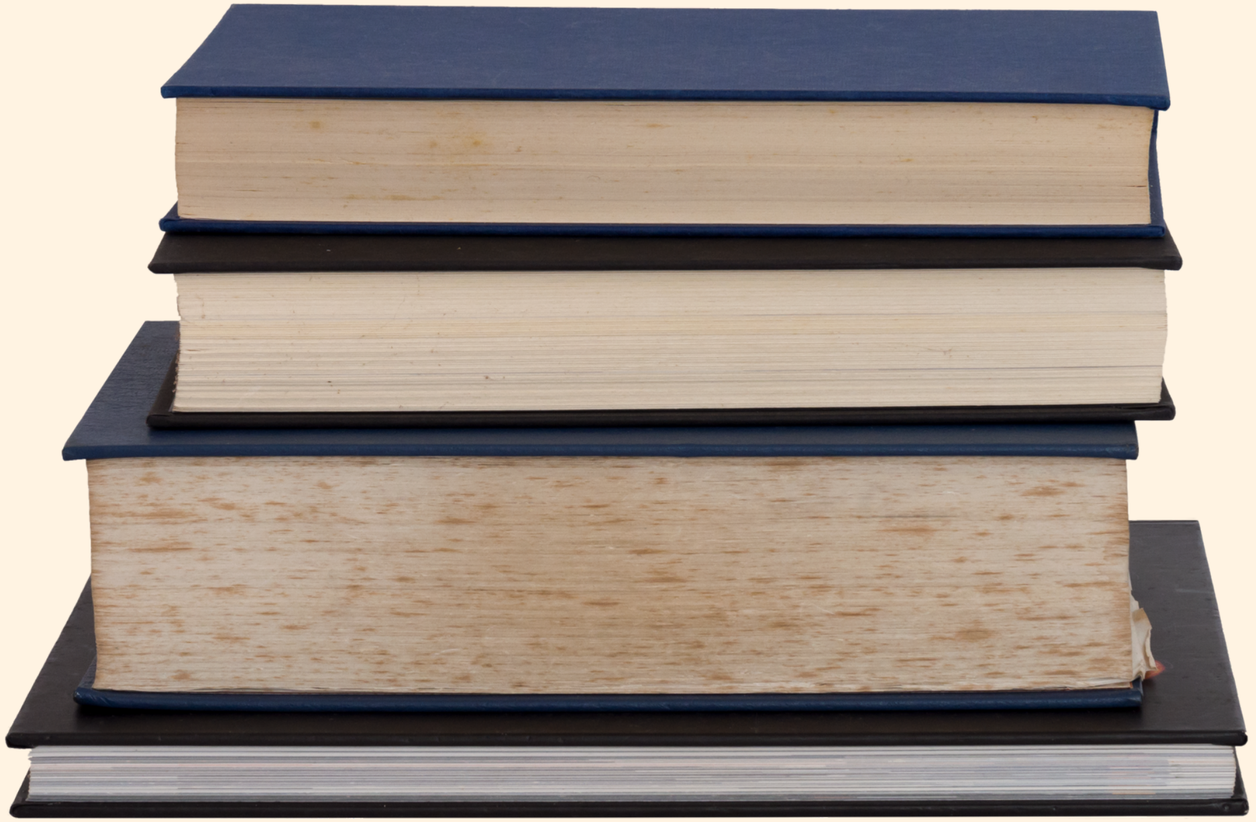
Grep



Grep

The **grep** command searches for patterns in each file's contents. Grep will print each line that matches a pattern we provide.

For example, grep "chicken" animals.txt will print each line from the animals.txt file that contains the pattern "chicken"

```
❯ grep PATTERN FILE
```

Case Insensitive

Use the **-i** option with grep to make the search case insensitive.

grep -i "Chapter" book.txt will print all matching lines from the book.txt file that contain the word "chapter" in any casing.

```
❯grep -i "Chapter" book.txt
```

Word Search

Use the **-w** option to ensure that grep only matches words, rather than fragments located inside of other words. A word is defined by non-word characters on either side (start of line, spaces, end of line, punctuation, etc.)

```
❯grep -w "cat" book.txt
```

grep -w "cat" book.txt would match cat but not catheter!

Recursive Search

Use the **-r** option to perform a recursive search which will include all files under a directory, subdirectories and their files, and so on.

If we don't specify a starting directory, grep will search the current working directory.

grep -r "chicken" will search the current working directory and any nested directories for lines that contain "chicken"

```
❯grep -r "chicken"
```

Regex Crash Course

We can provide regular expressions to **grep.** Regular expressions helps us match complex patterns, BUT the syntax does differ from what we've seen so far.

. - matches any single character
^ - matches the start of a line
$ - matches the end of a line
[abc] - matches any character in the set
[^abc] - matches any char NOT in set
[A-Z] - matches characters in a range
* - repeat previous expression 0 or more times
\ - escape meta-characters

Grep

This example matches a string that contains a digit 1-9 (not 0), followed by any 4 characters.

```
❯ grep '[1-9]....' prices.txt
$95.99
$30.75
```

```
$95.99
$30.75
$9.99
$0.50
$2.50
$0.99
$0.75
```

Grep -c

The **-c** option tells grep to print the number of matches instead of printing the actual matches

```
$95.99
$30.75
$9.99
$0.50
$2.50
$0.99
$0.75
```

```
❯ grep -c "\$[1-9]" prices.txt
4
```

Grep -o

The -o option tells grep to only print out the matches, rather than the entire line containing each match.

```
❯ grep -o "\$[1-9]" prices.txt
$9
$3
$9
$2
```

```
$95.99
$30.75
$9.99
$0.50
$2.50
$0.99
$0.75
```

Piping To Grep

A common use case is to use **grep** to whittle down or filter a large chunk of data.

In this example, the **ps -aux** command will output a huge list of all processes running on our machine. We pipe that data to grep, and then filter it down to only the processes that include "hermione"

In effect, this command lets us see what hermione is up to!

```
❯ps -aux | grep hermione
```

Piping To Grep

In this example, we are getting the man page for grep and then piping that to the actual grep command, where we search for the string "count". Basically, it's a weird way of searching the man pages.

```
❯man grep | grep "count"
```