LINUX Part One

ls
(list)

When you first login, your current working directory is your home directory. Your home directory is where your personal files and subdirectories are saved.

To find out what is in your home directory, type

ls (short for list)

The **ls** command lists the contents of your current working directory. There may be no files visible in your home directory, in which case, the shell prompt will be returned. Alternatively, there may already be some files inserted by the System Administrator when your account was created.

ls does not, in fact, 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 very 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. There are online manual pages that tell you which options a particular command can take, and how each option modifies the behaviour of the command. (See later in this tutorial)

mkdir (make directory)

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 **csnb224yourStudentID** in your current working directory type

mkdir OSStudentID

To see the directory you have just created, type

ls

cd (change directory)

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 OSStudentID

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

Exercise 1a

Make another directory inside the **OSStudentID** directory called **backups** Print Screen and save it as **Ex01a.jpg**

Still in the **OSStudentID** directory, type

ls -a

As you can see, in the **OSStudentID** directory (and in all other directories), there are two special directories called (**.**) and (**..**)

In LINUX, (**.**) means the current directory, so typing **%**

cd .

NOTE: there is a space between cd and the dot

means stay where you are (the **csnb224StudentID** directory).

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.

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

pwd (print working directory)

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

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). No need to print screen this part of the exercise

Understanding pathnames

First type **cd** to get back to your home-directory, then type

ls OSStudentID to list the contents of your **OSStudentId** directory.

Now type

ls backups

You will get a message like this –

backups: No such file or directory

The reason is, **backups** is not in your current working directory. To use a command on a file (or directory) not in the current working directory (the directory you are currently in), you must either **cd** to the correct directory, or specify its full pathname.

To correctly list the contents of your backups directory, you must type

ls OSStudentID/backups

~ (your home directory)

Home directories can also be referred to by the tilde **~** character. It can be used to specify paths starting at your home directory. So typing

ls ~/OSStudentID

will list the contents of your **OSStudentID** directory, no matter where you currently are in the file system.

What do you think **ls ~** would list?

What do you think **ls ~/..** would list?

REVISIONS

ls list files and directories

ls -a list all files and directories

mkdir make a directory

cd *directory* change to named directory
cd change to home-directory **cd ~**
change to home-directory **cd ..** change
to parent directory
pwd display the path of the current directory

LINUX Part Two

cp (copy)

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 **OSStudentID** directory.

You need to download sample file from moodle (or Fb group) and save it in the Desktop folder before you proceed with the example

First, **cd** to your **OSStudentID** directory.

cd ~/OSStudentID

Then at the UNIX prompt, type, **cp ~/Desktop/science.txt .**

(Note: Don't forget the dot (**.**) at the end. Remember, in LINUX, the dot means the current directory.)
The above command means copy the file **science.txt** to the current directory, keeping the name the same.

Exercise 2a
Create a backup of your **science.txt** file by copying it to a file called **science.bak** Show the content of the directory using the previous command in exercise 1 and print screen the content of the directory. Name the captured image as **Ex02.jpg**

mv (move)

mv *file1 file2* moves (or renames) **file1** to **file2**
To move a file from one place to another, use the **mv** command.

This has the effect of moving rather than copying the file, so you end up with only one file rather than two.
It can also be used to rename a file, by moving the file to the same directory, but giving it a different name.

We are now going to move the file science.bak to your backups directory.
First, change directories to your csnb224 directory (can you remember how?).

Then, inside the **OSStudentID** directory, type **mv science.bak backups/.**
Type **ls** and **ls backups** to see if it has worked.

rm (remove), rmdir (remove directory)

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

Inside your **OSStudentID** directory, type
cp science.txt tempfile.txt
ls (to check if it has created the file)
rm tempfile.txt
ls (to check if it has deleted the file)

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 UNIX will not let you remove a non-empty directory.

Exercise 2b

Create a directory called **tempstuff** using **mkdir** , (Print screen the content of your csnb224 directory and save it as EX2B01.jpg)

Remove tempstuff using the **rmdir** command (Print screen the content of your csnb224 directory and save it as EX2B02,jpg).

clear (clear screen)

Before you start the next section, you may like to clear the terminal window of the previous commands so the output of the following commands can be clearly understood.

At the prompt, type
clear
This will clear all text and leave you with the % prompt at the top of the window.

cat (concatenate)

The command **cat** can be used to display the contents of a file on the screen. Type: **% cat science.txt**

As you can see, the file is longer than the size of the window, so it scrolls past making it unreadable.

less

The command **less** writes the contents of a file onto the screen a page at a time. Type

% less science.txt

Press the **[space-bar]** if you want to see another page, type **[q]** if you want to quit reading. As you can see, **less** is used in preference to **cat** for long files.

head

The **head** command writes the first ten lines of a file to the screen.

First clear the screen then type

head science.txt

Then type **head -5**

science.txt

What difference did the -5 do to the head command?

tail

The **tail** command writes the last ten lines of a file to the screen.

Clear the screen and type

tail science.txt

How can you view the last 15 lines of the file?

Simple searching using less

Using **less**, you can search though a text file for a keyword (pattern). For example, to search through **science.txt** for the word 'science', type

less science.txt

then, still in **less** (i.e. don't press [q] to quit), type a forward slash **[/]** followed by the word to search

/science

As you can see, **less** finds and highlights the keyword. Type **[n]** to search for the next occurrence of the word.

grep

grep is one of many standard UNIX utilities. It searches files for specified words or patterns.

First clear the screen, then type **grep science science.txt**

As you can see, **grep** has printed out each line containing the word science.

Try typing

grep Science science.txt

The **grep** command is case sensitive; it distinguishes between Science and science.
To ignore upper/lower case distinctions,

use the -i option, i.e. Type **grep -i science science.txt**

To search for a phrase or pattern, you must enclose it in single quotes (the apostrophe symbol).

LINUX Part Three

Most processes initiated by UNIX 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.

We have already seen one use of the **cat** command to write the contents of a file to the screen.

Now type **cat** without specifying a file to read

cat

Then type a few words on the keyboard and press the **[enter]** key.

Finally hold the **[Ctrl]** key down and press **[d]** (written as ^D for short) to end the input.
What has happened?

If you run the **cat** command without specifying a file to read, it reads the standard input (the keyboard), and on receiving the'end of file' (^D), copies it to the standard output (the screen).

In UNIX, we can redirect both the input and the output of commands.

We use the > symbol to redirect the output of a command. For example, to create a file called **list1** containing a list of fruit,
type **cat > list1**
Then type in the names of some fruit. Press **[ENTER]** after each one.
pear
banana
apple
^D (Control D to stop)

What happens is the **cat** command reads the standard input (the keyboard) and the > redirects the output, which normally goes to the screen, into a file called **list1**

To read the contents of the file, type
cat list1

Exercise 3a

Using the above method, **create another file called list2** containing the following fruit: orange, plum, mango, grapefruit.

Read the contents of **list2**
The form >> appends standard output to a file.
So to add more items to the file **list1**, type **cat >> list1**
Then type in the names of more fruit
peach grape
orange
^D (Control D to stop)

To read the contents of the file, type cat list1

Joining Files

You should now have two files.
One contains six fruit, the other contains four fruit.
We will now use the **cat** command to join (concatenate) **list1** and **list2** into a new file called **biglist**.

Type

cat list1 list2 > biglist

What this is doing is reading the contents of **list1** and **list2** in turn, then output the text to the file **biglist**

To read the contents of the new file, type

cat biglist

We use the < symbol to redirect the input of a command. The command **sort** alphabetically or numerically sorts a list.

Type

sort

Then type in the names of some vegetables. Press **[ENTER]** after each one.

carrot
beetroot
artichoke
^D (control d to stop)

The output will be

artichoke
beetroot
carrot

Using < you can redirect the input to come from a file rather than the keyboard. For example, to sort the list of fruit, type **sort < biglist** and the sorted list will be output to the screen.

To output the sorted list to a file, type, **sort < biglist > slist**

Use **cat** to read the contents of the file **slist** (Print screen the content**: EX03.jpg)**