Exploring Text Files

Exploring Text Files

Doing Something at the Command Line

Just navigating around the command line is boring. Let's learn how to *do* some things.

Looking at Text Files

First, I'll download *Pride and Prejudice* from Project Gutenberg.

```
wget https://www.gutenberg.org/files/1342/1342-0.txt
mv 1342-0.txt PandP.txt
```

Since this is a plain text document, we can use the command line programs for text here. We'll begin with `less`.

`less`: Read, but not write a document

The program `less` lets you read a text document. This is useful for double-checking that you're looking at and working with the right text file. To use it, you simply type `less` followed by the name of the document. Here's what my terminal looks like when I type the following at the prompt.

```
less PandP.txt
```

I can scroll up and down in the document with the up and down arrows, or hit spacebar to jump down a page. To get out of viewing the document, just hit `q` and you'll get back to your command line prompt.

`wc`: Get document statistics

The `wc` program calculates statistics about your document, like how many lines, words, and characters there are in the document, and returns it to you. You can run it by typing `wc` followed by the name of the document. Here's how it looks on `site.html`

```
% wc PandP.txt
 14579 124749 798774 PandP.txt
```

What this is telling us is that there are 14,579 lines in `site.html`, 124,749 words, and 798,774 characters.

"Flags"

There are also a few options you can set to change how `wc` works that you set with things called "flags". Flags are used for many command line programs to set certain options, and they usually take the form of a dash, and a single letter placed immediately after the name of the program. For example, the flag `-l` tells the program `wc` to only return the number of lines in the document. Here's how that looks:

```
% wc -l PandP.txt
   14579 PandP.txt
```

There's also a flag to only get back the number of words in a document, `-w`. Here's how that looks:

```
% wc -w PandP.txt
     124749 PandP.txt
```

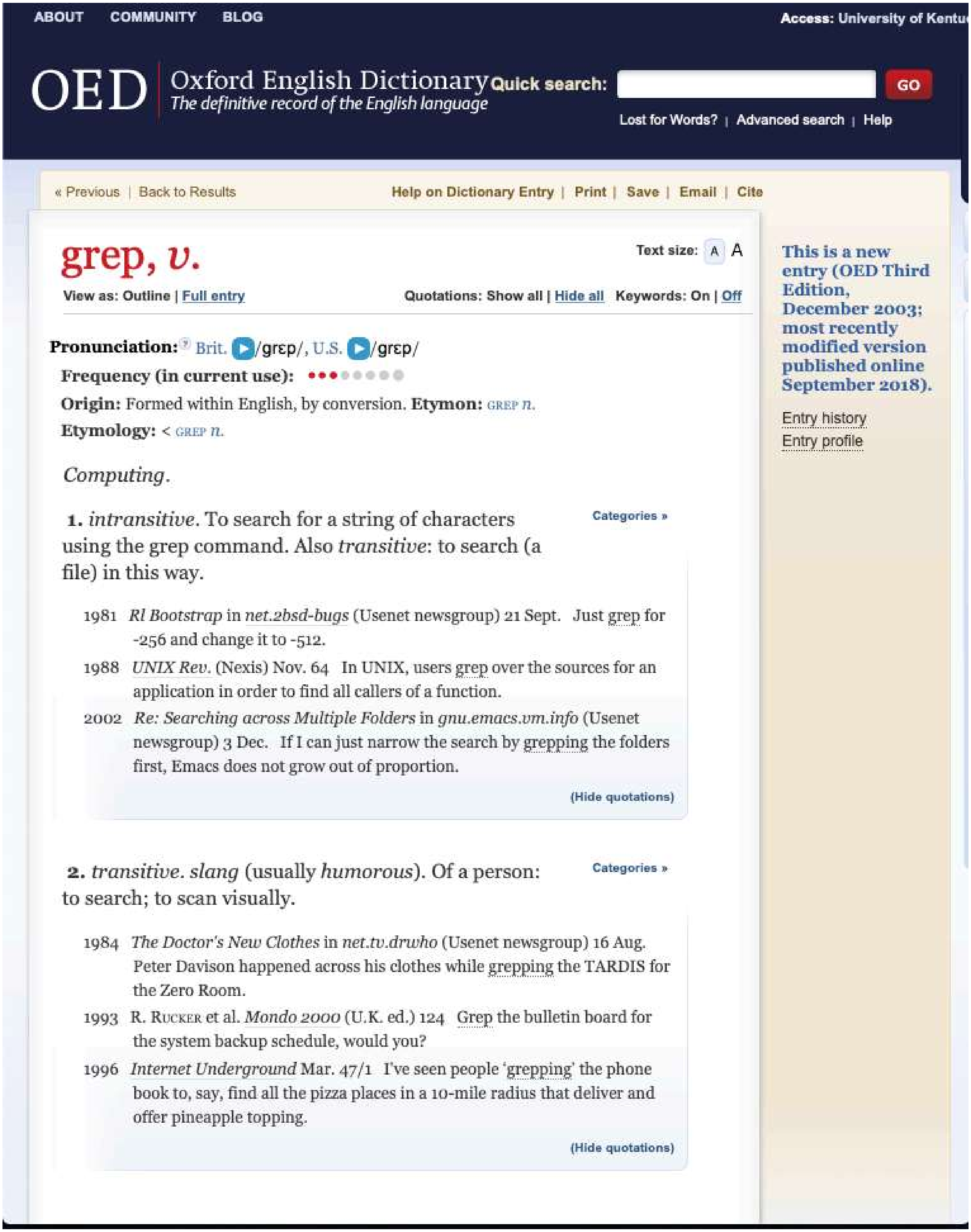
You can pass multiple options to a command line program, just by putting in all the flags you want one after the other. For example, if I wanted to see the number of lines and the number of words in `site.html`, I would do this:

```
% wc -l -w PandP.txt
     14579 124749 PandP.txt
```

Usually, when you look at the man page for a command line program, all of the flags and what they do are listed at the beginning.

`grep`: Search within documents

`grep` is a super useful and commonly used program to search within documents. In fact, in the 1980s and 1990s among certain nerds, the word "grep" was used as a general word for "search", and that made it into the Oxford English Dictionary.


ABOUT
COMMUNITY
BLOG
Access: University of Kentu
OED
Oxford English Dictionary
The definitive record of the English language
Quick search:
GO
Lost for Words? | Advanced search | Help
« Previous | Back to Results
Help on Dictionary Entry | Print | Save | Email | Cite
Text size: A A
View as: Outline | Full entry
Quotations: Show all | Hide all
Keywords: On | Off
Categories »
(Hide quotations)
Entry history
Entry profile

grep, *v.*

Pronunciation: Brit. /grɛp/, U.S. /grɛp/

Frequency (in current use): ●●●○○○○○

Origin: Formed within English, by conversion. **Etymon:** GREP *n.*

Etymology: < GREP *n.*

Computing.

1. *intransitive*. To search for a string of characters using the grep command. Also *transitive*: to search (a file) in this way.

1981 *Rl Bootstrap* in *net.2bsd-bugs* (Usenet newsgroup) 21 Sept. Just grep for -256 and change it to -512.

1988 *UNIX Rev.* (Nexis) Nov. 64 In UNIX, users grep over the sources for an application in order to find all callers of a function.

2002 *Re: Searching across Multiple Folders* in *gnu.emacs.vm.info* (Usenet newsgroup) 3 Dec. If I can just narrow the search by grepping the folders first, Emacs does not grow out of proportion.

2. *transitive*. *slang* (usually *humorous*). Of a person: to search; to scan visually.

1984 *The Doctor's New Clothes* in *net.tv.drwho* (Usenet newsgroup) 16 Aug. Peter Davison happened across his clothes while grepping the TARDIS for the Zero Room.

1993 R. Rucker et al. *Mondo 2000* (U.K. ed.) 124 Grep the bulletin board for the system backup schedule, would you?

1996 *Internet Underground* Mar. 47/1 I've seen people 'grepping' the phone book to, say, find all the pizza places in a 10-mile radius that deliver and offer pineapple topping.

This is a new entry (OED Third Edition, December 2003; most recently modified version published online September 2018).

With `grep`, you type in the string you want to search for in quotes, followed by the file name, and it will print out all of the lines of the file where that string appears. For example, let's say I wanted to find all of the lines in `PandP.txt` where `yes` appears. Here's how I'd do that:

```
% grep "yes" PandP.txt
      and agreeable in your eyes. I never heard you speak ill of a
      an object of some interest in the eyes of his friend. Mr. Darcy
      eyes. To this discovery succeeded some others equally mortifying.
      pleasure which a pair of fine eyes in the face of a pretty woman
      Miss Bingley immediately fixed her eyes on his face, and desired
      worthless in their eyes when opposed to the regimentals of an
      servant waited for an answer. Mrs. Bennet’s eyes sparkled with
```

This will be our first lesson in the fact that computers will do whatever you tell them to, even if it's not what you meant to tell it to do. I really just wanted to find all of the lines with the word `yes`. But `grep` has mostly just shown us lines where `eyes` appears. Why? Because all we asked grep for was lines with `yes`, and there is `yes` in `eyes`!

Figuring out more precise ways to search documents with `grep` is going to be a big topic coming up soon.

`|`: piping commands together

Let's say I didn't want to see all of the lines that had `yes` in them, but I just wanted to know how many *lines* had `yes` on them? Here's the tools we have available to us now:

- `grep`: We can search and get back all of the lines in a document
- `wc`: We can get the number of lines in a document.

We can connect these two functionalities together by "piping" the output of `grep` into `wc` with the symbol `|`. `|` is called a "pipe" and can be found on your keyboard above the Enter key. You may need to hit Shift+\ to find it. Here's how it looks to pipe the output of `grep` into `wc`.

```
% grep "yes" PandP.txt | wc
      83     888    5592
```

Here's what's happening

1. First, `grep` searched `PandP.txt` for every line that had `yes` in it, and returned all of those lines
2. Instead of printing out all of the lines, the pipe, `|`, passed them to `wc`, which counted all of the lines, words, and characters in them.
3. `wc` then printed out the number of lines, words, and characters from the output of `grep`.

Piping like this can be super useful!