**Lab 1: Review of basic commands using a Linux Shell**

Use putty (refer to document on web courses) connect to the linux server.

**The Linux Shell (bash)**

When you log into Linux, you are immediately placed into a command line interpreter called a shell.

There are many shell programs available for Linux but the default shell in

Ubuntu Linux is called bash. In a shell you may enter commands where you see the command prompt, like so:

user@ubuntu:~$

The prompt above conveys certain information about the server and environment as follows:

user

The name of the user who is logged in

@

Means "on" or "at" ubuntu

~

The current directory (or folder if you prefer). ~ is an alias for

"/home/Ubuntu"

$ Conventionally, the character denoting a command prompt

Additional Information : **Command line navigation and editing**

In the likely event that you make a mistake when typing in commands you need to know how best to correct them. Typing everything in again is an option but definitely not the most efficient. Commit the following to memory to save you time when on the command line

←

Left Arrow

Move left one character if not already fully left

→

Right Arrow

Move right one character if not already fully right

↑

Up Arrow

Recall the previous command typed

↓

Down Arrow

Go to the next command typed

Ctrl­‐a *Control-­a*

Go to the start of the command line

Ctrl­‐e *Control­‐e*

Go to the end of the command line

Ctrl­‐r Control­‐r

Start a search for a previously entered command. Followed by typing one or more characters from that command and pressing ENTER when selected

Ctrl­‐l

Control­‐l

Formfeed. **Clear** the screen

Enter a character(s) and press Tab File name completion

Pressing the **TAB** key when tying a file name may complete the typing of it for you if bash can guess what you intent

**Create a folder and move to that directory ; then revert back to root directory**

Type the following (the “user@ubuntu:~$” is the command prompt)

user@ubuntu:~$ echo "hello from Ubuntu Linux "

user@ubuntu:~$ pwd

user@ubuntu:~$mkdir lab1

user@ubuntu:~$ ls

user@ubuntu:~$cd lab1

user@ubuntu:/lab1$pwd

user@ubuntu:/lab1~$cd

*how do you think the previous worked?*

The Linux **help** command is called **man** (manual).

Type in the following to get information on some basic commands

user@ubuntu:~$man man

user@ubuntu:~$man echo

user@ubuntu:~$man mkdir

user@ubuntu:~$man ls

user@ubuntu:~$man pwd

user@ubuntu:~$man cd

user@ubuntu:~$man vi

Note the format of commands on the man page; **ls –l**, **ls –a**, ….

the use of the dash (-­).

List at least **two** command line arguments associated with the ls command

**Creating and Running a basic C program:**

Create a file using the VI editor in the lab 1 directory that you create from the previous exercise.

**vi hello world.c**

press I to insert code into the editor:

write a simple hello world program

pressing Esc to exit the insert mode (back to command mode)

save file and exit editor using **:wq**  or **:q!** (to exit without saving)

compile the program using the gcc compiler:

**gcc - o (filename) hello world.c**

enter ls to confirm the exe file was created

Run the exe using: **./(filename)** or if you did not give the exe a name **./a.out**

**4. The Linux host characteristics**

Now, let's explore the server machine itself. Run the following commands. When you do this there will be certain output displayed after each, not shown here

user@ubuntu:~$cd

how can cd command be used to change using the *absolute path* as opposed to the *relative path* ?

user@ubuntu:~$hostname

user@ubuntu:~$uname -a

user@ubuntu:~$cat /proc/cpuinfo

user@ubuntu:~$ lscpu

user@ubuntu:~$cat /proc/meminfo

user@ubuntu:~$ df -H

user@ubuntu:~$ps -ef | more

The last command is an example of a *pipeline*. There are two commands being issued at the same time. The first one's output is "piped" to a screen pager which allows you to see all of the output of a command which would otherwise exceed the number of visible lines on your screen.

What did each of the previous commands do?

Additional exercise (if you have time)

using the vi editor write and compile the **histogram**  c program that was covered in last week’s class.