CSE2421 PRE-LAB PRACTICE PROBLEM

Getting to know LINUX and your choice of editor

Due: Wednesday, August 31 by 11:30 p.m.

The correct submission of this pre-lab practice problem is considered an in-class assignment.

UNIX versus LINUX

http://www.diffen.com/difference/Linux_vs_Unix

Most common difference: UNIX is propriety system while Linux is an Open Source system.

UNIX/LINUX – What is it?

- Multi-user and multi-tasking operating system.
- Multi-tasking: Multiple processes can run concurrently.
- Example: different users can read mails, copy files, and print all at once (although actually, only one process can be running at any given time, if the processor has a single core).

Logging on using your CSE username and password: If you are already a major and you have a permanent account, be sure to use your already set/used passwords i.e. do not use the default password to logon. If you are logging on for the first time and/or your cse account is not a permanent one (which means you have a new account created each term), you will need to use the default password for your cse account; then, you should be prompted to change your password, keeping in mind that your password is case-sensitive. Also, the cursor doesn't necessarily move when you type in your new password. From the CSE labs, you will need to logon to the Windows environment first, which should then have an stdlinux icon on the desktop for you to choose (if it doesn't, change computers). Choose the stdlinux icon to logon to the LINUX environment using the LINUX password, remembering that LINUX, too, is case-sensitive. Your Windows and LINUX passwords should be the same, and are also identical to your OSU name.n account.

COMPUTING SERVICES

If you ever have questions about or problems with your CSE account, email help@cse.ohio-state.edu or visit the SOC lab in DL895. The hours of operation for the SOC lab and Help Desk are here:

http://www.cse.ohio-state.edu/cs/gethelp/helpdesk.shtml

FYI: If you want to create a personal web space, the directions are here:

http://www.cse.ohio-state.edu/cs/resources/web.shtml

COMPUTING LABS

https://cse.osu.edu/computing-services/computing-labs

THE LINUX ENVIRONMENT – stdlinux

You can log on to stdlinux remotely, as explained in class, using the remote access method appropriate for your system, or you can go to a CSE lab, as mentioned above. The way to access stdlinux from a lab is explained below.

After you have logged on to Windows and then Linux, as explained above, to get a terminal window (if you are in graphical mode), from a computer in a lab, choose the "Applications" tab (top left menu system), then from the drop down menus, choose "Other" then "Terminal". Notice the picture of the terminal. This icon may also be to the right of your 3 tab main menu of "Applications", "Places" and "System".

A window pops up with your "home directory" name. For example:

/home/3/greeng

%

The % sign is the LINUX prompt. You are ready to enter LINUX commands. Thus, this is called the "command line prompt".

To change your password, you have to change your OSU name.n account password.

Below are some other good/common commands for you to learn. The first set of characters up until the first space is the command name, the rest is either a description of what should go next (a command, a directory, an option, etc) or an actual set of characters to type in. You can always use the **man** command to look up more information for a given command.

The **man** command: **man**ual pages are on-line manuals which give information about most commands

- Tells you which options a particular command can take
- How each option modifies the behavior of the command
- Type **man *command*** at the command line prompt to read the manual for a command
  - Try to find out about the make directory command by typing: **man mkdir**

- Press the <enter> key to scroll through the manual information line by line
- Press the space bar to scroll through the manual information page by page
- Press the letter q to quit the manual information and return to the LINUX prompt

COMMAND (^ is control)	MEANING
*	match any number of characters
?	match one character
whatis command	brief description of a command
chmod [options] file	change access rights for named file
cd	change to home-directory
cd ~	change to home-directory
cd directory	change to named directory
cd ..	change to parent directory
cp file1 file2	copy file1 and call it file2
wc file	count number of lines/words/characters in file
cat file	display a file
tail file	display the last few lines of a file
ls -lag	list access rights for all files
ls -a	list all files and directories
ls	list files and directories
who	list users currently logged in
mkdir	make a directory
mv file1 file2	move or rename file1 to file2
man *command*	read the online manual page for a command
rmdir directory	remove a directory
rm file	remove a file
^C	kill the job running in the foreground
^Z	suspend the job running in the foreground
bg	background the suspended job
jobs	list current jobs
kill %1	kill job number 1
kill 26152	kill process number 26152
ps	list current processes

To create a new directory (i.e. folder) for your C programs called cse2421, type the following command and hit enter:

```
% mkdir  cse2421
```

To change to that new directory created above, type the following command and hit enter:

```
% cd  cse2421
```

Notice that the path name above the LINUX prompt has changed. Create

a new directory called *prelab:* **mkdir prelab**

Type in the following command to go to the prelab directory:

```
% cd prelab
```

THE FIRST ENCOUNTER OF A C KIND

Return to your home directory:

```
% cd ~
```

Notice that there must be a space before the ~ character.

You can use whatever editor you would like, but the following directions apply to the Xemacs editor. Jump to the LINUX EDITORS section at the end of this assignment description for information about other editors; and return to below where you are asked to type in the program.

At the command line prompt (the % symbol is the command line prompt), type in:

```
% xemacs  hello.c  &
```

The & symbol allows another window to open so that you have a command prompt in one window and your program file in another. This way, you can click on one window to edit and save the program, and click on the other window to execute LINUX commands (like compile).

If using the emacs editor for the first time, the bottom of the emacs window says:

```
Migrate init file~/.xemacs/? (yes or no) → answer: yes
Create compatibility .emacs? (yes or no) → answer: yes
```

At the Linux command line prompt, type the **which** command below, to verify that this editor has been set up correctly:

```
% which emacs
emacs:   aliased to xemacs
% which xemacs
/usr/local/bin/xemacs
```

FYI: The second time you log in to this environment (the xemacs editor), you will see the same thing at the bottom of the window (as described above). Please answer yes to both questions again.

Type in the following program exactly as you see it here (except be sure to insert your name as the author):

```
#include  <stdio.h>

/*  Author:  put your name here */

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

NOTES: The quotes are double quotes, not two single quotes side by side. **Be sure to hit the enter key after the last curly bracket.** The indentation is important for readability and is a standard that you should always incorporate when writing your programs – use the tab key to indent.

How are spaces (tabs, new lines/returns handled??? They are ignored by the compiler for the most part.

Click the SAVE icon on the secondary menu. Click on the File tab above the secondary menu. A drop down menu appears. Notice there is a “Save As…” option here. Also notice that there are keyboard shortcuts for some of the menu options to the right of each option. Click on the “Exit XEmacs” option at the very bottom of the File tab drop down menu to exit the program. The program window will be exited as well.

Use the up arrow key to scroll through previous commands (and the down arrow key to scroll in the opposite direction) to find the **emacs hello.c &** command then hit the enter key. This will bring your program back up in another terminal window. If you want/need to change the command a little, use the left and right arrow keys to move the cursor through the command characters.

To compile the program, type the following command at the LINUX prompt then hit the enter key:

```
% gcc  -ansi -pedantic -o  hello  hello.c
```

If the compiler gives errors, you made one or more typos in entering the code above, so go back and correct them, and then resave the edited hello.c file. Then, recompile. Once the program compiles with no errors, you have an executable which is produced by the compiler, and you can run the program, as explained below.

To run the program, type the following command at the LINUX prompt then hit the enter key:

```
% hello
```

You should see the output *Hello, World!* on the screen.

To find a list of all of your C programs in the current directory, type the following command and hit enter:

```
% ls *.c
```

Remember, this is where you originally stored your hello.c program.

Copy the **hello.c** file to the *prelab* directory/folder:

```
% cp  hello.c  cse2421/prelab
```

Notice that cse2421/prelab portion of the command is the path added to the current (home directory) path to get the file copied to its new location.

Check that the file copied correctly into the specified directory:

```
% cd cse2421/prelab

% ls
```

The ls command lists all the information stored in the prelab directory. If you want more information, try:

```
% ls  -l
```

That is a lower case letter L where the minus sign in front of it specifies that this is an option for the ls command. Check the manual pages for more information about the ls command by typing: **man ls**

Do you have any files in the cse2421 directory? Change to your cse2421 directory by typing:

```
% cd ..
```

There must be a space between cd and ..

This command takes you “up” one directory; notice the path no longer lists the prelab directory.

Now list the information in this directory: % **ls**

Now type in the command for more information: % **ls -l**

There should be only the prelab directory here.

Now, let's go back to your home directory. Whenever you want to go to your home directory, using any of the following commands:

% cd ~

% cd

[Note: In this case, you could also use:

% cd ..

This is because your home directory is the parent directory of your cse2421 directory.]

Now list what's in this directory: % **ls -l**

If you read about the **chmod** command (by using **man chmod**), it will help you understand what you are seeing; it has to do with the access/rights/privileges for each of these items.

Since you copied instead of moved your hello.c program, type the following commands to delete the hello.c file and the executable file from your home directory:

% **rm hello.c**

% **rm hello**

NOTE: You could have used the *mv* command instead of the *cp* command above (**cp hello.c cse2421/prelab**) to move the file to the prelab directory then a copy would not have been left behind in your home directory. The only difference is the command itself (*mv* instead of *cp*) as the rest of the line, after the name of the command, would remain the same.

A dot (.) specifies the current directory and the tilde (~) represents the home directory which can be used in combination with other commands when you get a little better at using LINUX commands. It will be up to you to explore more options as what you have been given at this point are just the basics of learning about your new environment.

LAB SUBMISSION

Always be sure your linux prompt reflects the correct directory or folder where all of your files to be submitted reside.

You should submit all your lab assignments electronically using the submit command. The format of submit command is as follows:

% **submit c2421xx labname files-to-submit** where the c2421xx is different depending on course section, labname is the lab you are working on (prelab, lab1, lab2, etc.) and files-to-submit is a list of the file(s) that make up the lab. Remember that you have to be at the correct location (the designated directory) for the command to be able to find your files-to-submit. This is why we created a "prelab" directory so that all of the files (all one of them in this case) could be stored in the same place. Thus, when executing the command above, we would need to be at the ~/cse2421/prelab path for it to work.

See the class slides on Linux for the submit commands and class submit directories for the prelab assignment for Mike Green's sections of CSE 2421 this semester.

NOTE:

• All of the files in a lab MUST be submitted using one command. If you use two submit commands, the second one ***erases the files from the first*** submission (actually, it deletes any files you submitted previously, and only saves the files that you submit with the current submit command). **Do not** press the enter key in the middle of your submit command line; let it wrap if necessary.

• Your programs MUST be submitted in source code form. Make sure that you submit the all the required .c files for the current lab (and .h files when necessary), and any other files specified in the assignment description. Do NOT submit the object files (.o) and/or the executable. This wastes disk space, because the grader will not use executables that you submit anyway. She or he will always build/compile your code using gcc -ansi -pedantic, and run the executable generated by that command.

• **CAUTION: DO NOT SUBMIT** DIRECTORIES using submit!! **If you do this, your lab will receive 0 points, and there are no exceptions to this rule.** The reason for this is that the graders frequently use scripts to grade labs, and the script does not look for files in any directory that you submit.

• It is YOUR responsibility to make sure your code can compile and run on CSE department server **stdlinux.cse.ohio-state.edu,** using **gcc -ansi -pedantic**.

*** More information about lab requirements, point deductions, due dates, late assignments, etc, will be designated on the first lab.

LOGGING OUT

To exit the terminal window, type the following command at the LINUX prompt then hit the enter key:

```
% exit
```

Be sure to logoff your account by choosing the menu option "System" tab, then "Logout" from the drop down menu and confirm.

LINUX EDITORS

1. Vim – type vi at the command line

prompt Home Page: http://www.vim.org/

Written in: C and Vim script.

2. gedit – type gedit at the command line prompt; can also access gedit through the menu system: Applications , Accessories, then gedit Text Editor

Home Page: http://projects.gnome.org/gedit/

Written in: C, Python

3.Nano – type nano at the command line

prompt Home Page: http://www.nano-

editor.org/

3. gVim – type gvim at the command line prompt

Home Page: http://vimdoc.sourceforge.net/htmldoc/gui.html

4. Emacs – type xemacs at the command line prompt; can also access xemacs from the menu system: Applications, Other, XEmacs

Home Page: http://www.gnu.org/software/emacs/

FAQ: http://www.xemacs.org/Documentation/21.5/html/xemacs-faq_1.html#SEC_Top

Written in: C and Emacs lisp

MORE LINUX COMMANDS and examples

http://www.thegeekstuff.com/2010/11/50-linux-

commands/

UNIX TUTORIAL RESOURCES

- http://www.ee.surrey.ac.uk/Teaching/Unix
- http://www.math.utah.edu/lab/unix/unix-tutorial.html
- http://www2.ocean.washington.edu/unix.tutorial.html

To learn more about the Bash shell

- BASH (Bourne Again SHell) is the Linux default shell. It can support multiple command interpreters.
- http://www.gnu.org/software/bash/manual/bashref.html#Bourne-Shell-Variables

Ever Wondered? What does it take to be a programmer?

http://www.cprogramming.com/whatdoesittake.html

What is the speed of dark? Answer during the next class…