CS163 Lab Session #1 – Getting Started

Background Information

Goals:

Get prepared to work in the unix environment for the programming assignments this term:

1. Demonstration of using odin from a PC and from a Mac
2. Be prepared to get a CS account (required)
3. Learn how to download putty or ssh for PCs and use terminal for Macs
4. Experience logging in, entering a program, compiling and running

Overview

1. **Everyone will need to go to FAB 88 to receive a CS account.**
    a. Once you receive an account, login to Solaris **using ssh or putty** (**cs.pdx.edu**)

2. **Until then, login to Odin using ssh or putty (odin.pdx.edu).** If you don't have an odin account or password you will need to get one. If you don't have one go to: **www.account.pdx.**edu and for help go to: **www.oit.pdx.edu**
    a. Start up the program called ssh – you should be able to find a free site from which to download it. It is a secure shell program and allows your computer to act as a 'dumb terminal' for logging in remotely. You start up ssh by double clicking on it.
    b. Then, connect to: odin.pdx.edu
    c. Use your odin login name and password (the same login and password that you use for D2L)
    d. **When you login into Odin, hit return when prompted. Then hit 4 <enter**> to exit the menu.

3. If you are using a **Mac**, open up the terminal program.
    a. Open Finder and navigate to Applications.
    b. In Applications, navigate to Utilities.
    c. Find Terminal and double-click it.
    d. Once you have a Terminal open, simply type:

        ssh <username>@<hostname>.cat.pdx.edu

4. The very first time you login to do your Computer Science homework, you should create a directory where your programs will live (and where you will work from). To make a new directory type:

        mkdir cs163 <enter>

5. **From then on, you will need to travel to that directory to actually get your work done. This will need to happen each time you login. So,** to enter that directory to start to work type:
   cd cs163 <enter>

6. Now you are ready to start typing in your program. **Use pico or nano** to type in a program. These are editors. If you are working on a Mac, you will want to use nano instead! (In this course you may use any of the following editors: pico, nano, vi, vim, or emacs). So, type at the unix prompt:

   pico prog1.cpp <enter>

7. All C++ programs should have a **.cpp extension**.
8. **Enter in the program**; with pico, you can use the error keys and delete to fix mistakes and move the cursor around. There is a menu that appears at the bottom of the pico screen.

9. **When done hit control o at the same time to write it out and control x to exit**.

10. **Compile** your C++ source code file. The command to do this is:

    g++ prog1.cpp<enter>

11. If your program successfully compiles, it will produce a file named 'a.out' in your directory. Otherwise, you will need to correct syntax errors before continuing - by using pico again:
    pico prog1.cpp<enter>

12. **Run** your program by typing:
    ./a.out

13. **Once completed, you will want to practice uploading your file to D2L to submit.** It is a good idea to also email a backup copy of your file to karlafgr@cs.pdx.edu **(notice the email address is different than your teacher's email address…)**
    a. When submitting a file to D2L, you will first need to transfer your .cpp file to your computer (from unix).
    b. To do this, you will need to use a file transfer program. There are many choices!
    c. On a PC, I use **WINSCP** which is a free program that allows you to drag and drop files from unix to your PC
    d. On a Mac, a free drag and drop program is **Cyberduck**, but there are many other possible programs you can use.
    e. When setting up the software, make sure you have asked for "Secure" file transfer (**SFTP**).

14. **Once you have transferred your .cpp file to your own computer, u**pload it to D2L's dropbox.
    a. Make sure to hit the **Submit** button after uploading, otherwise your file will not actually be stored in D2L

Common Unix commands:

Quick hint before starting – avoid names with blanks! Names with spaces will need double quotes surrounding the entire name.

1. **mkdir** – to organize files and directories. This will make a new directory: **mkdir cs163**
2. **rmdir** – to remove a directory (which must be empty): **rmdir cs163**
3. **cat** – to display the contents of a file: **cat test1.cpp**
4. **more** - outputs one page of a file and pauses. You can also pipe (|) to be used with other commands
    a. **ls -l cs163 | more**
    b. **cat test1.cpp | more**
5. **man** - show manual for a command
    a. **man g++**
    b. hit q to exit the man page.
6. **cd** - change directory
    a. To cd into the cs163 directory: **cd cs163**
    b. To cd back to the directory above: **cd ..**
7. **pwd** – to determine your current path and directory name : **pwd**
8. **ls** - list directory contents
    a. ls cs163
    b. use ls -l /etc to see more detail
    c. use the wildcard (*) if you want to list all of the .cpp files in a directory: ls *.cpp
9. **cp** - copy a file or directory, example: cp source dest if you want to copy a directory use the -R option for recursive: cp -R /source /dest
10. **mv** - move a file, example: mv source dest
11. **rm** - remove a file, example: rm test.cpp
12. **rmdir** – to remove a directory (which must be empty): rmdir cs163
13. **grep** - pattern matcher, grep takes a regular expression, or to match a simple string you can use fast grep, fgrep failure /var/log/messages
14. **tail** - prints the last few lines of a file tail -2 test1.cpp
15. **head** - same as tail, but shows the first few lines the file head -5 test1.cpp
16. **vi** - text editor, there are several text editors such as emacs, and nano, but vi is usually installed on any server so it is good to learn. To edit a file type vi file to edit a line press Esc i then to save changes and exit use Esc wq, or to quit without saving use Esc q!.

CS163 Lab #1 - Getting Started

Please complete this to be familiar with the systems used. You can get assistance through tutors@cs.pdx.edu
There is nothing with this lab that needs to be submitted.

_____1. **Power-up your computer**

_____2. **Connect to PSU's network (PSU or PSU Secure) using your ODIN login and password**

_____3. **PC Only:**

a. **Install ssh or putty**
b. **Double click on the ssh or putty icon**
c. **For putty,** experiment with different colors and text sizes (Window – Appearance)
d. **Use Host name of cs.pdx.edu** (if you have a CS account) or **odin.pdx.edu**
e. **Open a session and login**
f. **With Odin, hit return when prompted. Then hit 4 <enter>** to exit the menu.

_____4. **MAC Only:**

a. Open Finder and navigate to Applications.
b. In Applications, navigate to Utilities.
c. Find Terminal and double-click it.
d. Once you have a Terminal open, simply type:

ssh <username>@cs.pdx.edu or ssh <username>@odin.pdx.edu

_____5. Create a directory where your programs will live (and where you will work from.
mkdir cs163 <enter>

_____6. **Changed into** that directory to start to work type:
cd cs162 <enter>

_____7. Now start typing in a program to add nodes to the end of a linear linked list of integers: (nano should be used on Macs)
pico prog1.cpp <enter> or nano prog1.cpp <enter>

_____8. **When done hit control o at the same time to write it out and control x to exit**.

_____9. **Compile** your C++ source code file. The command to do this is:
g++ prog1.cpp<enter>

_____10. **Run** your program by typing:
./a.out

_____*11.* **Once completed, you will want to practice uploading your file to D2L to submit.**

CS163 Lab #1 - Practicing

Practice these questions on your own or whenever you have extra time at the labs (today or in the future).
You will be expected to be Proficient at these types of questions by midterm time.
Please contact us if you need help with any of these questions (tutors@cs.pdx.edu, karlaf@cs.pdx.edu)

First do these questions iteratively:

_____*1.* **Write the code to display the contents of a linear linked list**

_____*2.* **Write the code to remove the last item from a linear linked list**

_____*3.* **Write the code to remove all nodes from a linear linked list**

_____4. **Remove all items (there is no "node" destructor):**

_____*5.* **Display every other item, starting with the first**

_____*6.* **Check to find the requested data (sent in as an argument) is in the linear linked list**

Now re-write them recursively!

_____*7.* **Write the code to display the contents of a linear linked list**

_____*8.* **Write the code to remove the last item from a linear linked list**

_____*9.* **Write the code to remove all nodes from a linear linked list**

_____10. **Remove all items (there is no "node" destructor):**

_____*11.* **Display every other item, starting with the first**

_____*12.* **Check to find the requested data (sent in as an argument) is in the linear linked list**