Getting Started with regexps in vim

/regexp*ENTER*

- Vi is a great way to learn regexp because it is a main way to move the cursor
- Say where you want to go, don’t try to go there (with mouse or arrows)

- For example, /foo*ENTER* moves to the next foo
- n repeats that search in the same direction
- But /[0-9]*ENTER* moves to the next digit
- And /[Tt]he *ENTER* moves to the next The followed by space, whether upper or lower case
- /^[1-9][0-9]*\.*ENTER* is a good way to find lines that start with a number then a period
  - Note that . matches any character, so if we want a real period we say \.
  - This is backslash dot, or a protected/literal character
  - The . itself is a wildcard or meta character
- /F… *ENTER* is a good way to find four-letter words starting with capital F
  - But note that the space has to be there; this four-letter word cannot end a line

Vim / does not do extended regexps

- So you can do /[A-Z][A-Z]*ENTER* to look for two capital letters adjacent
  - Remember n for next so you can try it again
- You can do /\.$*ENTER* to find period at an end of line
  - Which won't match a space after period
  - But you could do /\. $*ENTER* to find a period and space that end a line
  - And you probably want /\. *$*ENTER* to find period and any number (including 0) spaces after that, that ends the line
- But you can't do /\.{3}*ENTER* to look for three periods in a row
- Nor /A|B*ENTER* to find an A or a B
  - But /[AB]*ENTER* works
- Nor /[0-9]+*ENTER* to find a run of digits
  - But /[0-9][0-9]**ENTER* works

:g/./ s/foo/bar/g*ENTER*

- This powerful idiom says go to each line that has a character (any character, a match for . anywhere in the line, so a line with at least one character)
- And on each such line, substitute any instance of foo with bar, and do all matches (global substitution)
  - Remember u for undo, so you can try it again
- So :g/foo/ s/foo/bar/g*ENTER* does the same thing
  - :g/./ s/foo/bar*ENTER* just replaces the first foo on each line
- :g/^[1-9]/ s/ /\t*ENTER* replaces the first space (if any) with a tab on any line starting with a non-zero digit
  - In related tools, you might have to terminate the \t with the matching /
    - But the delimiter does not have to be /
    - Try :g/./ s; ;\t; where semicolon is used as the delimiter
  - For example, :g/./ s/[\t][\t]*/ /*ENTER* will replace leading whitespace with a single space
  - :s/[aeiou]/x/g*ENTER* replaces vowels with x’s in the current line