This handout contains some of the most useful and commonly used Linux commands for quick reference. For tutorials on using Linux, see the links page at `http://www-users.york.ac.uk/~bd512/links.shtml`

Files and directories

When using the terminal, you have a current "working" directory (folder) which you're in. Directory names are separated using forward-slash '/', so `documents/essay.doc` means "file called `essay.doc` in a subdirectory `documents`". Some special directory names are:

name	meaning
`.`	Current directory
`..`	Parent directory
~	Your home directory

You can change directories, and run commands on files and directories. Linux comes with manual pages, so to see the page for **ls**, type:

```
$ man ls
```

Some common commands:

Navigating directories	
`pwd`	**P**rint your **w**orking **d**irectory
`ls`	**Lis**t files and directories
`cd`	**C**hange **d**irectory
`cd ..`	Change to parent directory
`cd -`	Go back to previous directory
Files and directories	
`mkdir`	**Mk**e **dir**ectory
`cp`	**Cp**y files
`cp -r`	**Cp**y directories
`mv`	**Mv**e files and directories
`rm`	**Rm**ove files
`rm -r`	**Rm**ove directories

Wildcards

When dealing with files, wildcards can be used to match file and directory names

Symbol	Matches
`*`	Anything
`?`	A single character
`[]`	One of the characters in the brackets

Examples:

```
$ ls *.txt
```

Lists all files ending in `.txt`

```
$ rm test?.txt
```

will delete files called `test1.txt` and `tests.txt`, but not `tests1.txt`.

```
$ mv *.[oc] ..
```

moves all files ending in `.o` or `.c` to the parent directory.

Searching

To find files, there is the **find** command:

```
$ find . -name "*.txt"
```

finds all files in this directory and subdirectories which end in `.txt`. There are many other ways to match files against last-modified times or other attributes using `find` (see the `man` page).

To search for text within a file, the standard UNIX tool is **grep**. To search for "flibble" in all text files in this directory, use

```
$ grep -i -n "flibble" *.txt
```

where the `-i` option means case insensitive, and `-n` makes `grep` print out the line number of each match.

These tools can be combined because `find` can be made to run a command for each file it finds using the `-exec` option:

```
$ find . -name "*.txt" -exec
      grep -i -H -n "flibble" '{}' \;
```

The `-H` option is there to make `grep` print out the file name. **Note**: Be careful when combining `find` with `mv` or `rm`!

Secure SHell

The standard way to connect to other UNIX machines is using encrypted `ssh`. To connect to a machine called `kink` you could use:

```
$ ssh -X kink
```

The `-X` option means "forward X windows", which means that you can run a graphical program on the remote computer and it will display on your desktop.

Note: Make sure you use a capital `X`, as lower-case means *don't* forward X.

If your username on the remote computer is different, then you can specify the username to use:

```
$ ssh -X username@kink
```

To copy files or directories between computers, there is the **s**ecure **c**o**p**y program **scp**. This works like `cp` but the from and to locations can be on different computers.

To copy a file to your home directory on remote computer `kink`:

```
$ scp file username@kink:~/
```

If you want to copy a directory, then you need the `-r` option. Similarly, copying a file from `kink` to your current directory

```
$ scp username@kink:~/file .
```

Other useful tools

All UNIX systems come with a huge collection of tools for manipulating files, text and network connections. One very powerful tool is **sed** which applies *Regular Expressions* to files. The most common use for this is searching and replacing:

```
$ sed "s/replace this/with this/g"
                    file > changedfile
```

Some other tools which you might find handy are:

command	purpose
`cat`	Read a file
`curl`	Download files
`diff`	Differences between files see also `tkdiff` and `meld`
`git`	Track changes
`less`	Navigate through text
`nc`	Network swiss-army knife
`sort`	Sort alpha. or numerically
`tee`	Write text to a file and pipe
`tr`	Translate characters
`wc`	Count words and lines

Other tools which you might like to google: LaTeX is extremely useful for writing technical documents like papers (and these handouts). Automating common tasks can be done using **Bash scripting**. For plotting graphs, **gnuplot** can be automated and combined with other UNIX tools. Similarly, **mail** can be used to automate sending emails.