Introduction to Linux - Practical Session

Computer-based practical session:

- Navigation of Linux system
- Manipulating files and directories

Access the terminal by going to applications and type "Terminal"

Tutorial I: Navigation of Linux Systems

1.1 Listing files and directories (ls)

When you first login, your current working directory is your home directory. Your home directory has the same name as your user-name, and it is where your personal files and subdirectories are saved.

To find out what is in your home directory type

```
ls
```

The ls command lists the contents of your current working directory.

However, it does not cause all the files in your home directory to be listed, but only those ones whose name does not begin with a dot (.) Files beginning with a dot (.) are known as hidden files and usually contain important program configuration information. They are hidden because you should not change them unless you are familiar with Linux.

To list all files in your home directory including those whose names begin with a dot, type

```
ls –a
```

ls is an example of a command which can take options: -a is an example of an option. The options change the behaviour of the command. To list all the files with file permissions

```
ls –l
```

1.2 Making Directories (mkdir)

We will now make a subdirectory in your home directory to hold the files you will be creating and using in the course of this tutorial. To make a subdirectory called unixstuff in your current working directory type

```
mkdir unixstuff
```

To see the directory you have just created, type

```
ls
```

1.3 Changing to a different directory (cd)

The command cd directory means change the current working directory to 'directory'. The current working directory may be thought of as the directory you are in, i.e. your current position in the file-system tree.

To change to the directory you have just made, type

```
cd unixstuff
```

Type ls to see the contents (which should be empty)

Exercise 1a ** Make another directory inside the **unixstuff directory called **backups**

1.4 The directories . and ..

Still in the **unixstuff** directory, type

```
ls -a
```

As you can see, in the **unixstuff** directory (and in all other directories), there are two special directories called **.** and **..** sIn Linux **.** means the current directory, so typing

```
cd .
```

There is a space between cd and the dot. There is normally always a space between the command and the argument. This may not seem very useful at first, but using (**.**) as the name of the current directory will save a lot of typing, as we shall see later in the tutorial. (**..**) means the parent of the current directory, so typing

```
cd ..
```

will take you one directory up the hierarchy (back to your home directory). Try it now.

Typing cd with no argument always returns you to your home directory. This is very useful if you are lost in the file system.

1.5 Pathnames (pwd)

Pathnames enable you to work out where you are in relation to the whole file-system. For example, to find out the absolute pathname of your home-directory, type cd to get back to your home-directory and then type

```
pwd
/home/collins
```

Exercise 1b

Use the commands ls, pwd and cd to explore the file system. (Remember, if you get lost, type cd by itself to return to your home-directory)

1.7 Shell Shortcuts for bash

```
Ctrl-A #(jump to start of line)
Ctrl-E #(jump to end of line)
Ctrl-K #(delete (kill) everything from the cursor onwards
Ctrl-W #(delete the previous word only)
Ctrl-Y #(paste whatever was just deleted)
Ctrl-C #(kill/exit a running process)
Ctrl-L #(clear the screen)
Ctrl-R #(search for previously executed commands)
Tab #(auto-complete command or file/directory name)
↑ / ↓ #(scroll back / forwards through previously entered commands)
```

Tutorial II : Manipulating files and directories

2.1 Copying Files and Directories (cp)

cp *file1 file2* is the command which makes a copy of **file1** in the current working directory and calls it **file2**.

What we are going to do now is to take a file stored in an open access area of the file system, and use the cp command to copy it to your *unixstuff* directory.

First, change to your *unixstuff* directory.

```
cd ~/unixstuff
```

Then at the shell prompt type:

```
touch protozoans.txt
cp protozoans.txt .
cp protozoans.txt ./backups/
ls backups/
```

Directories can also be copied with the cp command, but it's necessary to add the option **–R** to do so. This option means 'recursive' and will copy the contents of the directory as well as the directory itself, **for example**:

```
cp -R backups backups2
```

Exercise 2a ** Create a backup of your ***protozoans.txt* file by copying it to a file called **protozoans_2.txt**

2.2 Moving files and Directories (mv)

The move command has a variety of similar but subtly different uses. It can be used to move a file to a different location (i.e. a different directory). It can also be used to move *multiple* files to a different directory. It can also be used to rename a file or a directory. **For example**:

```
mv backups2 directory1/
```

This would move *file1* from the current directory into *directory1*.

```
touch file1 file2 file3
mv file1 file2 file3 directory1/
```

This would move *file1*, *file2* and *file3* from the current directory into *directory1*.

```
mv directory1/ directory2/
```

This would rename a directory. Finally,

```
touch file1
mv file1 directory2/file4
```

This would move *and* rename a file in one step.

We are now going to move the file **cfhss.bak** to your backup directory. First, change directories to your *unixstuff* directory (can you remember how?). Then, inside the **unixstuff**

directory, type

```
mv cfhss.bak backups/
```

To see if it worked type

```
ls
ls backups
```

2.3 Removing Files (rm) and Directories (rmdir)

To delete (remove) a file, use the rm command. As an example, we are going to create a copy of the **protozoans.txt** file then delete it.

Inside your **unixstuff** directory, type

```
cp protozoans.txt tempfile.txt
ls
rm tempfile.txt
ls
```

In order to delete an empty directory you can use the command

```
mkdir directory
rmdir directory
```

However this won't remove directories that already have files in them, instead you can use

```
rmdir directory1
rm –r directory1
```

to recursively delete files in directory (use sparingly - **there is no Recycle bin**!)

You can use the *rmdir* command to remove a directory (make sure it is empty first). Try to remove the **backups** directory. You will not be able to since Linux will not let you remove a non-empty directory.

Exercise 2b ** Create a directory called **tempstuff using mkdir, then remove it using the rmdir command.

2.4 Redirection

It is extremely common for processes initiated by Linux commands write to the standard output (that is, they write to the terminal screen), and many take their input from the standard

input (that is, they read it from the keyboard). There is also the standard error, where processes write their error messages, by default, to the terminal screen.

- Standard Input (STDIN) - Usually the keyboard
- Standard Output (STDOUT) - Usually the Terminal
- Standard Error (STDERR) - Usually the Terminal

2.5 Redirecting Standard Output

We use the **>** symbol to redirect the output of a command. Many of the commands we have seen so far write their output to the terminal (for instance *cat*, *ls*, *grep*, *tail*, *head* and *wc* all write to STDOUT). However we can redirect the output of any of these commands to a file instead (the file can have any name you chose. If the file does not exist it will be created, if it does exist it will be replaced.

The command *echo* prints its arguments to standard output. Compare these two commands

```
echo "Hello World"
```

and

```
echo "Hello World" > output.txt
```

You can view the contents of your new file using

```
less output.txt
```

Credits to:

Jon Wakelin, Liam Gretton, Gary Gilchrist, Teri Forey, University of Leicester.

Adapted* from Michael Stonebank's original course* *'UNIX Tutorial for beginners'

THE END

Tutorial By: Collins M. Morang'a and Vincent Appiah; WACCBIP, UG