An Introduction to PWF Linux

Dr MJ Rutter
mjr19@cam.ac.uk

Michaelmas 2001

Typeset by FoilTEX

UNIX

Development of UNIX started in 1969 in AT&T Bell Labs. Version 1, written in assembler, appeared in 1971.

A couple of years later it was rewritten in the new C language, also developed by Bell Labs. In 1975 UNIX was made available outside Bell Labs.

By the early 1990s it was *the* operating system for largish computers. Cray, DEC, HP, IBM, SGI and Sun all sold versions with their computers. None of its competitors achieved this level of *cross-platform* support.

Competitors included MVS (IBM only), VMS (DEC only) and WinNT (effectively Intel only).

Linux

Linux famously started being developed by a Finnish graduate student, Linus Torvalds, in 1991. It was intended to be a UNIX variant running on Intel i386-based PCs. From an early stage Linus made the source freely available and encouraged others to contribute.

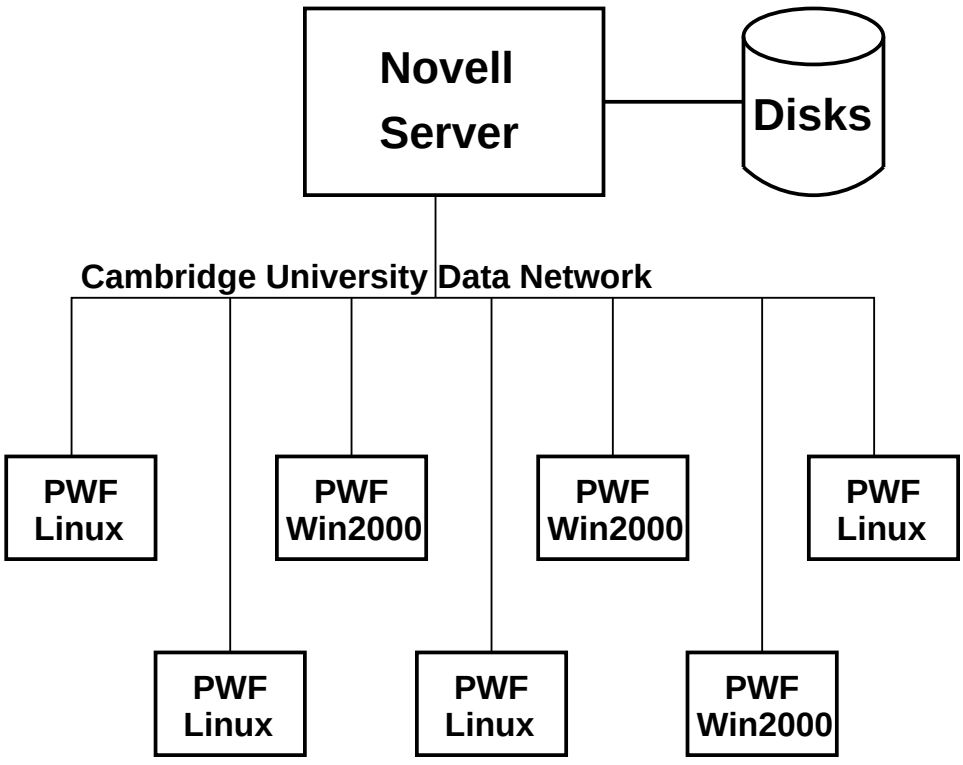
In 1994 version 1 was released, followed by 1.2 a year later, and 2.0 the following year. By this time support had grown to include Macintoshes and Alphas as well as PCs.

The current version is 2.4, released in 2001, and development continues.

The Computer Resources

The computers in the Physics PWF and in the CS-run PWFs in the centre of Cambridge all run a version of Linux which contains all the software needed for this course.

These computers use the same Novell server for your home directories that they do when running Windows.



Booting Linux

If the computer is already offering the Linux logon screen, with its unique penguin, fine. If it is running Windows, select 'Shutdown...' followed by 'Shutdown and Restart' from the login prompt.

If the computer is turned off, turn it on! Shortly a menu will appear:

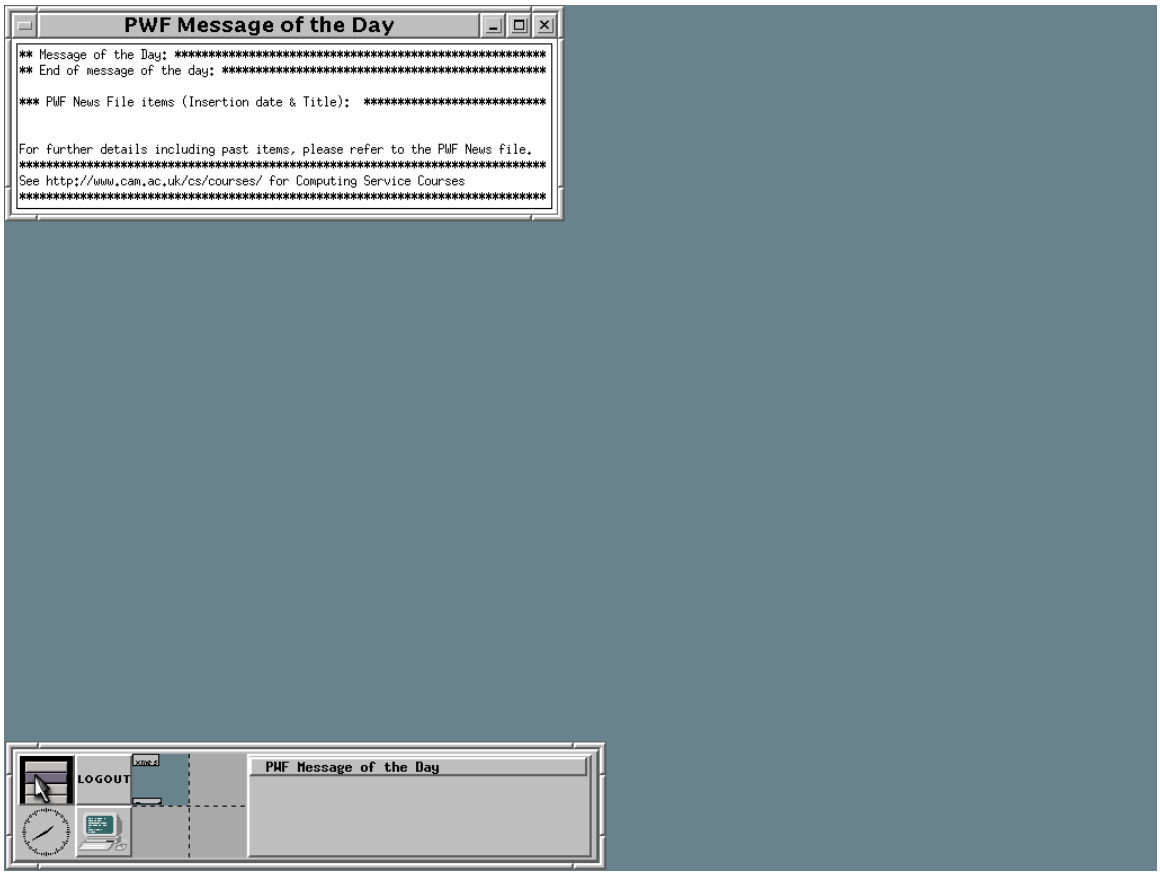
```
Please select operating system to start.

    PWF Microsoft Windows
    PWF Linux
```

Use the ↓ cursor key to select `PWF Linux` and press enter.

Logging in

Use your normal PWF userid and password in the usual way. After a few moments the following should appear.



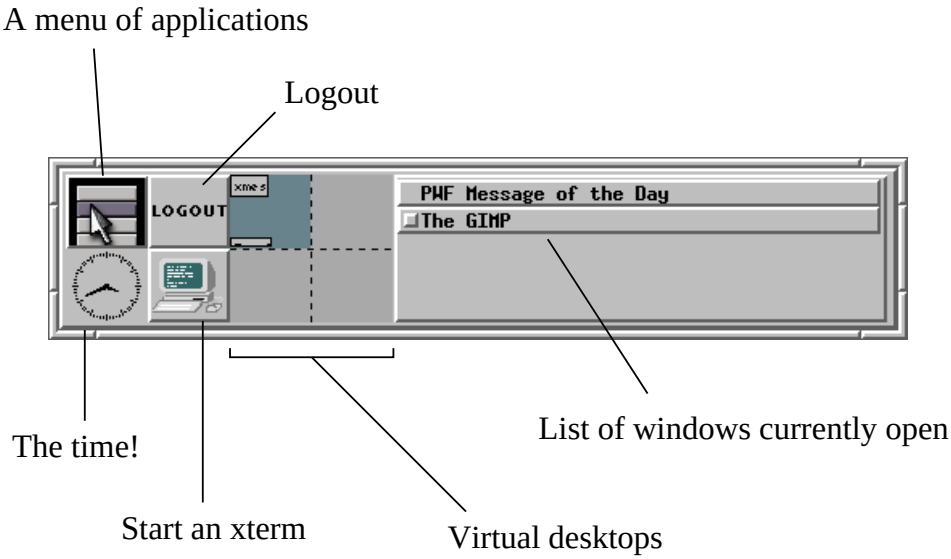
This is fairly close to a default RedHat 7.1 set-up.

Window Decorations

These decorations are broadly similar to those in Microsoft Windows. A left-click in the button on the top left produces a menu of various window operations. The buttons at the top right are iconise, expand to full height and close respectively.

The window can be moved by holding down the left mouse button on the title bar whilst dragging the mouse. Resizing can be done by dragging any edge of the window.

The Toolbox


A menu of applications
Logout
LOGOUT
xmes
PWF Message of the Day
The GIMP
The time!
Start an xterm
Virtual desktops
List of windows currently open

The Root Menu

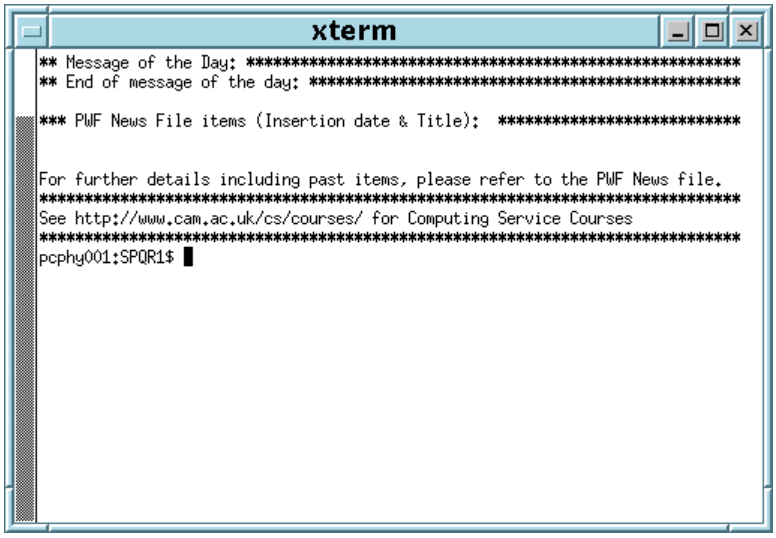
This useful menu can be obtained either by clicking the icon in the top left of the toolbox, or by clicking anywhere on the root window.

The triangle indicates the existence of a sub-menu.

The xterm

The xterm is a most useful and necessary application. It provides a command-line interface to the operating system. A command line interface (CLI) is generally more powerful and flexible than a graphical interface.

An xterm may be started either by clicking on the icon in the toolbox, or by choosing 'Unix shell' from the root menu.



The window with *focus*, that is the window currently accepting input from the keyboard, has a light blue border and is given focus by moving the mouse pointer into it.

The Basics

The xterm is a resizeable window, with a scroll-bar on the left, in which a *shell* or *command interpreter* runs. The latter obeys our commands and produces the prompt:

```
pcphy001:SPQR1$
```

This prompt consists of the machine name followed by the name of the current *directory*. (A directory is what the real world calls that which Windows and MacOS call a 'folder'.)

One can see what files are present by typing `ls`.

```
pcphy001:SPQR1$ ls
maple      PUTTY.RND       WinNT
msoffice   Windows NT...   Xrootenv.0
```

Six files or directories.

Spaces in filenames can be confusing with CLIs!

Note this is the same set of files that would be seen logging in under Windows2000.

The shell that is used by default is called `bash`, although many others are common.

More `ls`

```
pcphy001:SPQR1$ ls -l
total 3
drwxr-x--- 1 spqr1 pwf  512 Sep 25  1999 msoffice
-rw-r----- 1 spqr1 pwf  600 Aug  5 19:29 PUTTY.RND
drwxr-x--- 1 spqr1 pwf  512 Aug 24 15:47 Windows NT...
drwxr-x--- 1 spqr1 pwf  512 Oct 18  1999 WinNT
-rw-rw-r-- 1 spqr1 pwf  715 Aug 25 15:05 Xrootenv.0
```

The first line is the total size of the files, in kilobytes.

Then each line describes one file or directory. The first character is '`-`' for a file, and '`d`' for a directory.

The next three show the permissions the owner of the file has: `r` for read, `w` for write and `x` for execute. Then three for members of the same group as the file, and three for everyone else: there should be no `w`'s in the last three!

The single number can be ignored, after which we see the file's owner (`spqr1`) and the group the file is in (`pwf`). Then length in bytes, date and time last modified, and filename.

Simple file operations

Files can be copied, moved (renamed) and removed (deleted) using the commands cp, `mv` and `rm`.

Be Careful!: deletion is irreversible!

```
pcphy001:SPQR1$ ls
results2.dat  results.dat
pcphy001:SPQR1$ cp results.dat working_results
pcphy001:SPQR1$ ls
results2.dat  results.dat  working_results
pcphy001:SPQR1$ mv results2.dat rubbish.dat
pcphy001:SPQR1$ ls
results.dat  rubbish.dat  working_results
pcphy001:SPQR1$ rm rubbish.dat
pcphy001:SPQR1$ ls
results.dat  working_results
```

To prevent other people from reading a file, type:

```
pcphy001:SPQR1$ chmod go= results.dat
```

Note the space after the '=', and note the change this produces in the output of `ls -l`.

Tidiness

It is best to place files in tidy groups in subdirectories, rather than having everything in one directory. The command `mkdir` creates a directory, and `rmdir` will remove one provided it is empty. The `cd` command changes the current directory.

```
pcphy001:SPQR1$ ls
maple      PUTTY.RND        WinNT
msoffice   Windows NT 5.0   Xrootenv.0
pcphy001:SPQR1$ mkdir Linux
pcphy001:SPQR1$ cd Linux
pcphy001:Linux$ ls
pcphy001:Linux$
```

Unlike Windows, filenames are case sensitive: `Linux` is not the same as `linux`.

For sanity one should limit filenames to containing letters, digits, underscore, period and hyphen, and the first character should never be a hyphen. Some other characters are possible, but some may cause confusion.

Moving around

Directories form a tree: each directory has one parent directory, and may have multiple subdirectories. A filename is assumed to refer to the current directory. Other locations can be specified by forming a *path* using '/' to separate the components of the path, and '..' to refer to a directory's parent.

```
pcphy001:Linux$ ls -F
a/  results.dat
pcphy001:Linux$ cp results.dat a/results.dat
pcphy001:Linux$ ls -RF
.:
a/  results.dat

./a:
results.dat
pcphy001:Linux$ mkdir b
pcphy001:Linux$ cd b
pcphy001:b$ cp ../a/results.dat .
pcphy001:b$ cd ..
pcphy001:Linux$
```

`ls -F` distinguishes directories by placing a '/' after their names.
`ls -R` lists all subdirectories in a recursive fashion.
`cd` typed on its own returns one to one's home directory.

Laziness

You have probably discovered that the cursor keys allow you to edit the current command and recall previous commands in an intuitive manner.

Note also that pressing {`TAB`} when half-way through typing a filename will cause the rest of the filename to be filled in automatically if it is unique.

The *wildcards* '?' and '*' can be used to stand for any one character, and any string of characters respectively:

```
pcphy001:Linux$ ls
a.dat  b.dat  results.dat  write-up.txt
pcphy001:Linux$ ls ?.dat
a.dat  b.dat
pcphy001:Linux$ ls *.dat
a.dat  b.dat  results.dat
pcphy001:Linux$ ls *
a.dat  b.dat  results.dat  write-up.txt
pcphy001:Linux$ rm *
pcphy001:Linux$ ls
pcphy001:Linux$
```

`rm *` should be used with considerable caution.

Text editors

A text editor is not a word processor, and vice versa. Word processors break up lines spontaneously and concern themselves with the minutiæ of typography. This is not what a programmer wants.

Two common GUI UNIX text editors are `nedit` and `emacs`. These have far more features than one needs for simple work. . .

Both bring up their own windows, and are best started with a filename specified on the command line.

```
pcphy001:SPQR1$ nedit liber_I.txt &
pcphy001:SPQR1$
```

The final & ensures that one immediately gets a prompt back in the xterm. Otherwise, the prompt will not reappear until one exits the editor. Both editors are described in more detail in the exercises.

A process started with & is often referred to as a *background* process, as distinct from a *foreground* process which keeps full control of the terminal until it exits.

Cutting and pasting

One can cut and paste text between different windows by highlighting the text to be copied by dragging the mouse over it with the left button held down. Then the mouse can be moved to the point at which the text should be inserted, and the middle button pressed.

Notice that this is done entirely with the mouse: one need not touch the keyboard during this operation.

Viewing files

The `ls -l` command will tell you the size of a file. Other useful comamnds include:

- `file` which guesses a file's type
- `wc` which gives length in lines and words
- `less` which views the file screen at a time

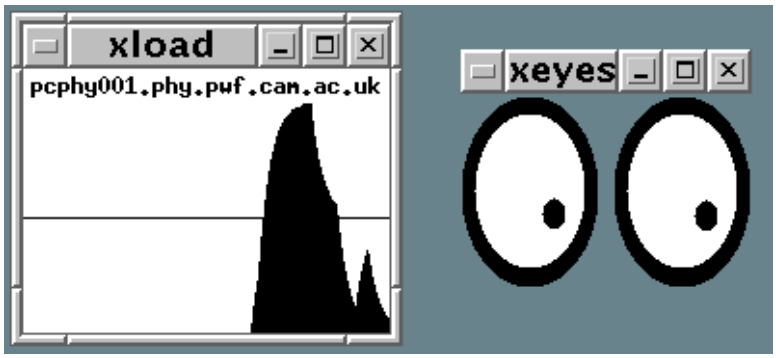
```
pcphy001:SPQR1$ ls -l liber_I.txt
-rw-rw-r-- 1 spqr1  pwf  306 Aug 30 19:24 liber_I.txt
pcphy001:SPQR1$ file liber_I.txt
liber_I.txt: ASCII text
pcphy001:SPQR1$ wc liber_I.txt
      7      45     306 liber_I.txt
pcphy001:SPQR1$ less liber_I.txt
```

Press 'q' to exit from `less`, cursor keys to move around long files.

The report from `wc` is lines, words, characters. The file type reported by `file` is an educated guess based on the first few characters of the file.

Toys

Two amusing 'toys' are `xeyes` and `xload`. The former allegedly helps one keep track of where the mouse cursor is on the screen, the latter of how hard the computer is working.



Notice that `xeyes` does not have a border around its window.

Process management

```
pcphy001:SPQR1$ sleep 30 &
[2] 1644
pcphy001:SPQR1$ ps
  PID TTY          TIME CMD
 1408 pts/0    00:00:00 bash
 1441 pts/0    00:00:00 xterm
 1644 pts/0    00:00:00 sleep
 1645 pts/0    00:00:00 ps
pcphy001:SPQR1$ kill 1644
pcphy001:SPQR1$ ps
  PID TTY          TIME CMD
 1408 pts/0    00:00:00 bash
 1441 pts/0    00:00:00 xterm
 1646 pts/0    00:00:00 ps
[2]+  Terminated              sleep 30
pcphy001:SPQR1$
```

`sleep` is a program which does nothing for the specified number of seconds, then exits.

`ps` lists (some) of the owner's processes. Here we see `sleep`, along with the `ps` command itself, the `xterm`, and `bash`, the command interpreter which runs in the xterm. The `TIME` column gives the amount of processor time each has used, in hours, minutes and seconds, the `PID` column the *process id*, which uniquely identifies the process.

`kill` tells the given process to exit, as long as the process is yours.

All your processes

```
pcphy001:SPQR1$ ps x
 PID TTY      STAT   TIME COMMAND
1347 ?        S      0:00 fvwm2
1395 ?        S      0:00 /usr/.../fvwm2/FvwmButtons
1398 ?        S      0:00 /usr/.../fvwm2/FvwmIconMan
1399 ?        S      0:00 xclock -bg grey -fg black
1400 ?        S      0:00 /usr/.../fvwm2/FvwmPager
1406 ?        S      0:00 /usr/X11R6/bin/xterm -ls
1408 pts/0    S      0:00 -bash
1441 pts/0    S      0:00 xterm -ls
1443 pts/1    S      0:00 -bash
1510 pts/1    S      0:00 xeyes
1530 pts/1    S      0:00 xload
1650 pts/0    R      0:00 ps x
```

Here `fvwm2`, the *window manager* responsible for the borders around windows, the clock, pager and icon manager making the task bar, two xterms and their shells (bash), `xeyes`, `xload` and the `ps` command.

Some processes, e.g. `bash`, do not exit when the `kill` command is used on them. For these,

```
kill -KILL 1408
```

will succeed.

Process trees

Every process must have been started by another process, and every process can start other processes.

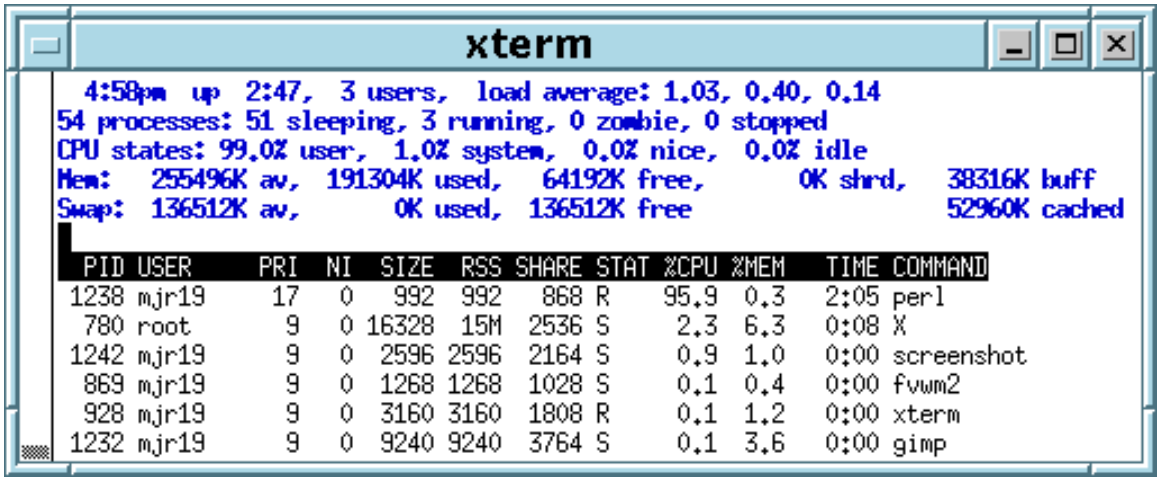
Hence every process has a *parent*, and can have *children*.

This concept is so strong that if a process exits for any reason, its parent will be told. If its parent exits, the child process is immediately adopted by a process called `init` (whose PID is invariably 1).

Typing `ps xf` produces a semi-graphical representation of this, whereas `ps xl` gives the information numerically, the `PPID` column being the parent process id.

Process monitoring

The `top` command provides a convenient way of monitoring which processes are running. It orders its output, which is continually updated until one quits by pressing 'q', by the amount of CPU time each process is using.



Printing

The command `lpr` prints a file to the printer. The default printer is chosen automatically in an intelligent manner based on where you are logged in. One should attempt to print plain text files and Postscript files only: printing other random data will cause disappointment and expense.

```
pcphy001:SPQR1$ lpr example.txt
```

To specify a particular printer:

```
pcphy001:SPQR1$ lpr -PPhy_Bragg_CLJ colour_diag.ps
```

Finally, use `a2ps` rather than `lpr` to print a plain text file with two pages per sheet (half price!), and with Fortran code neatly formatted.

Type `lpr -h` for a complete list of printers known. You do not necessarily have access to all of them!

`lpr` is an abbreviation of lineprinter, not laserprinter, though you may find thinking of the latter makes it easier to remember.

It's not all free

Two aspects in particular of your use of these computers are constrained. Firstly your home directories have a quota (a maximum size). You can find its value and your current usage by typing:

```
pcphy001:SPQR1$ quota
Entry 0: Level: 2
  Max:      25600 KB
  Current: 4632 KB
  Used:     20968 KB
```

Here the quota is 25MB, about $4\frac{1}{2}$MB are free and just over 20MB are used.

Printing is also charged.

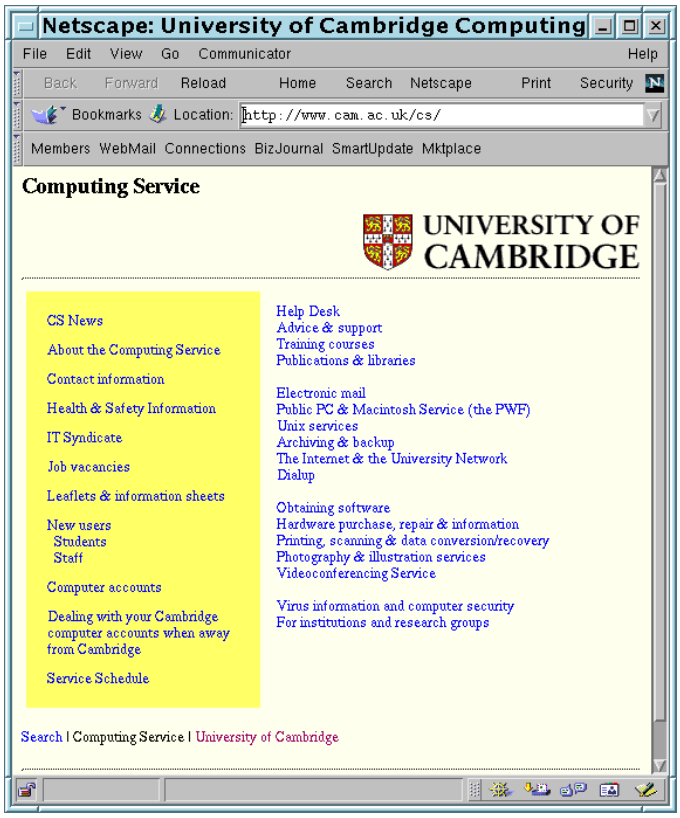
For credit on the CS printers, one must speak to the Computing Service Reception, for Physics observe the notices in our PWF, etc. To find out your current credit (in pence) choose 'Utilities' from the root menu, followed by 'prcred'.

To find how much space, in kilobytes, a directory and all it contains occupies, cd to its parent and type '`du -sk` *dir*'.

Wasting time

Though it is alleged that WWW browsers have academic uses too...

Internet Explorer is not available for Linux, but netscape is and can be started by typing '`netscape &`' or by choosing 'netscape' from the 'Web browsers' item on the root menu.



Redirection

Most Unix commands can have their output redirected to files or other commands.

```
pcphy001:SPQR1$ ls
results.dat  rubbish.dat
pcphy001:SPQR1$ ls > ls.out
pcphy001:SPQR1$ ls
ls.out  results.dat  rubbish.dat
pcphy001:SPQR1$ less ls.out
ls.out  results.dat  rubbish.dat
```

Use $>$ to send output to a file, and $>>$ to append output to a file. Use | to send the output to the input of another command, e.g. to display all processes, one page at a time, type:

```
pcphy001:SPQR1$ ps ax | less
```

`ls` is unusual in that it changes its behaviour when used with $>$ and switches to single column output. You will not be able to reproduce the above exactly!

Note also, | is not the key on the top left of your keyboard, but rather that marked with a broken vertical line, and probably found beside 'z'.

Configuration Files

Unix programs often store their configuration files in one's home directory, with names starting with '.'.

Often known as 'dot files.'

Such files are not shown by `ls` by default, nor does '`*`' match such files.

To find them, `ls -a` will serve.

One, called `.bashrc`, contains commands which each shell will execute on startup. This is described in more detail in the exercises.

Further help

Unix has a `man` command, which invokes the on-line manual. The style of this manual is compact, technical and of most use to those who already know the answers! Look first at a man page of a command with which you are familiar by typing '`man wc`'.

WC(1) FSF WC(1)

NAME

wc - print the number of bytes, words, and lines in files

SYNOPSIS

wc [OPTION]. . . [FILE]. . .

DESCRIPTION

Print line, word, and byte counts for each FILE, and a total line if more than one FILE is specified. With no FILE, or when FILE is –, read standard input.

–c, **––bytes**

print the byte counts

–l, **––lines**

print the newline counts

[etc.]

Logging Out

When leaving a computer it is important to remember to log out of it. Under Linux this is readily achieved: either click the large 'Logout' button on the taskbar, or select the logout item from the root menu. Both will prompt for confirmation.

Exercises

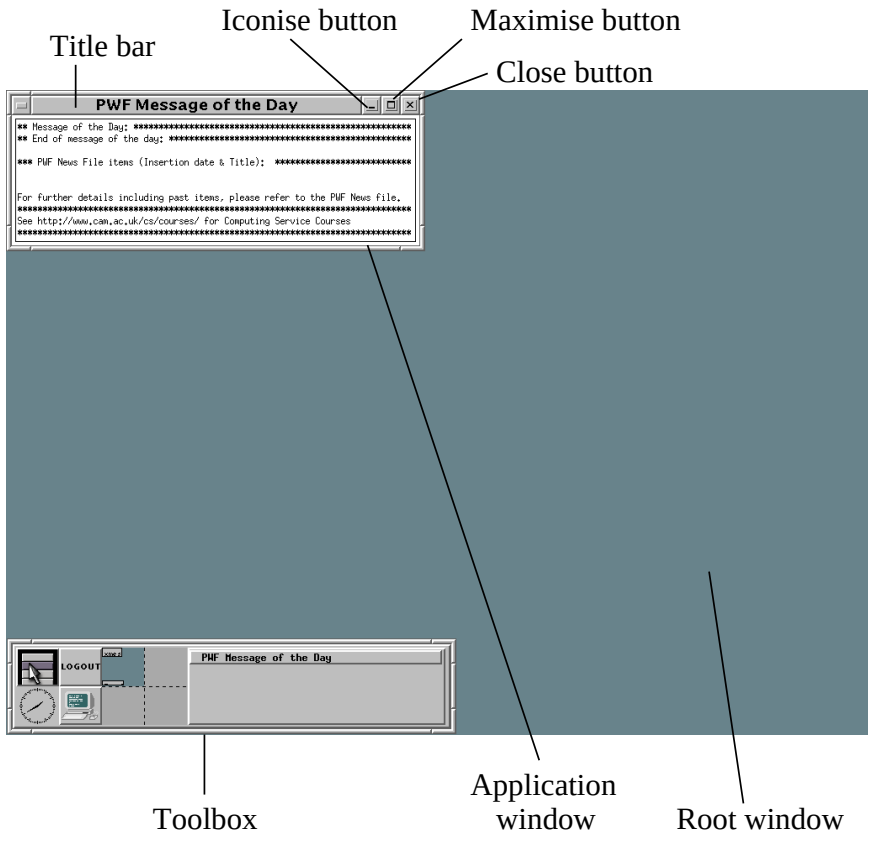
These exercises are not meant to be prescriptive. They are merely intended to stimulate ideas and to save you from discussing the weather with the demonstrators. They should enable you to gain some basic familiarity with UNIX.

Logging in

Firstly, and most importantly, try logging in under Linux. The procedure is:

- If the computer is running Windows, press {Ctrl}{Alt}{Delete} simultaneously, click OK to the RIP warning, choose 'Shutdown...', 'Shutdown and Restart', 'OK'.
- If computer is turned off, turn on!
- When presented with the '`OS Loader`' menu, you have thirty seconds to press the ↓ cursor key before the default action of booting Windows occurs.
- After about a minute, a graphical screen appears with another RIP warning. Click on 'Login to PWF-Linux'
- Finally one is presented with a standard login screen. Remember that both user ids and passwords are case-sensitive under UNIX (don't leave caps lock on!), and login with your usual PWF user id and password.

After a few more moments one should be presented with the standard RedHat Linux 'desktop'.



One of the first things to notice is the PWF 'message of the day'. This is actually an application called `xmessage`, and, once read, it is best removed from the screen by clicking on its close button.

To do anything useful, it is necessary to launch an xterm so that one has a shell (or comand interpreter) into which to type commands. This can be done by:

- Pressing the  icon in the toolbox.
- Pressing the  icon in the toolbox, and choosing 'UNIX shell' from the resulting menu.
- Clicking the right mouse button anywhere on the root window, and choosing 'UNIX shell' from the resulting menu.

Try all three!

Notice that an xterm, like any other application, can be iconised by clicking on the iconise button found in the cluster of three buttons at the top right of the window. To restore an iconised application, either click on its name in the list of running applications on the right of the toolbox, or click the right mouse button anywhere on the desktop, and choose the application from the menu of running applications which will appear. Try both methods.

Editors

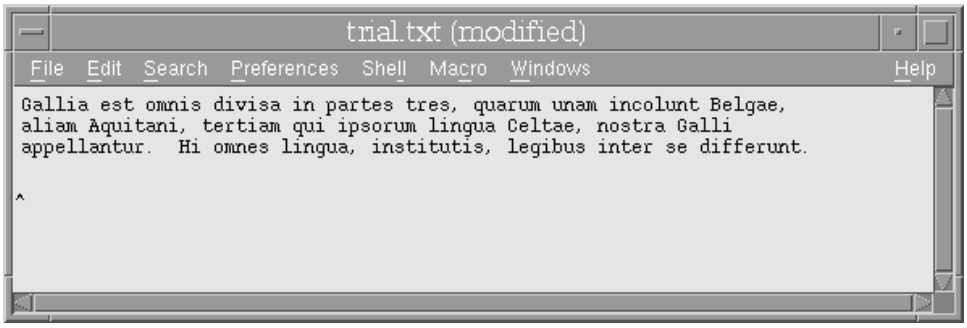
Two editors, nedit and emacs, are discussed in some detail. One is free to use whichever editor one likes. Masochists will inevitably use `vi`, nostalgic Cambridge-educated masochists might use `ne` (it is based on an editor which used to run on Phoenix, the University's mainframe from c 1971 to c 1993), but neither is recommended. Using `pico` or a word-processor are both strongly disrecommended. (Both will wrap long lines unless one is careful, and Fortran will not like this.)

Being the simpler (well, more similar to Windows applications), we shall first consider nedit. Nedit can be started by typing:

```
pcphy001:SPQR1$ nedit trial.txt &
```

If the file does not exist, it will immediately complain that it cannot open it, and offer the choice of creating it, cancelling (keep nedit with an unnamed file), or exiting from nedit. Choose create.

Text can now be typed in freely, but remember one has to press {Enter} (the key marked ↩) whenever one wishes to start a new line. Try typing in some text.



Notice that the titlebar acquired the text 'modified' as soon as you started typing. Now save this by clicking on the 'File' menu and selecting 'Save'. Now that what is shown has been safely saved to disk, the 'modified' disappears from the title bar. Add a little more (e.g. 'Gallos ab Aquitanis Garumna flumen, a Belgis Matrona et Sequana dividit.') and notice that the 'modified' returns to the title bar.

Move back to the beginning of the text, either by using the cursor keys, or by clicking the mouse at the beginning, and add a title (e.g. 'De Bello Gallico, Liber I.') and a blank line under it. Save the result, either as before, or simply press {Ctrl}'S' as the short-cut for saving, and notice 'modified' disappear again.

Return your attention to the xterm, and return the computer's attention to it by moving the mouse into it. If it is partially obscured by the nedit window, click on its titlebar to cause it to move on top of the nedit window. Type `ls` and observe that a file called `trial.txt` now exists. Type:

```
ls -l trial.txt
```

and observe that this file is owned by you, was last modified very recently, and is about 300 bytes long.

Make an extra copy of this file by typing

```
cp trial.txt liber_I.txt
```

Remember that after typing '`tri`' one can simple press {TAB} (the key next to 'Q' marked by two arrows) and the rest of the name will be filled in (*completed*) automatically if it is unique. Warning: copy (`cp`) will not produce any warning if used to overwrite the destination file.

Look at the contents of `liber_I.txt` by typing `less liber_I.txt`. The `less` command will display output one screenful at a time (that is, it will *page* the output). Here there are only a few lines, but one still must press 'q' to exit from `less` and return to the shell.

Return to the nedit window, and using the cursor keys, {Enter} and the delete key, reformat the text to use shorter lines, breaking the first line after 'tres,' the second after 'Aquitani.' and the others at similar lengths. There should now be about ten lines in total.

Resize the nedit window so that it is shorter than ten lines tall. Notice that the scrollbar on the right of the window now indicates what proportion of the total text is visible on the screen. The scrollbar can be dragged around with the mouse, or moved by clicking on the two arrows at each end of it.

The keys marked {Home} and {End} move to the beginning and end of a line, and in combination with {Ctrl} move to the beginning and end of the file: just like Windows.

Try choosing 'New' from the 'File' menu. A new window appears in which one can type a new file:

```
The Gallic War, Book I

All Gaul was divided into three parts,
```

```
of which in one lived the Belgae, in the other
the Aquitani, and in the third those who in
their language are called Celts, and in ours
Gauls. These all differed from each other in
language, institutions and laws.

The River Garumna divided the Gauls from the Aquitani,
and the Matrona and Sequana the Gauls from the Belgae.
```

Trying to save in this window results in being prompted for a filename. This reveals the full path name as being horribly long (starting `/servers/spqr1/PWF-HOME-KZ/`) but fortunately we need not worry about such details. Merely add a name (e.g. book_1.txt) to the end of the path in the `Save File As` box, and press `OK`.

Of course one can do simple searching and replacing. Try moving the cursor to the beginning of the file, choosing from the '`Search`' menu '`Replace`' and replacing 'Belgae' with 'Belgiums'.

What do the underlines mean? They hint at an alternative way of accessing the menus. Press the {Alt} key to the left of the space bar, and with it held down, press 'S'. The Search menu immediately appears. With this menu present, simply pressing 'R' produces the 'Replace' *dialog box*.

Finally, notice that the menu item for Replace reads:
`Replace [Shift]Ctrl+R`
If one simply presses {Shift}{Ctrl} and 'R' simultaneously at any point whilst typing in nedit, the 'Replace' dialog box will appear.

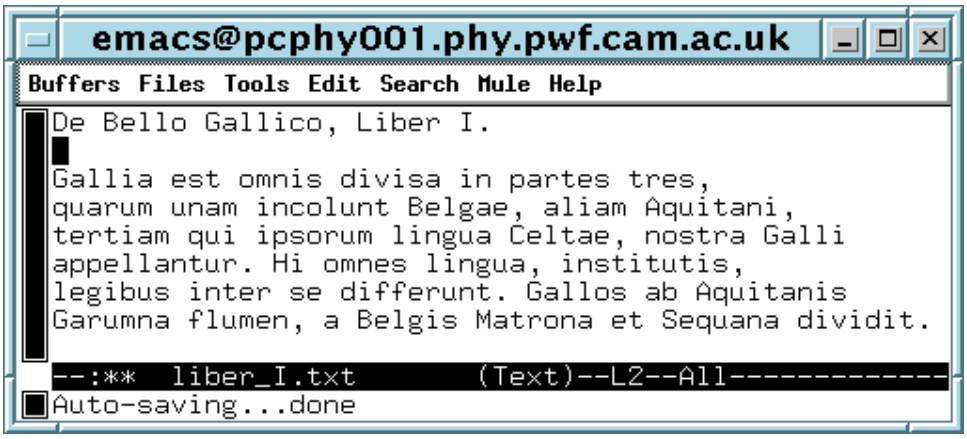
The following features of nedit are probably worth investigating. From the `File` menu: New, Open, Close, Save, Save as, Revert to saved (i.e. discard all changes you have made since last saving the file). From the `Edit` menu:

Undo and Redo (i.e. undo and redo the last changes made whilst editing. One can undo many times: try it.). From the Search menu Find, Find again, Goto line number.

emacs

Emacs too can be started by specifying a non-existent filename on the command-line. Unlike nedit, it assumes that you want the file created automatically if it does not exist.

```
pcphy001:SPQR1$ emacs liber_I.txt &
```



Again one can simply type text into the resulting window, and the cursor keys move around in the expected fashion. However, in emacs {Ctrl}'A' and {Ctrl}'E' move to the beginning and end of the current line, whereas {Home} and {End} move to the beginning and end of the whole document. Notice that emacs continuously displays the current line number at the bottom of its window. In nedit one has to go to the Preferences menu and choose 'Statistics line' or 'Show line numbers' in order to track line numbers.

One can save the text by clicking on the 'Files' menu, followed by the 'Save Buffer' item. The bottom line of the emacs window will then show that emacs has successfully saved the file.

A search and replace operation can be achieved by choosing ‘Query Replace’ from the ‘Search’ menu. The prompt for the text for which to search will appear in the bottom line of the emacs window. Type in the text to change, press {Enter}, type the new text, press {Enter} again, and then for each occurrence which is indicated press ‘y’ to replace, ‘n’ to skip, ‘q’ to quit, or ‘!’ to replace all others with no further prompting.

One finds the expected menu items for open, close (kill current buffer), save, save as, undo, search, replace. Many have shortcut key combinations specified beside them, in which ‘C’ means {Ctrl} and ‘M’ (meta) means {Alt}. Thus pressing {Alt}‘%’ (i.e. {Alt}{Shift}‘5’ on most keyboards) gives one the replace function. Similarly save is {Ctrl}‘X’ followed by {Ctrl}‘S’, and exit is {Ctrl}‘X’ followed by {Ctrl}‘C’. Unlike nedit, emacs has no explicit redo to undo an undo. However, undo, followed by moving the cursor, followed by undo, has the required effect.

A useful keystroke for cancelling a partially-requested command is {Ctrl} ‘G’. E.g., if you type {Alt} ‘%’ and then change your mind about wanting to replace anything, simply type {Ctrl} ‘G’.

Emacs’ search facility is particularly friendly. Move the cursor to the beginning of the document and press {Ctrl}‘S’ to start a search. The prompt at the bottom of the window changes to ‘`I-search:`’. Type ‘i’, and notice the cursor immediate moves to the first ‘i’ in the document. Type ‘n’, and notice the cursor move to the first ‘in’, ‘t’ and to the first ‘int’. Click in the main text window, press a cursor key to stop the search, or press {Ctrl}‘G’ to exit the search and return the cursor to its original position.

Emacs has an extensive help system, which can be entered by pressing {Ctrl} ‘H’. If the window in which you are working becomes split by the presence of a help window, click with the mouse in the window you wish to keep and type {Ctrl} ‘X’ ‘1’ (digit one, not letter l).

O'Reilly sells a 530 page book on emacs. Do not try to understand the whole of this program: it will take an unreasonable amount of time and effort! Understand sufficient to do simple editing, if you wish to use emacs rather than nedit.

Odd files

Both editors have a habit of creating backup files and autosave files as they are used. These tend to have names starting or ending with `#` or `~`. One can usually ignore these. It is best not to delete them whilst the editor is running.

The xterm again

By now you should have made some considerable use of the xterm: a couple of editors launched, a file copied and viewed with `less`. Try a few more simple file operations: can you create a directory using `mkdir` and then copy the files you have been editing into it? Take care:

```
pcphy001:SPQR1$ cp liber_I.txt book_I.txt
```

will overwrite `book_I.txt` with `liber_I.txt`, whereas

```
pcphy001:SPQR1$ mkdir Caesar
pcphy001:SPQR1$ cp liber_I.txt book_I.txt Caesar
```

will copy both files into the directory called `Caesar`. (If more than two *arguments* (names) are given to `cp`, the last must be a directory). Delete the original copies using `rm`.

If you wish to see commands now scrolled off the top of the xterm, a right mouse click on the scroll bar at the left scrolls backwards: near the top for a

small amount, half way down for half a screenfull, and the bottom for a full screenfull. Likewise a left mouse click scrolls forwards again. The middle button if clicked moves the shaded part of the scrollbar (representing the text visible) to that point in the whole scrollbar (which represents all the text remembered). Pressing {Shift} together with {Page Up} or {Page Down} also causes the xterm to scroll back and forth by a page at a time.

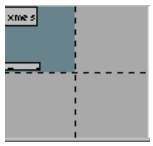
Holding down the {Ctrl} key whilst pressing the right mouse button anywhere in an xterm's window brings up a list of font sizes from which one may choose. The {Ctrl} with middle button brings up a menu of options, such as removing the scroll bar.

Netscape

Netscape is a remarkably poorly understood program. Did you know that if you click on a link with the middle button (rather than the left), a new window will open with the contents of that link? Or that pressing {Shift} whilst clicking with the left button on a link will download the target of the link as a file? (Very useful for downloading example programs.)

Virtual Desktops

Try clicking on one of the three grey rectangles in the *pager*, the virtual desktop representation in the toolbox:



Notice that all open windows disappear from the screen. Open a couple of new windows, and now click again on the now-grey top left rectangle in the pager. Notice that the old set of windows reappears. One can switch

between these virtual desktops at will, either by clicking on the pager, or by moving the mouse off the screen in the direction of the desktop to which you wish to move.

What use are they? Well, some like to have an email client on one desktop, a web browser on another, and to be doing their exercises on a third, to which they quickly switch when they think a demonstrator is looking. Others might use them to give them more space when simultaneously viewing on-line manuals and writing programs. Yet others dislike them and don't use them at all.

Window manager fun

Start both `xload` and `xeyes`. Remember the '`&`' on the end of the command line so that the command prompt returns immediately. Try resizing both. It *is* possible to resize xeyes: ask if you can't work out how. Try killing both: use the close button on the top right of their window, use the 'close' or 'delete' option from the menu found by clicking on the button on the top left of their window. Type '`ps x`' to find the process id (PID) of the process you wish to kill, and kill it using the `kill` command.

> If you want to create some 'fake' activity to register on `xload`, type (precisely)
>
> ```
> pcphy001:SPQR1$ perl -e 'while(){}' &
> ```
>
> which does nothing in an endless fashion. Do not ask for a detailed explanation of this command, or feel obliged to understand it: it uses `perl`, which is a language of similar complexity to Fortran... Remember to kill the process when you are bored with it, unless you want your computer to run at half speed!

Redirection

Try some simple examples of redirection, such as the those given on page 27. The Unix command '`echo`' simply echoes back its arguments:

```
pcphy001:SPQR1$ echo Hello
Hello
pcphy001:SPQR1$
```

Can you understand why very short files are easily created by:

```
pcphy001:SPQR1$ echo Remember to do supervision work > to_do.lst
```

Type '`echo *`'. Is this what you expected? Try to understand why it did that.

The on-line manual

Try looking at a few pages from the on-line manual, by trying

```
pcphy001:SPQR1$ man wc
```

or maybe `man ls`. There is also a rather basic keyword search facility.

```
pcphy001:SPQR1$ man -k time
```

This simply searches for fixed strings in the 'description' line of the man page, and produces an alarming amount of output.

The on-line manual is divided into 8 sections, and all common user commands are in section 1. Other sections describe system libraries and calls for C programmers, file formats and utilities for system administrators, and other things we generally do not want. As the section number is given in the output of `man -k`, we can achieve this with:

```
pcphy001:SPQR1$ man -k time | grep 1
```

The `grep` command will show lines containing the given text. One can even try:

```
pcphy001:SPQR1$ man -k time | grep 1 | less
```

The | character is often called 'pipe' – can you see why? Notice in the output one multimedia command, found because of multimedia, one command called `date`, which does indeed give the current time, the `sleep` command which we have already met, and other commands such as `uptime`.

Dot files

Unix programs often use files beginning with a period for their configuration. The default shell, called `bash`, uses one called `.bashrc` from which it reads commands which it executes every time it starts (and hence every time an xterm is started).

Ensure you have an xterm free, and try creating a file in your home directory called '`.bashrc`' and containing the following three lines:

```
alias rm='rm -i'
alias mv='mv -i'
alias cp='cp -i'
```

(Note that all the quote mark characters are identical: they are the single closing quote, or apostrophe, which is probably found on the right-hand side of your keyboard, and is certainly not the top left key.)

Before doing anything else, check that you can still start an xterm, and thus a new shell. If you make a bad mistake in this file, you will not be able

to start any more shells until you correct it, which could be an interesting catch-22. As it is, the free xterm you started before creating this file can be used to delete it if things go wrong!

What does the file do? The `alias` command means that every time you type `rm`, the shell will behave as though you had typed `rm -i`, and will thus prompt you before deleting the file. Similarly `cp` and `mv` will now prompt before overwriting files. If you like this behaviour, leave it, if not, delete it! One can create completely new commands in this fashion, e.g.

```
alias hermes='ssh hermes.cam.ac.uk'
```

Logging out

One should always remember to log out when leaving a computer. This enables your honest colleagues to know that the computer is now free for their use, and prevents your dishonest colleagues from doing nasty things with your account.

UNIX Command Summary

`a2ps` *file*	Print text file, two pages per sheet
`cal` *month year*	Show calendar (use four digit year!)
`cd`	Change current directory to home directory
`cd` *dir*	Change current directory to *dir*
`chmod go=` *file*	Prevent others from reading *file*
`cp` *file1 file2*	Copy *file1* to *file2*, overwriting *file2* if it exists
`cp` *files... dir*	Copy multiple files to a directory
`date`	Show date and time
`du -sk` *dirs...*	Show disk usage of *dirs*
`echo` *text*	Repeats its arguments
`file` *file*	Guess file type
`kill` *pid*	Ask process to exit
`kill -KILL` *pid*	Cause process to be killed
`less` *file*	View a file, page at a time. Keystrokes include: q - exit, ↑, ↓ up and down one line b - back one page, {space} - next page G - end of file, 1G - beginning of file
`lpr -h`	List possible printers (a PWF-specific command)
`lpr` *file*	Print *file* (text or Postscript) to default printer
`lpr -P`*printer file*	Ditto, to named *printer*
`ls`	List contents of directory
`man` *command*	On-line manual for *command*
`man -k` *keyword*	Search on-line manual for *keyword*
`mkdir` *dir*	Make directory
`mv` *file1 file2*	Move (rename) *file1*, deleting *file2* if it exists
`mv` *files... dir*	Move multiple files to a directory
`passwd`	Change password
`ps`	List processes
`ps x`	List all your processes
`quota`	Show filespace quota
`rm` *files...*	Delete files
`rm -i` *files...*	Delete files, prompting before each one
`rmdir` *dir*	Remove directory (if empty)
`top`	View process activity. Press q to quit.
`wc` *file*	Count lines and words in text file
`xload`	A CPU activity monitor
`xterm`	A command shell in a window