

Basic Computer Skills

Goals:

This paper introduces the basic skills needed to complete future assignments, including:

- logging into and navigating the Linux system
 - log in
 - change password
 - learn basic navigation/file system commands
 - create a text file
 - compile a text file
 - run a program
- logging into and navigating the Windows system
 - log in
 - change password
 - remotely access a CS Linux machine

Guided Activities:

The following step by step instructions will describe how to complete all of the operations listed in the goals section. It is to your benefit to try out the following activities, but feel free to discuss them with your peers.

A. Navigating the Linux system

1. If your Linux account still has the password assigned by root, it's a good idea to change it. For simplicity, you may want to use the same password you used for your Windows login.
 1. At the prompt, type in: `yppasswd`
 2. Follow the directions as they appear
2. Now that you've seen how to login, and plugged a big security hole your account, it's time to learn more about moving around the way file system. Linux uses a hierarchical system of folders to organize its files. Simply put, you have a structure of folders. Each folder may contain either files or other folders, or both.

When you log in, you're automatically brought to your home directory. To see what directory you're currently in, type **pwd** (print working directory) at the command prompt:

```
$ pwd
```

You will get:

```
/users/yzhang
```

**note: When you type `pwd`, you will get your user name in the place of `yzhang`.*

**note1: `$` is the default prompt on our Linux system, so you do not need to type it in. This is the way the computer tells you it's waiting for you to do something. Some other Linux systems use `>` or `%` as prompt.*

This directory is where you can store your personal files. The notation `/users/yzhang` denotes that there is a directory `yzhang` that is nested inside of another directory called `/users`. We would normally call `/users` a *parent directory* of `yzhang`, and call `yzhang` a *subdirectory* of `/users`.

To list the contents of the current directory, type `ls` (short for list). Try the following options:

```
$ ls
$ ls -l
$ ls -a
$ ls -al
```

`ls` gives the standard listing of the directory. `ls -l` will display additional information about each files. `ls -a` will show all files, including hidden files.

To make a new directory, you use the **`mkdir`** command (short for make directory). Try using the `mkdir` command to make a folder to store all of your future assignments:

```
$ mkdir CSCI1320
```

You can use `ls` to verify that the directory `CSCI1320` was created properly. Please note that file and folder names on a Linux system are case sensitive. Thus, `CSCI1320` is NOT the same as `csci1320`.

Let's change our current directory to be the new folder we just created. To accomplish this, we use the **`cd`** command (change directory).

```
$ cd CSCI1320
```

Now, if we type `pwd`, we get:

```
$ /users/yzhang/CSCI1320
```

If you type `ls`, you'll see that the directory is empty. However, there are some hidden directories that are automatically added to every folder. To display them, use:

```
$ ls -a
```

You should get the following output:

```
...
```

The `“.”` refers to the current directory. Thus, if we type:

```
$ cd .
$ pwd
```

We'll get the directory we started in. The `“..”` refers to the parent directory. Thus, if we started in `/users/yzhang/CSCI1320` and typed:

```
$ cd ..
```

```
$ pwd
```

we would get the /users/yzhang directory.

If you ever want to refer to your home directory, you can use the ~ symbol. Thus, to get to my /users/yzhang/CSCI1320 directory, I could use the following command:

```
$ cd ~/CSCI1320
```

If you ever wish to clear the screen, you can use the **clear** command:

```
$ clear
```

Two more commands, **cp** and **mv**, will be demonstrated after the introduction to the text editor that follows.

3. Now that now the basics of moving around the Linux file system, it's time to create a file of your own. Many people like **vi**, though there are many others who hate it with a vengeance. If you'd like to use vi, you are more than welcome to. I recommend <http://www.thomer.com/vi/vi.html> as a good place to get started. **Pico** is another text editor available on Linux. It is very easy to learn and more than adequate for the documents you will write in this class. Let's create a file called hello.c:

```
$ pico hello.c
```

You'll notice that your screen will change. You are now running the pico editor. Until you exit out of it (see the commands at the bottom of the screen for the commands to save, exit, etc), you cannot use any of the commands like cd, ls, etc. Pico is almost exactly like using notepad in Windows. All it does it allow you to store text in a file.

Type the following program into pico exactly as it is shown:

```
/* The traditional first program in honor of Dennis Ritchie who
   invented C at Bell Labs in 1972.
*/

#include <stdio.h>

int main(void)
{
    printf("Hello!\n");

    return 0;
}
```

Now, press ctrl and O at the same time (hit enter when prompted), then press ctrl and X. This saves your file, then exits pico.

You should see the command prompt once again. Now, you can enter ls to see the file you just created:

```
$ ls
```

This should return:
hello.c

Although this is a valid c program, it's not yet in a form that the computer can understand. To convert it, we need to use a *compiler*. To invoke the compiler on our system, type:

```
$ cc hello.c
```

When you do this, you'll either get errors, or be returned to the command prompt. If you get errors, read them and see if they tell you what you may have mistyped, or, if that fails, ask for help. If you see just the command prompt, your file has most likely been compiled correctly.

The compiler should have outputted a file called **a.out**. You can check by using ls. If it is not there, it has not been compiled correctly and you will need to recompile.

Once the file is produced, you can run the program using:

```
$ ./a.out
```

If it prints out "Hello!", Congratulations! You've just written a working program!

Now that we have some files to work with, let's try a few more commands for renaming, moving, copying, and deleting files.

The **mv** command is used both for moving files around and changing their names. For instance, try changing the name of the a.out file you created to hello:

```
$ mv a.out HELLO
```

This moves the source file, a.out, to the target file, HELLO. Typing ls will show that a.out is gone, and a file called hello with the same contents is in the current directory.

It can also be used for moving files or folders into different directories. Try moving a.out into CSCI1320's parent directory:

```
$ mv a.out ../
```

or

```
$ mv a.out ../a.out
```

To make a copy of a file, use the cp command. For instance:

```
$ cp hello.c hello1.c
```

will create a copy of hello.c called hello1.c

To delete a file, use the **rm** command:

```
$ rm hello1.c
```

will delete the file hello1.c

Deleting a directory is similar, except that the command is rmdir, and you must delete the contents of a directory before you delete that directory. Or you can delete a directory without deleting the contents of the directory in ahead:

```
$ rm -r CSCI1320
```

You should be carefully to use this command because it will delete all contents under CSCI1320 directory (I do not think you want to do this forever!).

There are some other commands that you may find helpful. First is the **man** command. man is the help system in Linux. To get the manual page on any command, which contains a description of how to the command, type:

```
$ man <command>
```

Thus, to get the manual page on the ls command, you would type:

```
$ man ls
```

To get out of the man page, hit the "q" key.

more command can help you see more of the text. Type:

```
$ more hello.c
```

history command shows a history of the commands you have typed with its history number beside. Type:

```
$ history
```

exit command allows you to exit from a program, shell or log you out of a Linux network. Try:

```
$ exit
```

On your own:

Try to complete the following tasks using what you have learned:

1. Under your home directory, create a directory called “foo” with a directory inside it called “bar”.
2. Copy your hello.c file into foo/bar
3. Rename the hello.c file in your foo/bar directory to howdy.c
4. Edit howdy.c and change the word “Hello!\n” to “Howdy!\n”
5. Compile howdy.c
6. Run the compiled file.

B. Navigating the Windows system

If you want to work on a CS Linux machine from a computer which has Windows Operating System, there is a program on the Windows machine you are logged into that allows you to work on the remote machine. This program is called Putty. You can download Putty from the department FTP web site:

<http://www.cs.trinity.edu/ftp/pub/winxp-software/>

and use it to connect to the Linux machines in the CS building from any other computer.

a. Logging in

At a Windows computer, hold down the ctrl, alt and delete keys simultaneously. This should bring up a login dialog. Your user name and password are the ones you use to login trinity domain.

Example: Yu Zhang with password 12345678 would have:

```
Username:  yzhang  
Password:  12345678
```

b. Change your password

It’s pretty easy for someone else to get into your account right now. Since no one wants someone else getting in to their account, now would be a good time to change your password.

- i. Press the ctrl, alt, and delete keys at the same time.
- ii. Select the change password option.

iii. Type in your old password, and the new password. Please be careful in your choice of password; make it something easy to remember but hard for others to guess.

c. Remotely login to the Linux system:

- Open Putty – Click on All Programs -> Computer Science -> PuTTY -> PuTTY
- Enter the following Host Name: bianca.cs.trinity.edu
- Select the SSH protocol
- *note: at this time, you can save these settings so you don't have to type them in again. Just put some name in the "Saved Sessions" field, then hit the Save button.*
- Hit the "Open" button
- Use your username and **Linux** password when prompted. The password change in Windows does not affect you the password on the Linux machine.
- If you've logged in correctly, you'll see a message and a prompt.
- After the prompt, type in command **ssh** janus01. This command locally login you to a local Linux machine named janus01. janus01 is one of computers in classroom 228; you can ssh any computer in this classroom from janus01 to janus21.

If this is your first time login, you will be asked to generate a public key.

The message is like this:

*The authenticity of host 'janus01 (131.194.71.151)' can't be established.
RSA key fingerprint is
e3:10:98:ad:a8:2b:f5:37:05:49:88:05:d9:44:b2:64. Are you sure you
want to continue connecting (yes/no)?*

You will type yes.

Next you will be asked for password. After you type in the **Linux** password, you will successful login janus01.