CIS 200 - Lab 5

# Notes

* See the end of this document for hints and additional information (Appendix A)
* **Carefully read** all the deliverables that are to be turned in before starting the task
  + If you don’t, you may forget to obtain certain snippets and other information
* You should use one Word document for all your snippets for this Lab
  + Put your name on the first page
  + Identify snippet by task number and step (if multiple snippets per task)

# Task 1

For this task, you will use PuTTY (through WinSCP or independently – see Appendix A) to connect to **login.umd.umich.edu**, specifying your username and your password when prompted.

1. Make a directory named cis200 using mkdir
2. Move into that directory using cd
3. Use the **pwd** command to print the working directory. This should be your home directory/cis200
4. Use **ls** command with the appropriate options (a and l) to list all files and folders in long format, including hidden files.

## Task 1 Deliverables

You should capture a snippet of the output after both step 3 and step 4 and paste it into your Word document.

# Task 2

1. Create a basic “Hello World” program **using Visual Studio on Windows**, naming the source file **hello.cpp**.
2. Verify that it works (compiles and executes) through Visual Studio.
3. Then, use the WinSCP File Transfer interface (see Appendix A), and transfer (upload) the **hello.cpp** file to your Unix account.
4. Compile the source file using **g++** and using the **-o** option to name the executable hello
5. Run the executable you just created (./hello)

## Task 2 Deliverables

Capture a snippet of the Unix commands you issued to compile and run the hello executable. The output from your program should be visible as well.

# Task 3

1. Create a new file from inside of your command line using the vi command, naming it hello2.cpp
2. Write the appropriate code, using the vi editor, to print “Hello, Professor!” to the console
3. Save the file and exit vi (save using the :w command and quit using the :q command)
4. Compile the hello2.cpp file and create the executable with the name hello2
5. Execute the hello2 program

## Task 3 Deliverables

Capture a snippet of the Unix commands you issued to compile and run the hello executable. The output from your program should be visible as well.

# Task 4

1. Use the **mv** (move) command to change the name of hello2 executable to hello\_prof.
2. Execute and run this program

## Task 4 Deliverables

Capture a snippet of the Unix commands you issued to change the name of the hello2 executable. Also, the snippet should show the output from the hello2 program.

# Task 5

1. Create a subdirectory named **lab1assignment** under the cis200 directory you made earlier
2. Using the cp command, copy all the ***source files*** (.cpp) that you created (do not copy the executables)
3. Using the rm command, remove all the files from the top level of the cis200 directory, but leave the copies in tact in the lab1assignment directory

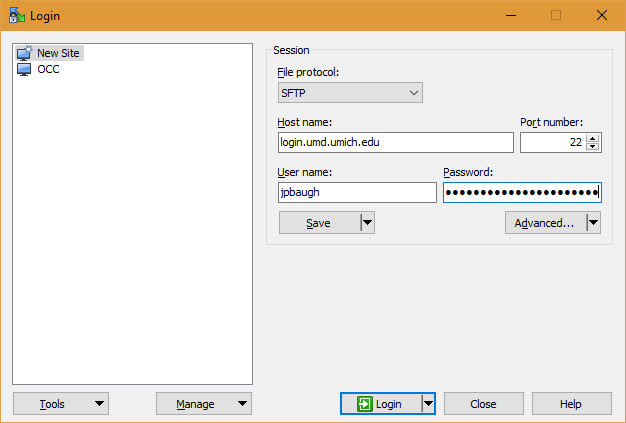
## Task 5 Deliverables

* Capture a snippet of the **cis200** directory with **pwd** and **ls** commands issued (to show you’re in the cis200 directory, and to show the contents of that directory)
  + This directory should only have the subdirectory in it (lab1assignment) and not the source and executable files (that you copied to lab1assignment, and then deleted from the cis200 directory)
* Capture a snippet of the newly created **lab1assignment** directory with **pwd** and **ls** commands issued
  + This directory should contain the source files only

# Appendix A Useful Information about WinSCP, Unix, and vim editor

## WinSCP

When you launch WinSCP, you should open the Login window, and enter the correct information for your username, and the host name, using SFTP (Secure File Transfer Protocol.)

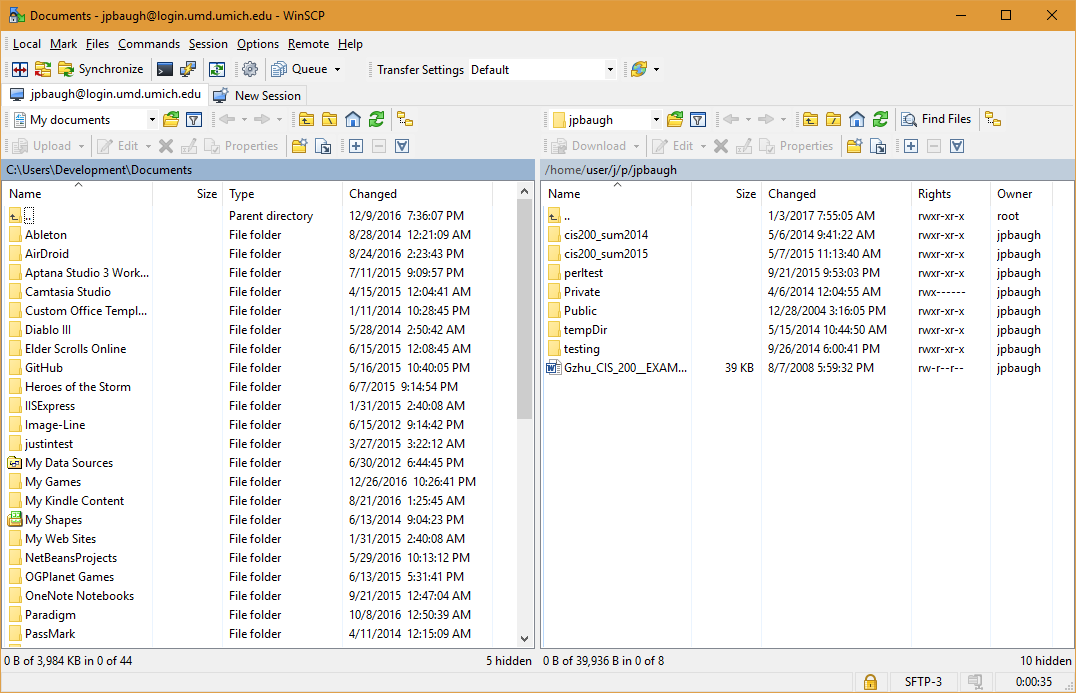


* The host name is **login.umd.umich.edu**

If you get a warning about the “host key”, hit Yes. Then, hit Continue on the next screen that is displayed.

Enter your password as prompted, and then you should see something like the following:

### The FTP Client of WinSCP



What you see initially is your **FTP client** provided by WinSCP.

The **left pane** is your **local computer’s file system**. You can navigate through your local computer with this pane.

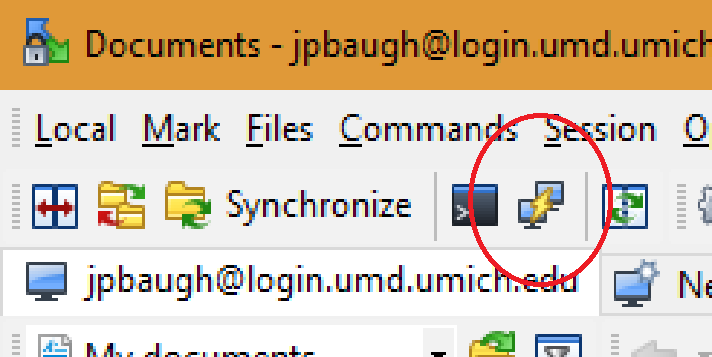
The **right pane** is the **remote computer’s file system**. The term **remote** means it is not the computer you’re in front of. If you’re at home on your own local computer and you connect to the University’s Unix system (using the steps above), then the University’s system at URL login.umd.umich.edu is the *remote computer*.

You can use these panes to download and upload files as you wish.

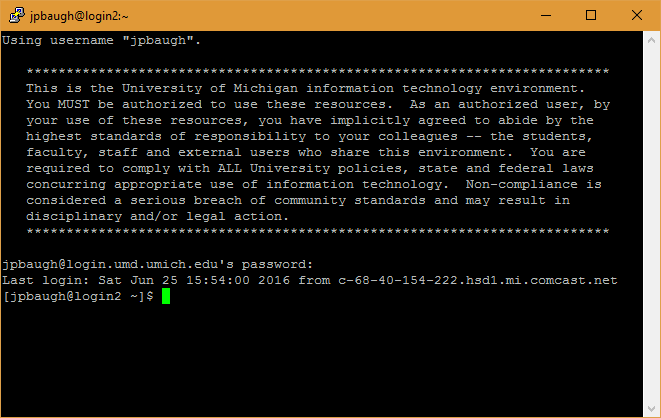
### The Terminal / Command Prompt

Another application within the WinSCP software that you might wish to use is the **PuTTY** **terminal**. This allows you to interact with a Unix/Linux system by issuing **text-based commands**. This is often called a **command-line interface (CLI.)** This is different from what most computer users interact with on a daily basis, with buttons, text boxes, radio buttons, check boxes, and other controls. The type of interface with these controls is called a **graphical user interface,** or **GUI** (pronounced as “gooey”.)

To access the terminal, you click the **PuTTY icon** in WinSCP, near the upper left, as follows:



You may be asked if you want to open up PuTTY[[1]](#footnote-1). *Confirm*, and you will see something like the following:



**Enter your password and hit Enter**. **Note: you won’t see the characters in the password as you type.**

Congratulations, you’re now logged in to the University’s Unix system. You can log in from home, campus, or across the world. All you need is WinSCP and an Internet connection.

## Unix Commands

The following are some useful Unix Commands.

|  |  |  |
| --- | --- | --- |
| Command | Description | Examples |
| echo param | Used to echo the parameter. If an environment variable, the contents will be displayed | echo $SHELL |
| mkdir dirName | Used to make a directory (folder) in which you can store files | mkdir lab1\_dir |
| cd dir | Used to change directory | cd lab1\_dir cd .. |
| pwd | Used to print the path of the current directory | pwd |
| ls | Used to print the contents of the current directory. Some Options:   * ls lists all non hidden * ls -a lists all * ls –l lists in long format * ls -al lists all in long format | ls -a  ls  ls -l  ls -al |
| cat filename | Used to display the contents of a text file without opening it | cat someFile.cpp |
| mv file1 file2 | Used to move *and/or* rename a file | mv myFile myOther mv myFile direct/ |
| cp file fileCopy | Used to copy a file | cp myFile myFileBackup |
| rm filename | Removes a file | rm someFile |
| rmdir dirName | Removes a directory (folder) | rmdir myDirectory |
| vi filename | Opens vi (or vi improved, known as vim) and creates a file | vi test.cpp |
| g++ filename.cpp | Compiles a C++ source file. If no output flag and name are specified, the executable is generated with the name **a.out** If you use the -o flag with a name, the executable will be generated with that name | g++ test.cpp g++ test.cpp -o test |

## vi/vim Editor

vi/vim is a text-based editor that is installed on all Unix/Linux systems. It operates in two primary modes: **Insert mode**, in which you type text, and **Command mode** in which you can issue commands to the software itself.

Since it is terminal/CLI based, there are no buttons, text boxes, menus, or other controls. Therefore, it is necessary to have some way of distinguishing between issuing commands, and typing text in the document you are creating. This two-mode technique is the choice that the developers of vi/vim made to establish this distinction.

You can use vi/vim to create any text file you’d like. This could be just simple note files, to entire C++ or Java programs, or anything you’d like.

**When you first open up vi/vim**, you are in **command mode**. This mode allows you to navigate through the file and also to issue commands. In this mode, all keystrokes are considered commands, NOT text to be added to the file.

Some useful commands:

**a** append right after current position

**[Shift]+a**  append to end of line

**:w** Write file (save)

**:q** Quit vi

**:q!** Force quit (quit without saving)

**i Enter insert mode**

To enter insert mode from command mode, just hit the **i key.**

To return to command mode, hit the **Escape key.**

1. If you get an error when you try to open up PuTTY from inside WinSCP, then you may not have it installed, or at least not have it installed in the location WinSCP expects. You can install PuTTY separately as a stand-alone application, or install it to the directory that WinSCP expects, and then you’ll be able to access it from inside WinSCP. [↑](#footnote-ref-1)