Pipes & Pipelines

Input redirection gets input from a file and output redirection sends its data to another file. Pipes, however, redirect the **output** of one program so that it acts as the input **of another program**. The pipe character is the **vertical bar**. Several pipe commands is called **a pipeline**.

The Unix ls command shows the files of the **current directory** on standard output.

```
$ ls
err.txt filter.cpp input.txt moby.txt output.txt
```

Of course you can save the directory listing to a file using output redirection:

```
$ ls > files.txt
```

However, instead of saving it, we can **pipe the output** to the `wc` (**word count**) filter, adding a command-line switch `-l`, to indicate that we only want to count the number of lines. Try it yourself and see what happens.

```
$ ls | wc -l
```

Here is another pipeline which lists the current directory, and then **sorts the output** in reverse order, sending that output to the screen.

```
$ ls | sort -r
```

One of the most useful Unix filters is `grep` (which stands for the mouthful "global regular expression parser"). While quite complicated, especially when used with **regular expressions**, it is easy to use for searching through text to find a particular word.

Let's find out, for instance, on which lines the name **Ishmael** is used in Moby Dick.

```
$ cat < moby.txt | grep "Ishmael" -n
```

And how many lines are there??

```
$ cat < moby.txt | grep "Ishmael" -n | wc -l
```

This course content is offered under a CC Attribution Non-Commercial license. Content in this course can be considered under this license unless otherwise noted.