

Working With Files





less

The less command displays the contents of a file, one page at a time. We can navigate forwards and backwards through the file, which is especially useful with very large files.

`less somefile.txt` will display the contents of somefile.txt using less.





less navigation

When viewing a file using less...

- press **space** or **f** to go to the next page of the file
- press **b** to go back to the previous page
- press **Enter** or **Down arrow** to scroll by one line
- to search, type forward slash **/** followed by a pattern
- press **q** to quit





cat

The cat command **concatenates** and prints the contents of files.

cat <filename> will read the contents of a file and print them out. For example, **cat instructions.txt** will read in from the instructions.txt file and then print the contents out to the screen.

A dark blue terminal window with three colored window control buttons (red, yellow, green) in the top left corner.

```
> cat <filename>
```





cat cont'd

If we provide cat with multiple files, it will concatenate their contents and output them.

`cat peanutbutter.js jelly.css` will output peanutbutter.js first and immediately after print the contents of jelly.css

A dark-themed terminal window with three colored window control buttons (red, yellow, green) in the top-left corner. A light blue prompt character is visible before the command.

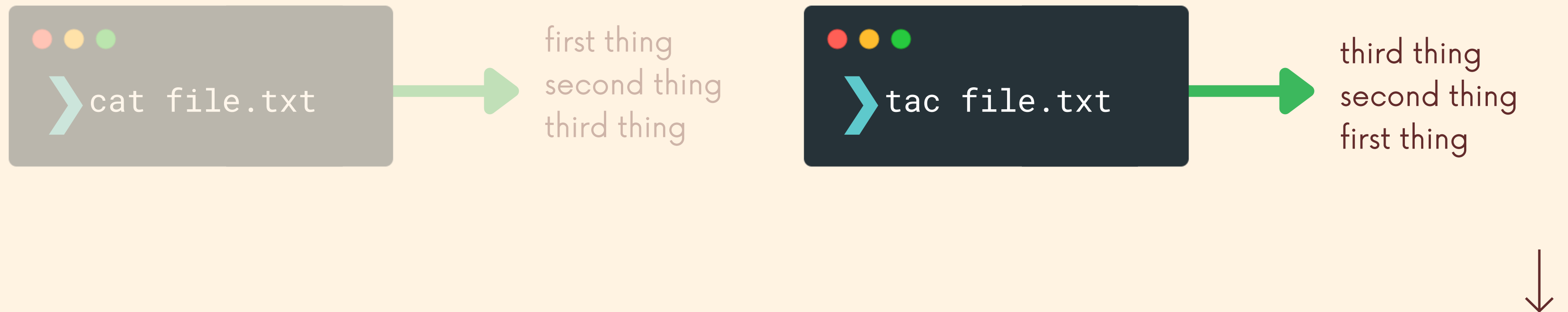
```
>cat <file1> <file2>
```





tac

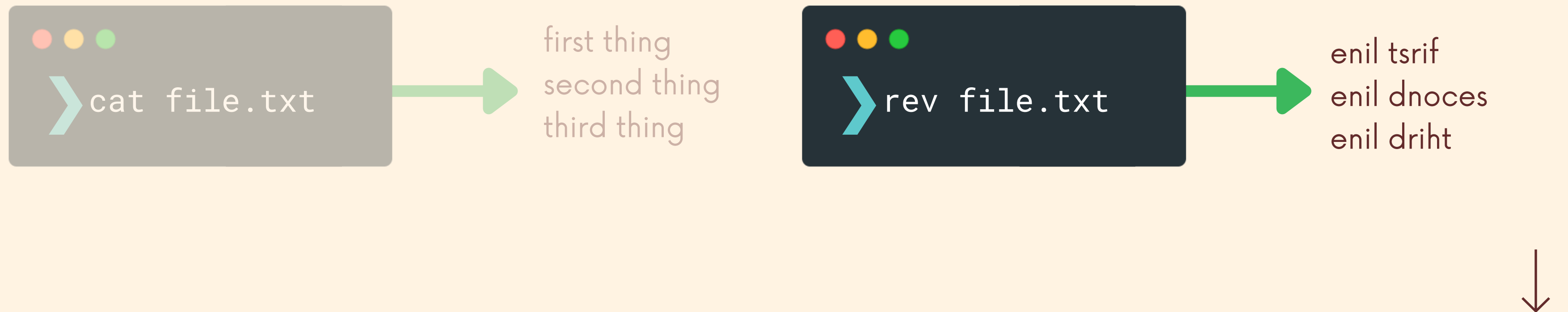
tac (cat spelled backwards) will concatenate and print files in reverse. It prints each line of a file, starting with the last line. You can think of it as printing in reverse "vertically"





rev

the **rev** command prints the contents of a file, reversing the order of each line. Think of it as a "horizontal" reverse, whereas tac is a "vertical" reverse.





head

The head command prints a portion of a file, starting from the beginning of the file. By default, it prints the first 10 lines of a file.

`head warAndPeace.txt` would print the first 10 lines of the warAndPeace.txt file

A dark blue terminal window with three colored circles (red, yellow, green) in the top left corner. A light blue prompt character is followed by the command `head filename.txt`.

```
> head filename.txt
```






head contd

We can also specify a number of lines for head to print using the `-n` option (or `--lines`) followed by an integer.

`head -n 21 warAndPeace.txt` would print the first 21 lines of the warAndPeace.txt file

We can also use an even shorter syntax to specify a number of lines: `head -3 filename.txt` will print the first 3 lines of the file.



```
> head -n 13 filename.txt
```





head contd contd

We can also provide a number of bytes to print out, rather than lines using the `-c` option.

`head -c 8 warAndPeace.txt` would print the first 8 bytes of the warAndPeace.txt file.

A dark-themed terminal window with three colored window control buttons (red, yellow, green) in the top-left corner. A light blue prompt character is visible on the left.

```
> head -c 8 filename.txt
```





tail

The tail command works similarly to the head command, except it prints from the END of a file. By default, it prints the last 10 lines of a file.

`tail warAndPeace.txt` would print the last 10 lines of the warAndPeace.txt file

The same `-n` and `-c` options we saw with head also work with the tail command.

A dark blue terminal window with three colored dots (red, yellow, green) in the top left corner.

```
> tail filename.txt
```





WC

The word count command can tell us the number of words, lines, or bytes in files. By default, it prints out three numbers: the lines, words, and bytes in a file.

We can use the **-l** option to limit the output to the number of lines.

The **-w** option limits the output to the number of words in the file.

A dark blue terminal window with three colored circles (red, yellow, green) in the top left corner.

```
> wc -l students.txt
```





sort

The sort command outputs the sorted contents of a file (it does not change the file itself). By default, it will sort the lines of a file alphabetically.

`sort names.txt` would print each line from names.txt, sorted in alphabetical order.

A dark blue terminal window with three colored circles (red, yellow, green) in the top left corner.

```
> sort filename.txt
```





sort cont'd

The **-r** option tells the sort command to sort in reverse order.

sort names.txt -r would print each line from names.txt, sorted in REVERSE alphabetical order.

A dark blue terminal window with three colored circles (red, yellow, green) in the top left corner. A light blue prompt character is visible.

```
> sort -r filename.txt
```





sorting numerically

The **-n** option tells the sort command to sort using numerical order.

sort -n prices.txt would print each line from names.txt, sorted in numerical order.

We could also reverse it with **sort -nr prices.txt**

A dark blue terminal window with three colored circles (red, yellow, green) in the top-left corner.

```
> sort -n prices.txt
```





uniques only

The **-u** option tells the sort command to ignore duplicates and instead only sort unique values

A dark blue terminal window with rounded corners. At the top left, there are three colored circles: red, yellow, and green. A light blue prompt character, resembling a right-pointing arrow, is positioned to the left of the command text.

```
> sort -u prices.txt
```





sorting by field

We can specify a particular "column" that we want to sort by, using the **-k** option followed by a field number.

In this example, **sort data.txt -nk 2** tells sort to use the numeric sort and to sort using the 2nd field.

```
pencil 0.50  
flowers 9.99  
pie 4.99  
soda 0.99
```



```
pencil 0.50  
soda 0.99  
pie 4.99  
flowers 9.99
```

```
> sort data.txt -nk 2
```

