Step 1: Getting Started

This assigment is done in class on (monday)01-06-14 AND (friday)01-10-14. This document also describes a lot of commands you'll use a lot in this course and will likely need to remember.

This assignment is due for monday class 01-06-14 AND for the firday class 01-10-14.

1. Some Unix Commands

To open a terminal window on your screen, do Applications/Accessories/Terminal.

You should now have a terminal window on the screen. Remember how to do this; we'll often use a terminal window. We'll type our commands into that terminal window.

pwd

The pwd command prints the current directory. Type this command:

```
pwd
```

It should say something like: /user/yourid.

ls

The ls command lists the current directory. Type this command:

```
ls
```

You should see a listing of the files in your directory. There probably are not many, yet. To ensure we have something to look at, type the following command:

```
touch MyFirstFile
```

Now do an ls command and you should see the file you just created. The touch command "touches" a file, setting the time and date of the file to the current time and date. If the file does not exist, it creates an empty one.

Sometimes you want to know things about the files. You can get a more verbose output from ls using this *switch*.

```
ls -l
```

Now you should see a listing that has the file and its date and time.

Here's two more variations that are useful:

```
ls -a
```

This lists all files, including a lot of "hidden" files that your system uses. You'll see many more files this way.

```
ls -F
```

This lists directories with a trailing "/". This makes it easier to tell directories from files.

rm

The rm command removes a file. Type the following command to remove that file we just created:

```
rm MyFirstFile
```

Do a directory listing to be sure this worked.

mkdir

The mkdir command creates a new directory. Type this command:

```
mkdir cse251
```

Now do a listing to be sure it created the directory.

Create another directory call: stuff

rmdir

The rmdir command removes a directory. Type this command:

```
rmdir stuff
```

The stuff directory should have just disappeared.

> The rmdir command will only remove an empty directory. If you have content in the directory it will fail with a message like "failed to remove stuff: Directory not empty". In that case, you can remove the directory using this command:
>
> rm -r

cd

The cd command changes the current directory. Type this command:

```
cd cse251
```

You are now in that directory. If you do a listing, you'll see no files because we have not created any. Try typing pwd to see the current directory.

If you type 'cd cse251' again, it will fail because there is not a cse251 directory in the current directory.

There are several special directory names. Type this command:

```
cd ..
```

The special directory name ".." means the parent of the current directory. If you list the directory, it will now be the same home directory you were in before. Chang back into the cse251 directory again. Type:

```
cd .
```

The special directory name "." means the current directory. We just told it to change to the current directory. It didn't do anything, of course, since we are already in the current directory. But, we'll use this later on for other things.

Type this command:

```
cd ~
```

The ~ directory name means your home directory. If you pwd, you should be there.

2. Paths

Try typing the following command:

```
cd ~/cse251
```

After each command type pwd to be sure you know what it does.

You have now typed a complete path. We separate the directory names with a "/" character. This command takes you to the directory cse251 that is a child of the home directory. You should be able to type this again and it should work, leaving you in the same directory.

Try this command:

```
cd ../cse251
```

This moves you to the directory cse251 as a child of the parent directory. This will be the same directory.

Type this command:

```
cd ../cse251/.././cse251
```

Remember that ".." means parent directory and "." means the current directory. Do you know what this command is doing? Think of it as driving back and forth between directories.

Summary of Important Commands So Far

pwd	Displays the current directory	
ls	List the current directory	-l : Include file details, -a : Include hidden files, -F : show directories
rm	Remove a file	
mkdir	Create a new directory	
rmdir	Remove a directory	Or rm -r if the directory is not empty.
cd	Change current directory	

3. Files

If you are not already, move into the cse251 directory. Type the following command:

```
ls /user/cse251/exercises
```

We have again used the ls command to list the contents of a directory. But, this time we told it which directory to list. This is the class account exercises directory. You should see a file hello.c in that directory.

cp

The cp command copies a file. Type this command:

```
cp /user/cse251/exercises/hello.c .
```

The "." means current directory. If you do an ls you should now see a copy of the file in your directory.

Now type the following commands to compile and run this program:

```
gcc hello.c
./a.out
```

You should see a message from the course staff. The first command used the GNU C Compiler to compile your program. The second command executed the standard output file from the compiler: a.out. Note the ./ means the file to execute is in the current directory. It won't work without it (try it).

mv

The mv command moves or renames a file. Type this command:

```
mv hello.c myhello.c
```

This renames the file hello.c to myhello.c.

To move a file to another directory, type the filename then the directory instead of a new filename. Be careful, if the directory does not exist, it may rename your file to the name of the directory.

less and cat

Type this command:

```
cat myhello.c
```

The cat command displays the contents of a file.

Type this following command:

```
less myhello.c
```

The less command is a simple program to display the contents of a file. Type "q" when done to exit. less not only displays the file contents, but allows you to page up and down in the file and to view it a page at a time.When in less, hit "h" for help.

4. Help

Type this command to get details on the options for the ls command:

```
man ls
```

You can page through with the space bar and hit q to quit when you get tired of looking at this. This is how you get documentation on a command you may need to use.

Now, try this command:

```
yelp &
```

This starts a program that makes it easier to view available help. The & at the end tells it to run as another process. This means the command starts it and immediately returns, while yelp continues to run in another process on its own.

This is a nicer way to view manual page and other Linux help. In the box labeled Search, type:

```
man ls
```

Isn't this easier?

5. ps and kill

Copy the file /user/cse251/exercises/evil.c into your cse251 directory.

Type these commands:

```
gcc -o evil evil.c
./evil
```

The first command compiles the program evil.c. The -o switch tells it to create an output file named "evil" instead of "a.out".

The second command starts the program.

You should notice that you can't do anything on that terminal anymore. The reason is because this program is broken and creates a continuous loop. It's a program that never ends. From the menus, open another terminal window. In the other terminal, type this command:

```
top
```

This brings up a display of what processes are using how much CPU and puts the heaviest user first. You should see evil right there likely using 100% of the CPU. We need to fix that. Press q to quit top.

Type this command in the working terminal window:

```
ps u
```

You should see a list of running processes, each with a process id (PID). Using the PID for the evil process, type this command, subtituting your PID for my example of 15507:

```
kill 15507
```

Now do a ps -a again and evil should be gone. Look at top again and it should not show that process anymore.

> Important: You are going to occasionally make a mistake a create a program with a continuous loop or that otherwise does not end. This is the process you use to kill such an offending process.

Summary of Important Commands So Far

pwd	Displays the current directory	pwd
ls	List the current directory	ls ls -l ls -a -l : Include file details, -a : Include hidden files, -F : show directories
rm	Remove a file	rm filetonuke.c
mkdir	Create a new directory	mkdir cse251
rmdir	Remove a directory	rmdir cse251 Or rm -r if the directory is not empty.
cd	Change current directory	cd ~/cse251
cp	Copy a file	cp souce destination
mv	Move or rename a file	mv hello.c newname.c
cat	Display the contents of a file	cat hello.c
less	Display file contents nicely	less hello.c
man	Manual pages	man ls
yelp	Nicer help display	yelp &
top	Displays CPU usage	top q to quit
ps	Lists processes	ps u
kill	Kills a process	kill 15577

6. The Remainder of the Day

At this time, start the program gedit, either from a menu or from the command line using this command:

```
gedit &
```

The first thing I want to do is change a setting in gedit. Select Edit/Preferences. In the View tab, check the option "Display line numbers". In the Editor tab, change Tab Width to 4 and check Enable automatic indentation. Then hit Close. This setting should stay later on.

Open the file myhello.c. This is the simple C program you ran earlier. It should look like this:

```
#include <stdio.h>

int main()
```

```
{
    printf("Hello, and welcome to CSE 251!!!\n");
}
```

The first thing we are going to do is to add a comment to this program. Comments in C are content between /* and */. Add this before "int main()", substituting your name for '(your name)':

```
/*
 I'm (your name)
 This is my first C program!
*/
```

Save this. Compile and run it to be sure you've not messed something up. If you get an error try to figure out what you did wrong. Be sure there are no spaces between the / and * each time.

What is in the quotations makes (double quote!) in the printf statement is what is going to print. "\n" means "newline", causing a return. Change the program by adding an new printf statement right after this one that says "I'm so glad to be here!". Be sure you put a semicolon on the end of the line just as I have done.

And We Are Done!

Go to http://www.cse.msu.edu/handin. Select CSE 251 and Step 1 and hand in the file myhello.c. Then we are done for today.

CSE 251