**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, and type [**q**] if you want to quit reading. As you can see, **less** is used in preference to **cat** for long files

### 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**, 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.

### Simple searching using grep

**grep** is one of many standard UNIX utilities. It searches files for specified words or patterns. First clear the screen, then type

To ignore upper/lower case distinctions, use the -i option, i.e. type

**% grep -i science science.txt**

## **3.1 Redirection**

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 specifing a file to read

**% cat**

Then type a few words on the keyboard and press the [**Return**] 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 specifing 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.

## **3.2 Redirecting the Output**

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 [**Return**] after each one.

**pear  
banana  
apple  
^D {this means press [Ctrl] and [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**

### 3.2.1 Appending to a file

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**

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 outputing the text to the file **biglist**

To read the contents of the new file, type

**% cat biglist**

## **3.3 Redirecting the Input**

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 animals. Press [Return] after each one.

**dog  
cat  
bird  
ape  
^D (control d to stop)**

The output will be

**ape  
bird   
cat   
dog**

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**

## **3.4 Pipes**

To see who is on the system with you, type

**% who**

One method to get a sorted list of names is to type,

**% who > names.txt  
% sort < names.txt**

This is a bit slow and you have to remember to remove the temporary file called names when you have finished. What you really want to do is connect the output of the who command directly to the input of the sort command. This is exactly what pipes do. The symbol for a pipe is the vertical bar |

For example, typing

**% who | sort**

will give the same result as above, but quicker and cleaner.

To find out how many users are logged on, type

**% who | wc -l**

### Exercise 3b

Using pipes, display all lines of **list1** and **list2** containing the letter 'p', and sort the result.

[Answer available here](http://www.ee.surrey.ac.uk/Teaching/Unix/pipeanswer.html)

## **Summary**

|  |  |
| --- | --- |
| **Command** | **Meaning** |
| **command > file** | redirect standard output to a file |
| **command >> file** | append standard output to a file |
| **command < file** | redirect standard input from a file |
| **command1 | command2** | pipe the output of command1 to the input of command2 |
| **cat file1 file2 > file0** | concatenate file1 and file2 to file0 |
| **sort** | sort data |
| **who** | list users currently logged in |