LAB #1 – Introduction/Logon

1. **(2 pts) Icebreaker:** Believe it or not, we do have to communicate with others being a computer scientist and electrical engineer☺ Let's get to know our peers. Get into small groups of 4-5, and answer the following about everyone in the group:
    - What is your major?
        - If you are cs, why do you want to be a computer scientist?
        - If you are ece or a different major, why do you feel computer science is relevant to your discipline?
    - Do you have any experience programming in C/C++? If so, how long?
    - What did you like (or dislike) the most about your Winter Break 2014?
    - "BE PROACTIVE" is one of our class mottos this quarter. Being proactive, establish one goal for yourself this Winter quarter.

2. In this course, all our labs involve paired programming. You do not have to keep the same partner for each lab, but you MUST work with someone in each lab. For each lab, you will follow the paired programming model, as specified in the student handout.

3. At this time, you need to split up in the classroom based on your answer to the second question above. If you have had at least 15 weeks of a programming class, then put yourself on one side of the room. If you have not, then put yourself on the other. Now, pair with someone from separate sides of the classroom, and finish the rest of the lab as a pair.

    Your TA will go over what the "driver" and "navigator" is in the paired programming model before proceeding with the rest of the lab.

(2 pts) Simple Unix/Forward engr email:
For this lab, the **exercises on the computer need to be repeated by each student** in the pair. This is to ensure that both students get their computers set up properly for the rest of the course!!!

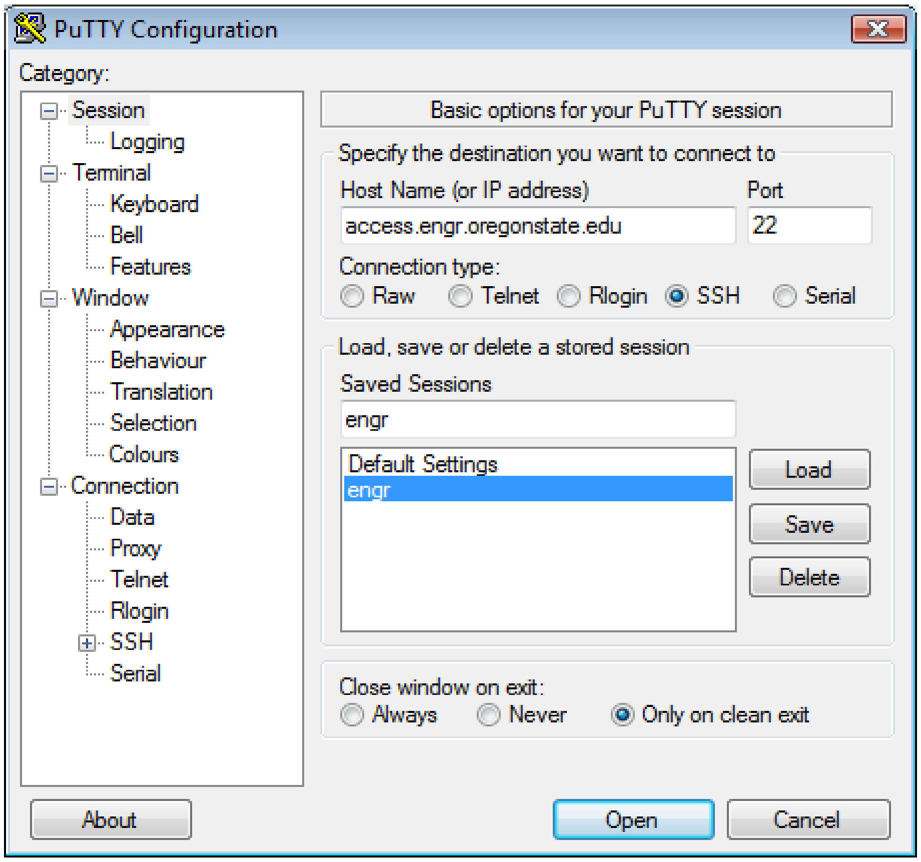
4. **Windows:**
    Download a free Secure Shell (ssh) from OSU (preferably **PuTTY**):
    http://oregonstate.edu/helpdocs/software/recommended-software/osuware
    Once you download putty, the file is an .exe that can be opened without being installed.

    Macintosh/Linux (skip step 6):
    If you are using a Mac or Linux laptop, you do not need to download Putty!!!
    You probably have a terminal w/ ssh built into the OS. Open the terminal, and type **ssh username@access.engr.oregonstate.edu** at the prompt.

5. First, **if you are not an ENGINEERING student, then you need to go to the TEACH website and create a new account:** https://secure.engr.oregonstate.edu:8000/teach.php?type=want_auth

6. Once you have an engineering account,
open PuTTY, the ssh client, and type the following address in the hostname:
access.engr.oregonstate.edu

   I suggest you load the engr host name under a saved session by typing something in the **Saved Sessions** text area and pressing the **Load** button. This makes it easier to save specific properties to the session such as font, background color, text color, etc. Then, you can simply select the saved session each time before pressing Open.



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): access.engr.oregonstate.edu

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

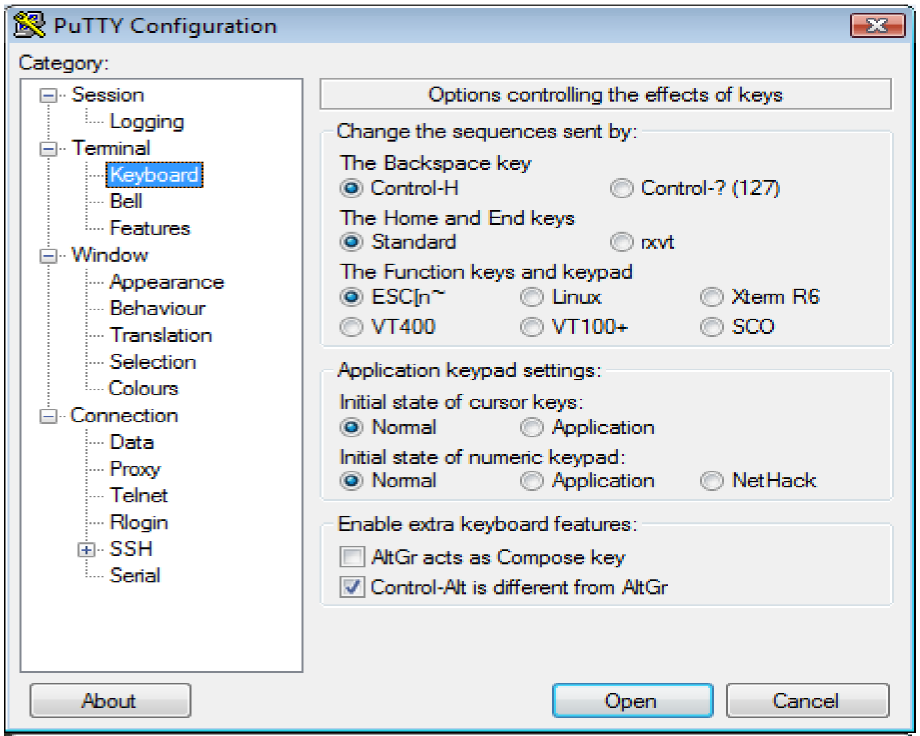
Saved Sessions: engr

Default Settings
engr

Close window on exit: Always Never [✓ Only on clean exit]

Two properties you may want to change immediately in the ENGR session is **The Backspace key** located under the **Terminal** -> **Keyboard** tab. This will eliminate those annoying ^? when pressing the backspace key in the vim text editor.

You can also eliminate the **Access denied** message that pops in Putty up after entering your user name when logging into the system by removing the check mark in the **Attempt GSSAPI authentication** box under the **Connection** -> **SSH** -> **Auth** -> **GSSAPI** tab (See first image on next page).



Options controlling the effects of keys

Change the sequences sent by:

The Backspace key
[✓ Control-H] Control-? (127)

The Home and End keys
[✓ Standard] rxvt

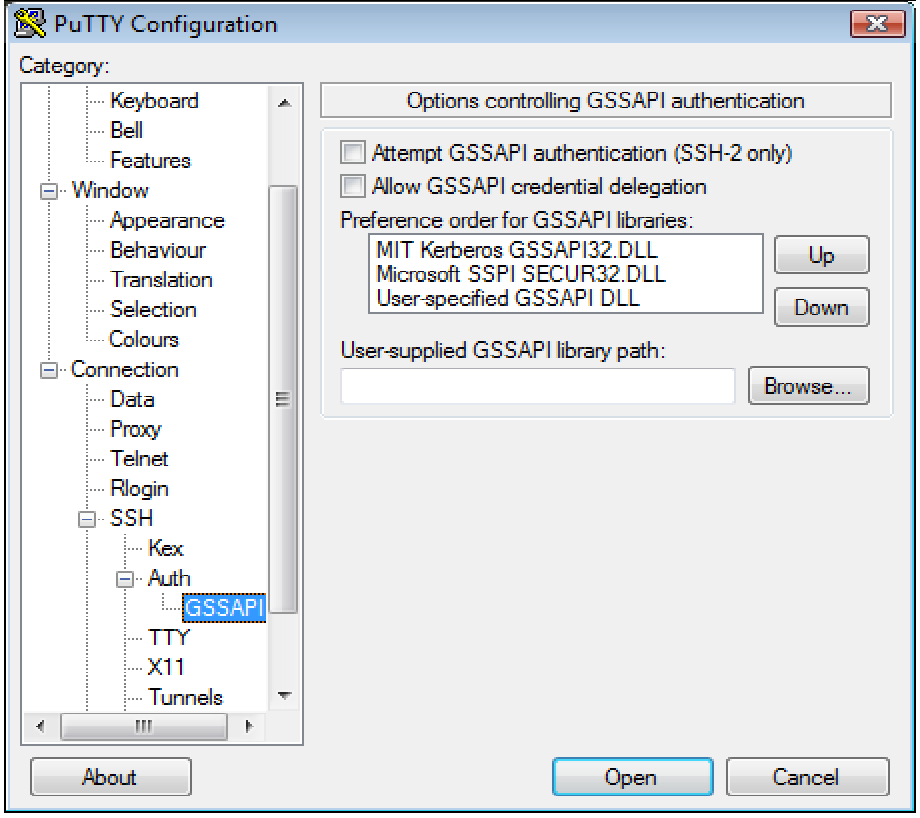
The Function keys and keypad
[✓ ESC[n~] Linux Xterm R6
VT400 VT100+ SCO

Application keypad settings:

Initial state of cursor keys:
[✓ Normal] Application

Initial state of numeric keypad:
[✓ Normal] Application NetHack

Enable extra keyboard features:
- ☐ AltGr acts as Compose key
- ☑ Control-Alt is different from AltGr



Options controlling GSSAPI authentication

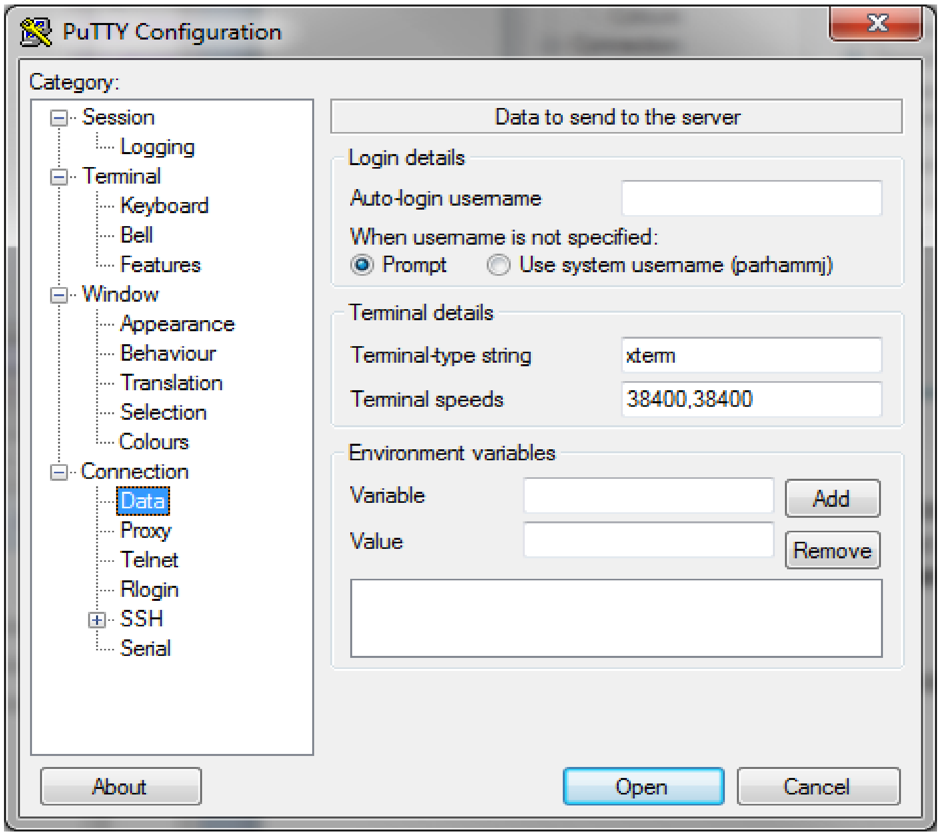
- ☐ Attempt GSSAPI authentication (SSH-2 only)
- ☐ Allow GSSAPI credential delegation

Preference order for GSSAPI libraries:
MIT Kerberos GSSAPI32.DLL
Microsoft SSPI SECUR32.DLL
User-specified GSSAPI DLL

User-supplied GSSAPI library path:

Another property you might want to check is the terminal type you are using. Go to the **Connection -> Data** tab, and type **xterm** in the **Terminal-type string** text area

(See second image on next page). This terminal type ensures that color coding is used for your programs in the vim text editor. Don't worry if you don't understand what this means right now, you will later☺



Auto-login username
When username is not specified:
[✓ Prompt] Use system username (parhammj)

Terminal details
Terminal-type string: xterm
Terminal speeds: 38400,38400

Make sure you go back to the **Session** tab and press the **Save** button before opening the ENGR session. This will save these properties and ensure that you do not have to do ALL this again☺

7. Now, open the ENGR session, and enter your username and password at the prompt. ****Note**: You will not see anything as you type your password. This is a security feature in Linux!!!
   After a successful logon, you should get a prompt like the one below:
   flip1 ~ %

   If you do not have an ENGR username and password, then your TA will help you apply for one.

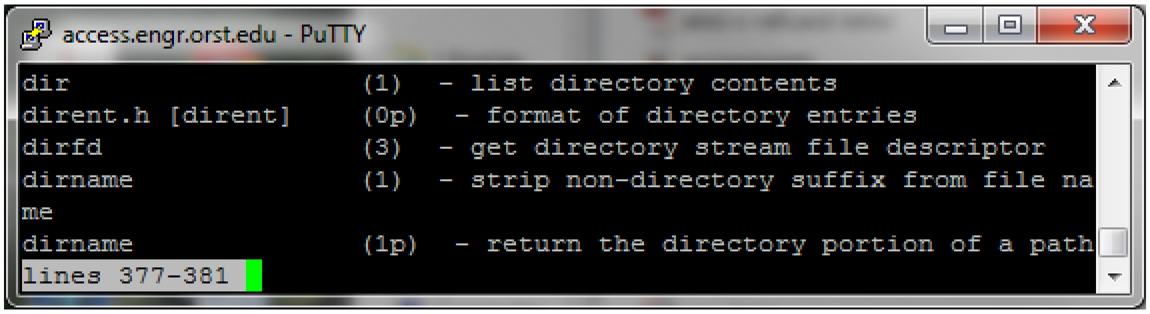
8. At the prompt, type the following commands to look at your files and directories in your home directory. Note the differences to your lab instructor.
   ls
   ls -a
   ls -l
   ls –al

Note: You should notice a **.** and **..** directory listed. The **.** directory refers to your current directory, and the **..** directory refers to one directory above your current directory. The **~** refers to your home directory.

9. UNIX/Linux provides you manual pages for the commands. You are encouraged to read these manual pages when you have questions about a specific command or want more details about the options to use with a command. Use the **space bar** to scroll forward through the manual pages (one page at a time), **press b** to scroll backwards (one page at a time), and **press q** to quit the manual page. You can also use the up and down arrow keys to scroll forward and backward one line at a time, but who wants to do this☺
   man ls

10. In addition, if you are not sure what a command is in UNIX/Linux, then you can find the appropriate command using apropos and a keyword. For example, what are the UNIX/Linux commands for editing a file, working with a directory, etc.
    apropos editor
    apropos directory

11. You may have noticed that you get more text than what can fit in your terminal window. To view data a page at a time in your terminal, you can pipe the command contents through another command called less. This will allow you to scroll through the pages using **space bar, b, and q** just as you did with the manual pages.
    apropos directory | less



```
dir                     (1)   - list directory contents
dirent.h [dirent]       (0p)   - format of directory entries
dirfd                   (3)   - get directory stream file descriptor
dirname                 (1)   - strip non-directory suffix from file na
me
dirname                 (1p)   - return the directory portion of a path
lines 377-381
```

12. The files preceded by a period/dot are system files, which are hidden with a normal directory listing using ls. Let's **make sure you have your engr email messages forwarded** to an account that you check often. You have several options for doing this.

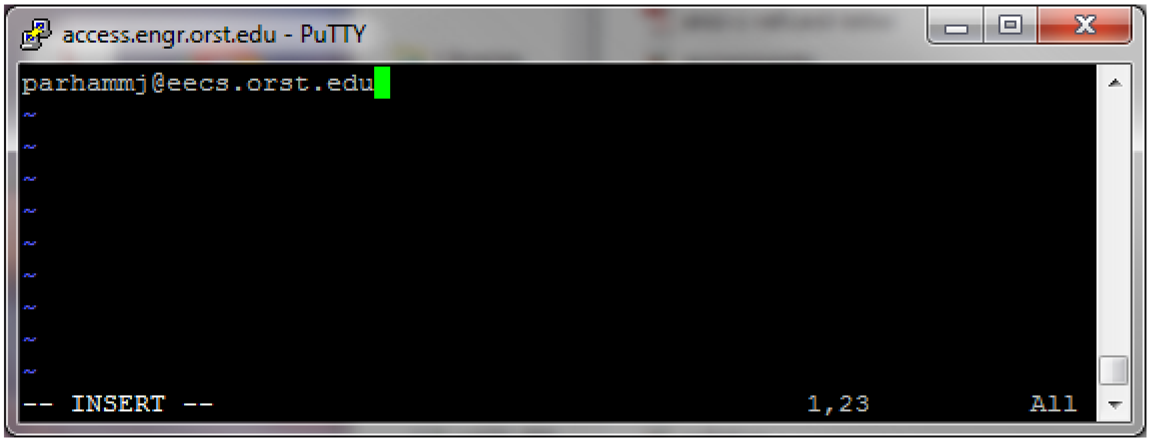
    You can implicitly modify the Linux system file by going through TEACH:
    https://secure.engr.oregonstate.edu:8000/teach.php?type=want_auth

    OR

You can explicitly modify/create a **.forward** system file. This system file is used to forward your email to a different location, e.g. **username@onid.orst.edu**. In order to create this file, we are going to use the vi/vim text editor.
vim ~/.forward

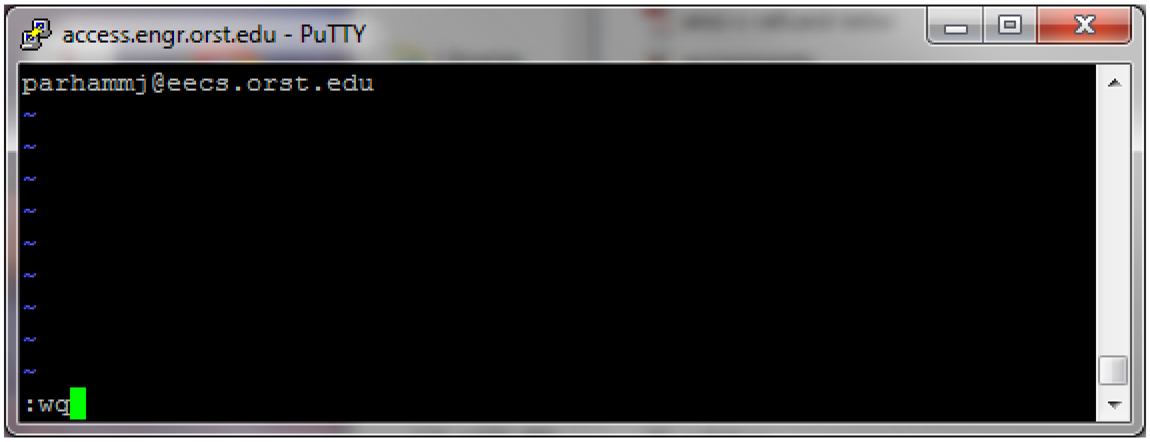
When in vim, press **i** to put yourself in insert mode where you can type text, and you will notice the word INSERT appear at the bottom of the editor.

Now, type the email address to use when forwarding you engr email. For example, I forward my engr email to my eecs email.



```
parhammj@eecs.orst.edu
~
~
~
~
~
~
~
~
~
-- INSERT --                                 1,23          All
```

To save the text you have typed, you need to press **Esc** to go back into command mode, and now you can type **:wq** to write/save the text to the file and quit (press enter after the command). You can also type **:w** and enter to write the file without quitting out of vim or **:q** and enter to quit without saving.



```
parhammj@eecs.orst.edu
~
~
~
~
~
~
~
~
~
:wq
```

Below is an overview of vi/vim and a set of useful commands. **Remember you can **man vim** to see the manual pages for vi/vim. In addition, I have provided a useful vim tutorial under **Useful Links on our class website**. One of the default editors that come with the UNIX operating system is called vi (**vi**sual editor) or vim (**v**isual editor **im**proved). [Alternate editors for UNIX include emacs and pico.]

The UNIX vi editor is a full screen editor and has two modes of operation:

- *Command mode* commands which cause action to be taken on the file, and
- *Insert mode* in which entered text is inserted into the file.

In the insert mode, every character typed is added to the text in the file; pressing the <Esc> (*Escape*) key turns off the Insert mode.

Basic Commands (in command mode):

:w<Return> - write out modified file to file named in original invocation
:wq<Return> - quit vi, writing out modified file to file named in original invocation
:q<Return> - quit (or exit) vi
:q!<Return> - quit vi without saving the latest changes
:0<Return> - move cursor to first line in file
:n<Return> - move cursor to line n
:$<Return> - move cursor to last line in file
:set number<Return> - insert line numbers into vi file

/search<Return> - find first occurrence of search from the location of cursor in the downward direction
?search<Return> – find first occurrence of search from the location of cursor in the upward direction
n – move cursor to next occurrence of last search (**in direction of search**)

j [or down-arrow] - move cursor down one line
k [or up-arrow] move cursor up one line
h [or left-arrow] move cursor left one character
l [or right-arrow] move cursor right one character
0 (zero) - move cursor to start of current line (the one with the cursor)
$ - move cursor to end of current line
^ - move cursor to the first non-whitespace character in a line
w - move cursor to beginning of next word
b - move cursor back to beginning of preceding word
u – undo whatever you last did
x – delete current character
dd – delete current line
yy – yank line and put into buffer for pasting
p – paste text in buffer to line below cursor
i – enter insert mode and enter text before the cursor
a - enter insert mode and append text after cursor
o - enter insert mode and enter text on line below cursor
<Esc> - get out of insert mode and enter command mode

(2 pts) Create Directories, Write C++ Program, & Answer Metacognitive Questions

13. After you and your partner create your .forward file, make a directory in your home directory named labs, and change into the labs directory.

mkdir labs
cd labs

14. Create a directory in your labs directory named lab1, and change into the lab1 directory.
    mkdir lab1
    cd lab1

15. Make a note of your present working directory.
    pwd

16. You can go back/up a directory by using two periods/dots together, and you can go back to your home directory by using the tilde, ~. Use the pwd to confirm you are back in your home directory.
    cd ..
    cd ~
    pwd

17. Now, change into the labs/lab1directory by using your **up arrow key** to take you through the history of commands you've used in the past. You should see the **cd labs** and **cd lab1** command you typed earlier. You can also change directly into the lab1 directory by using **cd labs/lab1**.

18. A good rule of thumb is to use **pwd** at any time to determine where you are, in case you forget☺ Also, don't be scared to use **ls** as often as you need to see a listing of your current directory!

19. Use the vim editor to create a C++ file containing your first C++ program.
    vim hello.cpp

20. Write the infamous "hello world" program as your first piece of C++ code. Use the style guideline on the class website for suggestions on how to format your code:
    http://classes.engr.oregonstate.edu/eecs/winter2015/cs161-001/161_style_guideline.pdf

```
#include <iostream>

int main() {

   std::cout << "Hello World!" << std::endl;

   return 0;
}
```

21. Compile and execute your C++ "hello world" program.
    g++ hello.cpp –o hello
    ./hello

21. Logout of the remote machine.
    exit (or logout)

22. At this point, you and your partner need to address these metacognitive questions by either typing or handwriting your answers:

    Planning:
    Why do you think we are learning Unix and C++?
    What resources did you use to complete this lab?
    Did you use your time wisely to finish the lab on time?
    Monitoring:
    What was the most difficult and confusing part of this lab?
    What did you do to address the difficulties and confusions?
    Evaluation:
    How well do you think you accomplished the lab?
    If you had to do this over, what would you have done differently?
    What worked well for you and what didn't?

(2 pts) Write Steps to Solve Problem:

23. As a pair, design/write out the steps on a piece of paper that you need to go through to solve the following problem from your book.

    Problem Statement: A metric ton is 35,273.92 ounces. Write the steps needed to get the weight of a package of breakfast cereal in ounces from the user and output the weight in metric tons, as well as the number of boxes needed, to yield one metric ton of cereal.

    You and your partner write down how long you think you will live and how many boxes of cereal you each per week. Determine if you will ever eat a ton of cereal in your lifetime☺

(2 pts) Upload Program (.cpp) and Questions (.pdf) to TEACH:

24. Since you have to get your assignments off the engr server and onto the TEACH website, then we should practice now☺ You have two options for this. You can transfer the file to your own computer, then upload it on TEACH (look into a free sftp client). Or, you can map a network drive, and choose the file from the mapped network drive. Here is some software to help you with this.

Lastly, you want to map a network drive to the ENGR server. This allows you to directly work off the server as if it were a disk drive on your computer. You can follow these instructions to map a network drive for Windows or MacOS.

Windows:
http://engineering.oregonstate.edu/computing/fileaccess/windows_file_sharing/#map_network_drive
MacOS:
http://engineering.oregonstate.edu/computing/fileaccess/smb/

If you want to use the drive off campus, then you must download the **Cisco VPN Client** from OSU: http://oregonstate.edu/helpdocs/network/vpn-campus-access

**NOTE: You may want to put these programs on a flash drive to carry with you in your backpack. This will help you get around from any computer without needing your laptop all the time.