ECS518U: Operating Systems

Lab 1: Introduction to Linux Commands

This exercise is not assessed (but you must get it ticked off by the demonstrators)

Preliminary

Aims

The aim of this lab exercise is to use basic Linux commands to manipulate files and directories. Depending on how familiar you are with these commands **you may need to do more, or less, research into what these commands do.**

How do I access a Linux terminal?

- In the ITL and Electronics lab all PCs are dual boot: restart your PC and choose "Ubuntu" in the boot menu that will come up. You can log in using your standard QMUL login credentials.
- If you want to do work also on your own laptop/desktop: Read the relevant sections on QMPlus with info for installing Ubuntu/WSL on Windows machines.
- With MacOS you have access to a Unix terminal (which is quite similar to a Linux terminal) but this is not entirely the same environment as the Linux distribution you get in the ITL machines.

Why do I need to do this?

Because all labs for the module will require you to be able to use the Linux OS. Being familiar with Linux / Unix based Operating Systems is a skill that employers value. **You will not need to "remember" all the commands and options for the exam**, but many of these (e.g. ls, cd, pwd, cp, etc.) will be essential for "survival" in the rest of the module.

Command Argument and Options Commands have arguments:

```
command arg1 arg2
```

Arguments are usually separated by spaces. A command can also have an option (or 'flag' or 'switch'):

```
command -opt arg1 arg2
```

The options usually come first. Options are introduced with a '-'.

Getting Help with commands

You can get help in the following ways:

- Follow a tutorial (e.g. https://linuxcommand.org/ , look at https://linux.die.net/, etc.)
- Try any / all of the following (**where `command` is any Linux command – don't just type command in the shell expecting something to happen!!**) You can also find the man pages for commands online, e.g. search for something like "linux man pages grep" on Google. `type command`

```
help command
man command
info command
```

Step 1: Linux Commands for File Manipulation

Open a terminal window in Linux. Look up the following Linux command in the 'man' pages (e.g. typing `man cat` will display the man pages for the command `cat`) cat, cd,

pwd, mkdir, rm, rmdir, link, cp, mv, more, head, tail, chmod, chown

Try each command (**take care when using rm and rmdir – they can wipe out files / entire folder structures with subfolders**). Write a brief description of each command (1-2 sentences max) (on the answer sheet).

Step 2: Using 'ls'

The 'ls' command is complex and has many options. Read the `man` page (`man ls`).

Try 'ls –l' which uses the option 'l' (the letter before 'm', not a digit) and gives lines such as:

```
-rwxr--r--. 1 tassos staffphdguest 949 Oct  5 10:16 Fork.c
```

The man pages do not fully explain all the output and options (though there are instructions on getting more detailed documentation at the end of the man page entry). Try experimenting to answer the following questions:

1. The number '949' in the example above is the file size. What are the units?
2. There is a date given – in the example above 'Oct 5, 10:16'. What happened at this date / time?
3. The format of the date can vary (e.g. try to use the command in a directory where you have some files from e.g. Year 1). Explain the two formats and when they are used.

Now use the `ls` command to list the files inside one of the directories (folders) that you have under your home directory, **but run the command from your home directory**.

Answer the following question:

4. What is the command you issued to list the contents of that directory? What is the argument for the command?

Step 3: Copying Files

Use Linux commands to create a new directory in your file space (e.g. make a new directory ECS518U in your home directory). Then copy the files from any of your directories to that new directory that you created. Record the commands you used. Using the appropriate 'ls' command options / flags, determine:

- What information about the files is preserved by the copy?
- What information, if any, is changed?

Step 4: Using Pipes

A Linux 'pipe' passes data output by one process (something like a program… more in Weeks 2/3) to the input of another process. It is like a temporary and invisible file between the processes. The symbol '|' is used to represent a pipe.

Examples

Try the following examples:

```
ls | wc
```

`ls -l | wc` (note that l is the letter)

In each case explain what is happening (you may have to read the man page on 'wc').

Sorting the output of ls

The Linux program '`sort`' can sort a file, or the output from another program. The aim of this task is to use `sort` to sort the output of the `ls` command:

1. by file name
2. by file size

Note: there are options of the 'ls' command to do this anyway, but using them is **not** the aim of the exercise. Instead:

- Choose options for the 'ls' command to show size
- Choose options for the 'sort' command to control sorting
- Combine using a pipe

Step 5 – File access permissions

We will learn more about this topic during the week 4 lectures (File Systems), but we can have a first taste here. Files and directories in Linux have access permissions for the owner, owner's user group and the rest of the users. In the example of the output of ls in Step 2 we saw:

```
-rwxr--r--. 1 tassos staffphdguest 949 Oct  5 10:16 Fork.c
```

- Do a little bit of research on file access permissions in Linux (plenty of resources on the Web, do your own research).
- Describe what the access permissions are for this file for the owner, his/her user group and for the rest of the users.

The `chmod` command can be used to change the access permissions for files / directories. There are two different ways in which you can use the `chmod` command.

- Do a little bit of research on how you can use the `chmod` command to change access permissions.
- In the example output before, what effect would the following command have?
  `chmod 644 Fork.c`
- Briefly explain the numerical coding used (as in the previous example) to describe file access permissions in Linux.

Answer Sheet

The following page should be completed during/before the lab.

Step 1: Linux file manipulation commands

	Description
cat	Print the entire contents of a file. Concatenate and write to stdout
cd	Change directory. Move your current working directory (folder)
pwd	Print working directory.
mkdir	Make a directory (folder)
rm	Remove (a file). Delete a file
rmdir	Remove a directory. Delete a folder
link	Create a symbolic link between a file and a location
cp	Copy. Copy a file from place X to Y
mv	Move. Move a file from place X to Y (like cut)
more	Scrolling view of a file. Allows you to tab through without scrolling
head	Output the first part of files. Cat the beginning of a file.
tail	Output the last part of files. Cat the end of a file.
chmod	Change the file mode. Chmod 777 (or +x) to give read write execute permissions.
chown	Change the file ownership.

Step 2: Using 'ls'

Answers to the questions (refer to lab sheet for the questions)

1. Almost all sizes in Linux are represented in bytes. Remember to divide by 1024 NOT 1000 to convert to MB/GB

2. This was the last modified time of the file. You can try to get the file creation time using the "stat" command but you will rarely be able to get it.

3. The date format can vary between the standard format (e.g. Oct 5, 10:16) and the numerical format (e.g. 2022-10-05 10:16). The standard format is used when the modification time is less than 6 months ago, and the numerical format is used when the modification time is more than 6 months ago.

4. Command: ls /home/morebj/abc

   ls => list files

   argument is the folder you want to look inside

Step 3: Copying Files

- Command used

```
james@James:~$ pwd
/home/james
james@James:~$ mkdir ECS518U
james@James:~$ cd ECS518U/
james@James:~/ECS518U$ touch test.txt
james@James:~/ECS518U$ echo "abcdefjskbdajk" > test.txt
james@James:~/ECS518U$ cat test.txt
abcdefjskbdajk
james@James:~/ECS518U$ ls -lrt
total 0
-rw-r--r-- 1 james james 15 Jan 30 21:58 test.txt
james@James:~/ECS518U$ whoami
james
james@James:~/ECS518U$ sudo -s
[sudo] password for james:
root@James:/home/james/ECS518U# whoami
root
root@James:/home/james/ECS518U# cp test.txt /home/james
root@James:/home/james/ECS518U# cd ..
root@James:/home/james# ls -lrt
total 0
drwxr-xr-x 1 james james 512 Jan 30 21:58 ECS518U
-rw-r--r-- 1 root  root   15 Jan 30 22:00 test.txt
```

- Information unchanged
  File Name, File Size, File Permissions (Chmod)
- Information changed (if any)
  File Modification Time, File Ownership and Permission Group (Chown)

Step 4: Using Pipes

- Sorting files by name; command:

```
ls | sort
```

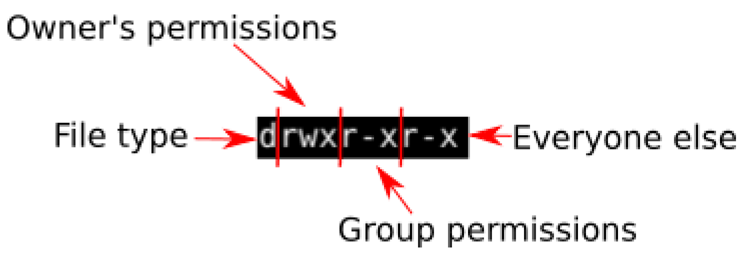
- Sorting files by size; command

```
ls -S
ls -l | sort -nk 5 (Sorting by the 5th column)
```

Step 5: File access permissions

- `-rwxr--r--. 1 tassos staffphdguest 949 Oct  5 10:16 Fork.c`

  What are the file access permissions for user `tassos`, his user group and the rest of the users?

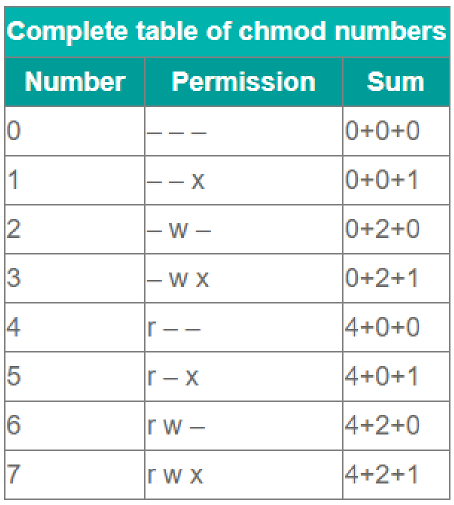


The owner can Read, Write and Execute.

The Group staffphdguest can only Read the file.

Everyone else can only Read the file.

- In the previous example, what effect would the following command have?
  `chmod 644 Fork.c`



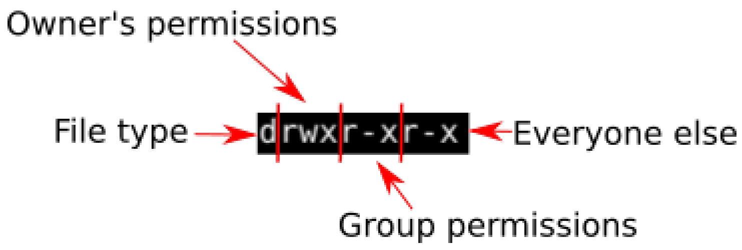
Complete table of chmod numbers		
Number	**Permission**	**Sum**
0	– – –	0+0+0
1	– – x	0+0+1
2	– w –	0+2+0
3	– w x	0+2+1
4	r – –	4+0+0
5	r – x	4+0+1
6	r w –	4+2+0
7	r w x	4+2+1

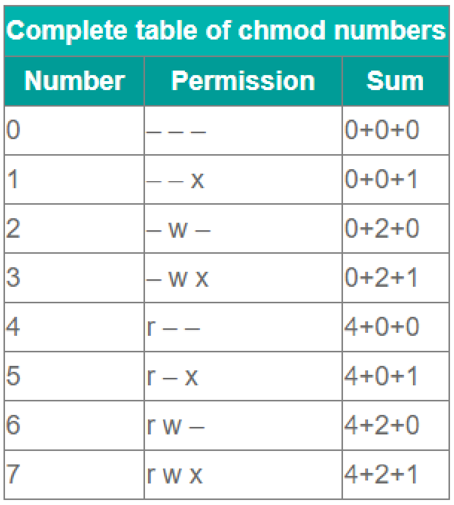
It would give User Read/Write permission while giving User Group and Everyone else just Read only.

```
james@James:~/ECS518U$ ls -lrt
total 0
-rwxrwxrwx 1 james james 15 Jan 30 21:58 test.txt
--wx--x--x 1 james james  0 Jan 30 22:10 abc.txt
-rw-r--r-- 1 james james  0 Jan 30 22:10 zgs.txt
james@James:~/ECS518U$ chmod 000 zgs.txt
james@James:~/ECS518U$ ls -lrt
total 0
-rwxrwxrwx 1 james james 15 Jan 30 21:58 test.txt
```

```
--wx--x--x 1 james james  0 Jan 30 22:10 abc.txt
---------- 1 james james  0 Jan 30 22:10 zgs.txt
james@James:~/ECS518U$ chmod 644 zgs.txt
james@James:~/ECS518U$ ls -lrt
total 0
-rwxrwxrwx 1 james james 15 Jan 30 21:58 test.txt
--wx--x--x 1 james james  0 Jan 30 22:10 abc.txt
-rw-r--r-- 1 james james  0 Jan 30 22:10 zgs.txt
```

- Briefly explain the numerical coding used (as in the previous example) to describe file access permissions in Linux.





Complete table of chmod numbers		
Number	Permission	Sum
0	– – –	0+0+0
1	– – x	0+0+1
2	– w –	0+2+0
3	– w x	0+2+1
4	r – –	4+0+0
5	r – x	4+0+1
6	r w –	4+2+0
7	r w x	4+2+1

The first number refers to the Owner.
The second number refers to the group permission.
The third number refers to everyone else.

The numbers can be easily defined in this table. Where the number is the sum of the row of permissions.