you are always in some such directory, known as your *working directory*. If you open up your command-line tool, you will typically start in your *home directory* on your own computer; when you use `ssh` to access the remote servers, you will be in your home directory for the account you have on those machines. The below examples assume that you are doing the latter, and seeing results from the remote server. (Depending upon your system the same commands may or may not work on your own machine; they will all work if you are connected to the remote servers.)

[1] It will be super tempting to copy and paste this sort of thing. Don't! Over the course of this class, and the rest of your degree, you are going to be using the command line a lot. If you learn the commands by actually typing them in a few times, it will be much faster and easier than copy-paste. Really.

1. Type `pwd` (for "*print working directory*), and hit enter/return. This will show you where you currently are. You will see something like `/h/username`, where `username` is your CS login. This is your home directory, where you can store your own work, and the information before it is the *path* (this path would mean you are on some machine, in drive `h`).

2. Enter `ls`. This will list everything in your current directory. To start, there may be nothing to list, but as you add files and folders, you will see them by name.

3. Make a new directory for your CS 15 work, using the command `mkdir cs15`. Now use `ls` again—you should see your new directory.

4. You can move into the directory you just created using the `cd` command. This command, if you type it by itself (without arguments) does nothing; if you give it the name of the directory, it will move you into that location. Try typing `cd cs15` followed by `pwd`. You will see a longer path now; changing to this sub-directory is just like clicking into a sub-folder on in a graphical user interface (GUI), as on Windows or Mac.

5. Next, enter `cd ..` (that's a space and two dots (periods) after the `cd` commands). The two dots here are a shortcut argument that tells `cd` to move up one directory. If you use `pwd` again, you should be back in your home directory again.

6. You can rename files and folders using the `mv` command. Enter `mv cs15 cs_15`, and then use `ls` again. You should see that your folder has changed names. If you want to change the name back, you can use the same command, with the arguments reversed.

7. Next, enter `rm -r cs_15`, followed by `ls`. Your directory should now be gone, as `rm` is the *remove* command.[2]. The optional *flag* `-r` here tells `rm` that we want to remove a directory, and everything inside it—if it were a file, we would not need that.

   **IMPORTANT**: Be *very careful* with `rm`! Unlike putting something in the Trash on your home computer, this command has no temporary take-back feature. You can put something in the Trash and then take it back out, so long as you haven't emptied it, but `rm` is a one-way street—once something is removed that way it is really gone for good (or bad).

[2]If, in the previous step, you did something to change the name back from `cs_15` to something else, then this won't work; you need to use the correct name of the directory you want to remove

## 0.3 Compile and Run Your First CS 15 Program

Now it is time to access the server, get some starter code, modify it using VS Code and finally download it for your own use.

### 0.3.1 Create a Directory for Your Work on the CS Server

Connect to the CS `homework` server using `ssh` (if you're not already connected). Use the commands you have already learned to:

1. Create a directory for all your CS 15 work.
2. Change directory so that you are inside the one you just created. Create another directory for this lab, called something like `lab0`.
3. Change directory again, so that you are now inside your `lab0` sub-directory.

### 0.3.2 Copy Over Some Starter Code, then Edit, Compile, and Run

From the command line, copy over our starter code:

```
cp /comp/15m1/files/lab00/* .
```

In this command, we are copying everything (that's what the *wildcard* `*` symbol means) from the class file-path to our current directory (that's what the short-cut single dot `.` symbol means). If you run `ls` at this point, you should see that a source-file called `welcome.cpp` is now in your directory.

Now access your directory containing the code using Remote-SSH in VS Code. (This was demonstrated in the first class meeting, and is documented in the CS 15 Tech Guide on Canvas.) Open the source-file. Edit it to add your name in the two spots indicated.

Next, compile your code in the terminal, using command:

```
clang++ -Wall -Wextra welcome.cpp
```

By default, an executable file named `a.out` is created by the preceding step. At this point, you can run the file with the command `./a.out`. You should see a welcome message printed in the terminal.
Note the `./` in the preceding command. This is to indicate that the `a.out` file is in the current working directory. If you want a more meaningful name for your program, you can specify an output filename using `-o outputfilename` when compiling:

```
clang++ -Wall -Wextra -o welcome_program welcome.cpp
```

and then you can run the program using `./welcome`

### 0.3.3 Download your Work to Your Own Machine

Once you are done the previous steps, you should close your remote connection in VS Code, use the `exit` command to close your SSH connection, and go about your daily life. However, if you want to get a copy of your work so that you have it on your local machine, you will want to access it using `sftp` (*Secure File Transfer Protocol*). See the CS 15 Tech Guide on Canvas for instructions on how to do this from the command line. While it's optional for this lab (and your work will be safely stored on the CS servers in the meantime), you will need to do this in future when it's time to get a copy of what you have done to submit for grading. Probably worth practicing it now.

## Read the Course Website!

At this point, it would be a good idea to read carefully through the course website. There is lots of information there designed to help you.