Outline

Programming

- Regex

Practicalities

- Quiz review

Cool media art pieces

Prof. Angela Chang

Lecture 9: Regular Expressions

Fall 2017. Oct 4

CODE, CULTURE, AND PRACTICE

REGEX

Regular Expressions: search patterns

language of symbols that describes a text pattern.

Literal matches- strings match exactly
"The cow, camel, and cat communicate" -walk it through and backtracks

Metacharacters-- characters with special meaning

. * + [abc] ^ # ! ? (abc) \d(2,3)

They transform literal characters into powerful expressions. they can have more than one meaning (used in context

http://regexpal.com

Investigating a file of text

Download Pride and Prejudice:

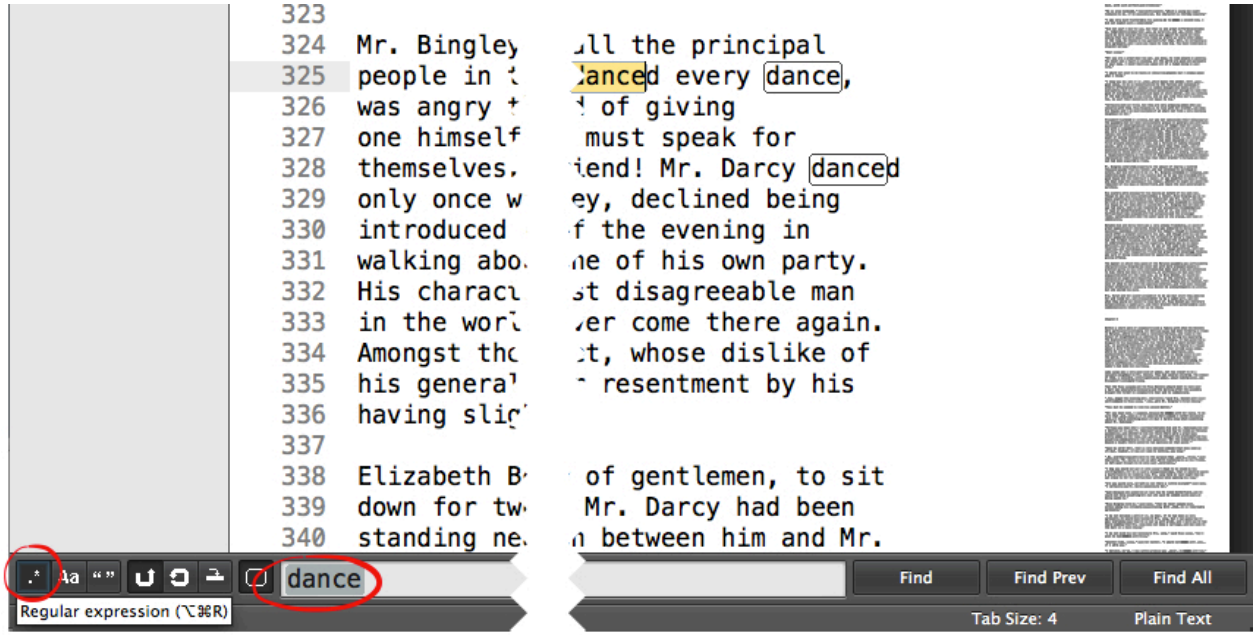
http://www.gutenberg.org/files/1342/1342.txt

you might have to visit this link twice to get the file

Save ASCII version to your computer (e.g. ⌘+**S** on Ctrl+S)

Open it in your text editor (Geany, TextWrangler, Notepad++, Sublime)

Find how many occurences of the pattern ‘dance’ exists (⌘+**F)**

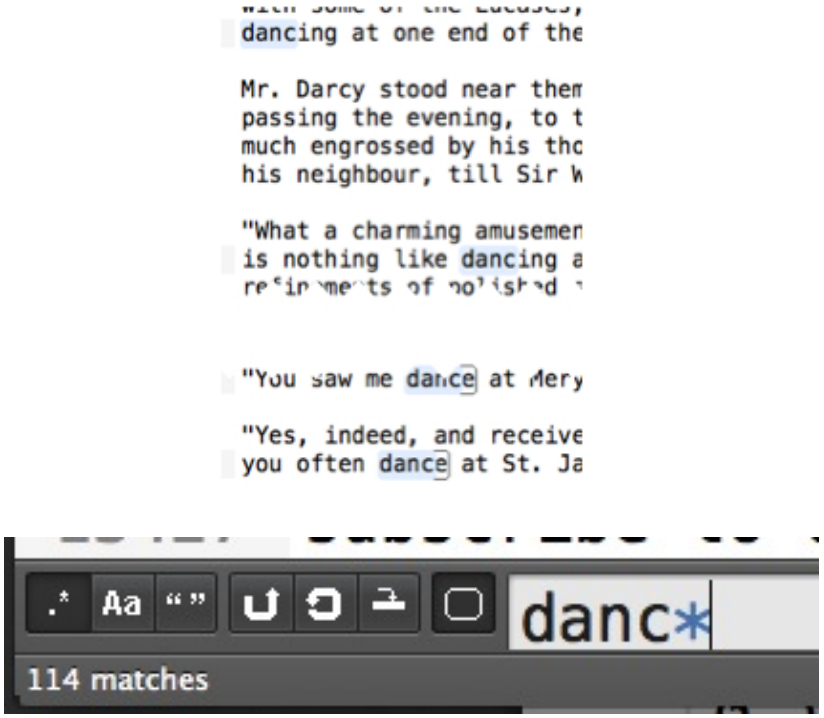


click regex search

64 occourrences of ‘dance’

Wildcard character *

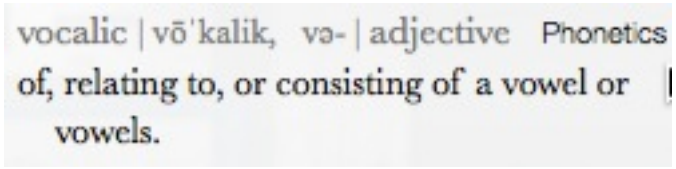
type danc* (asterisk) to get all verb forms and tenses of dance



* means matching 0 or more times – dance, dancing, dances, danced, danc

Simple text analysis

How vocalic is *Pride and Prejudice*?



Character class pattern [ ] - match any characters inside
Define a character class of five vowels: **[aeiou]**

Simple text analysis

How vocalic is *Pride and Prejudice*?

vocalic | vō'kalik, və- | adjective Phonetics
of, relating to, or consisting of a vowel or vowels.

case-sensitive search

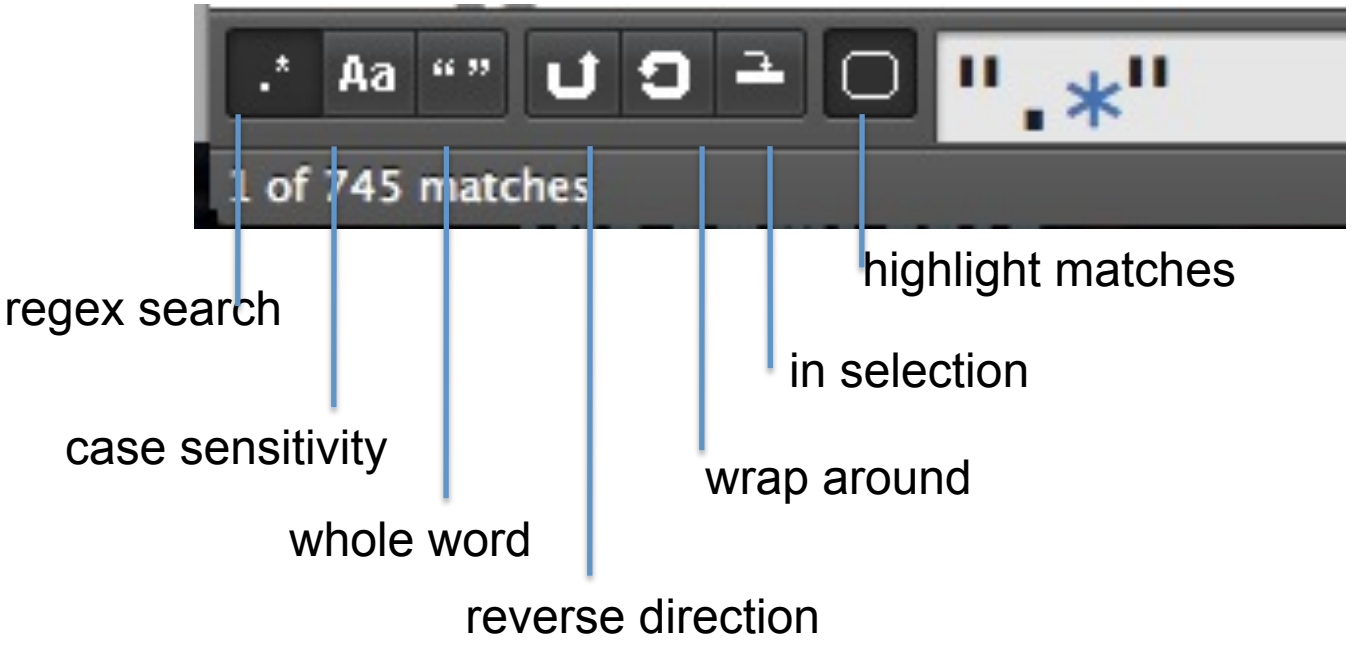
Case doesn't matter, so uncheck case sensitivity.

We get an extra 4,423 matches.

So there are 209,723 vowels in Pride and Prejudice.

Play with other settings

Some settings will change your results, others won't.

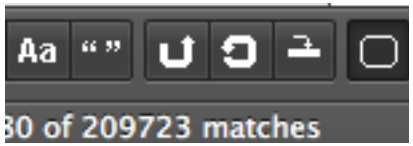


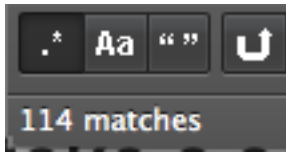
Comparing files

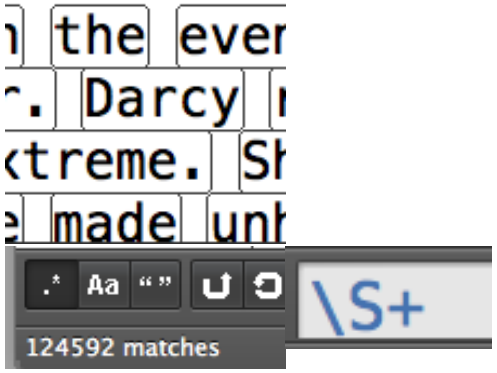
Download *Leaves of Grass*

http://www.gutenberg.org/cache/epub/1322/pg1322.txt
http://www.gutenberg.org/files/1322/1322.txt

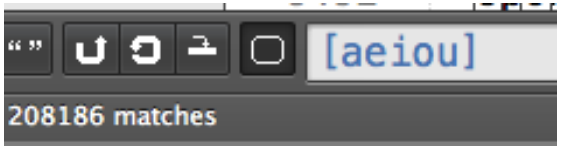
Pride and Prejudice

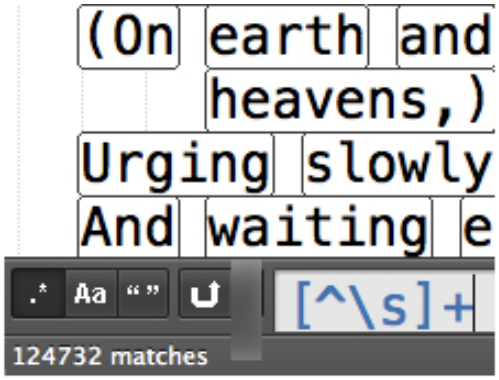






Leaves of Grass





Which is more vocalic?
[aeiou]

How many times is dancing mentioned?
danc*

Which is wordier?
[^\s]+

"+ 1 or more sequence of
^\s non-whitespace letters"
^ not
\s whitespace
\S non-whitespace letters
so \S+ == [^\s]+

Building up a regex string

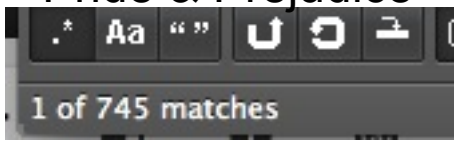
A first pass searching for `".*"`

"non newline, arbitrary characters inside straight double-quotation marked"

Create regex pattern for "one-line quotation"

1. Start and end with " " (double quote)
2. Something that's not a newline newline **.**
3. Zero or more occurrences. *

Pride & Prejudice



Leaves of Grass

(Control+G to go to line.6477 In Leaves of Grass.

"But it is not merely this affair," she continued, "on which my dislike is founded. Long before it had taken place my opinion of you was decided. Your character was unfolded in the recital which I received many months ago from Mr. Wickham. On this subject, what can you have to say? In what imaginary act of friendship can you here defend yourself? or under what misrepresentation can you here impose upon others?"

"You take an eager interest in that gentleman's concerns," said Darcy, in a less tranquil tone, and with a heightened colour.

"Who that knows what interest in him?"

"His misfortunes!" re have been great indee

"And of your inflicti

```
6477    That matter of Troy and Achilles'
6478    Placard "Removed" and "To Let" or
6479    Repeat at Jerusalem, place the nc
6480        Mount Moriah,
```

This regex expression doesn't work very well:

- Line breaks cause long quotations to be counted.
- Most of the quotes for Leaves of Grass are in Project Gutenberg text.

Also, this regex is greedy– `".*"` counts two quotes as one.

These four words are indeed a double-quotation mark followed by a bunch of arbitrary characters, followed by a double quotation mark.

Changing the quotation regex

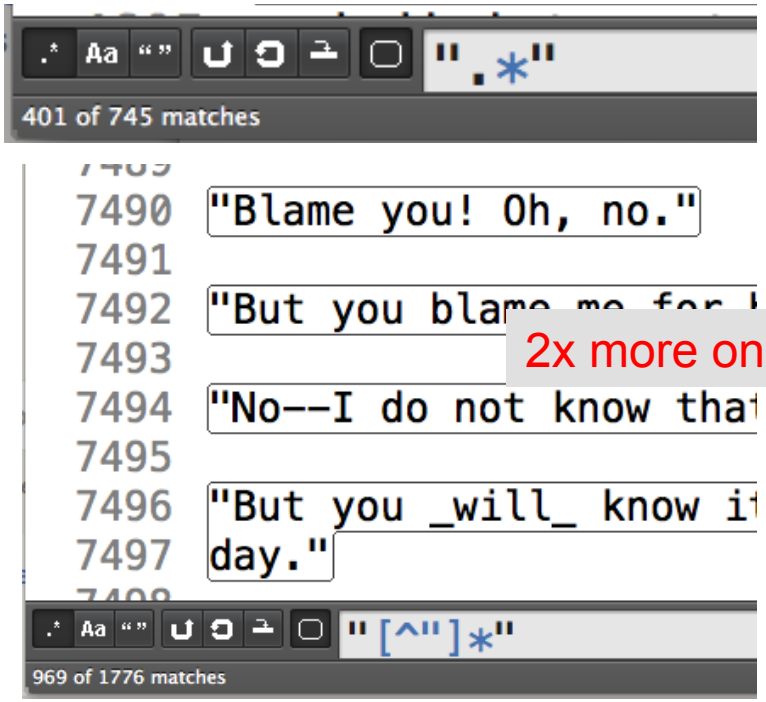
A new one-line quotation search using a pattern that has:

1. “ (double quote mark) at the start
2. [^”] Followed by *anything* that is not a double quote mark
- * zero or more occurrences any characters (except newline).

1. and ends with a double quote mark “

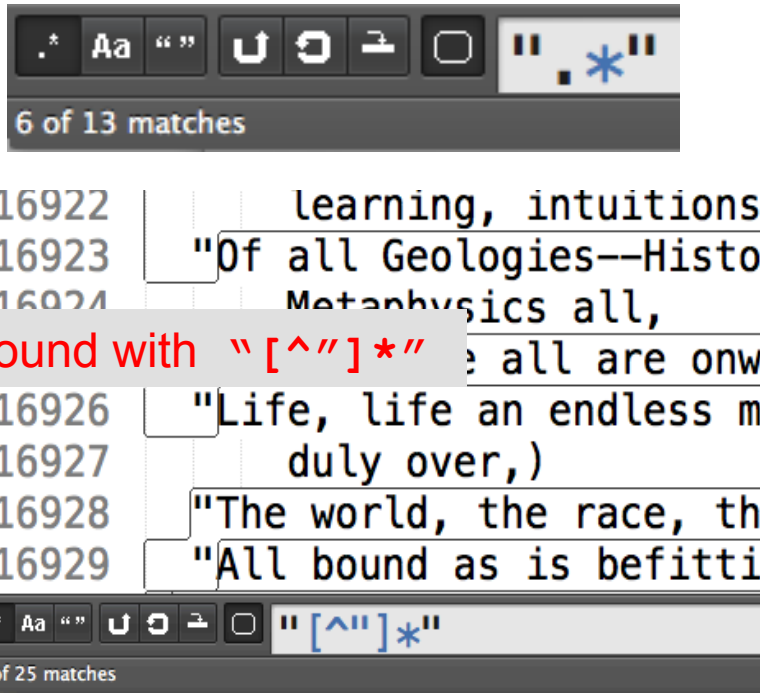
Pride & Prejudice

745 vs. 1776 matches



Leaves of Grass

13 vs. 25 matches



2x more one-liners found with “[^”]*”

Using ^ to exclude characters

Placard "Removed" and "To Let"

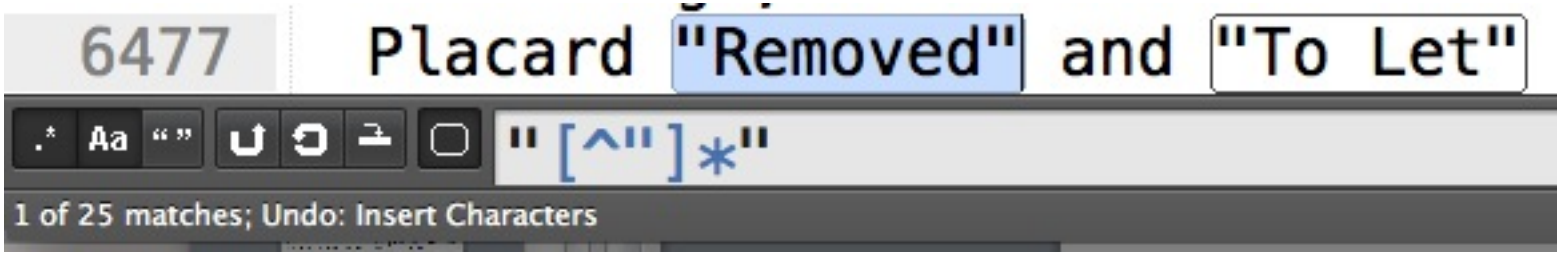
Problem: The regex ".*" will grab two quotes in a single line.

Solution: we should restrict what is inside the quotation marks so that it excludes double-quotes.

“[^”]*”

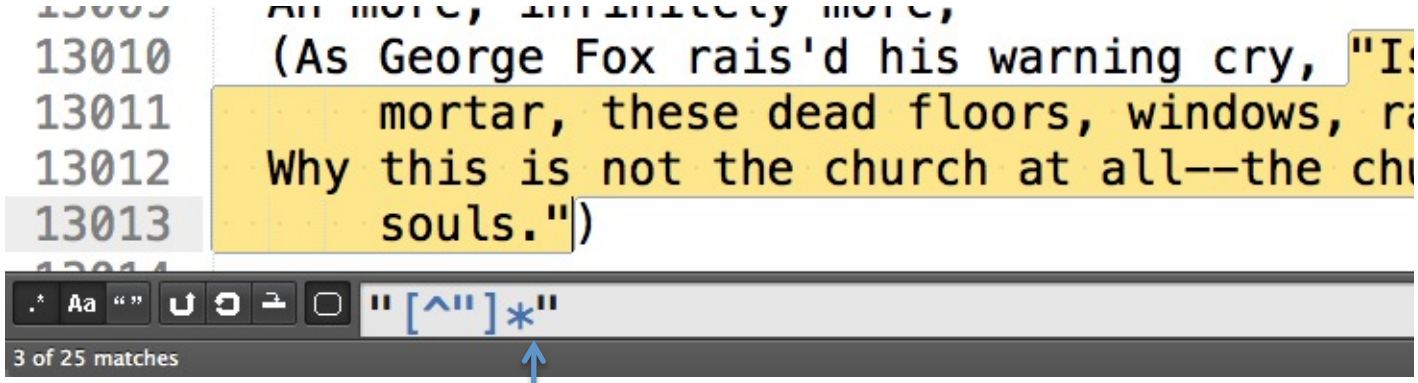
“ start with a double quote
[^”]any character that is NOT a double quote
followed by zero or more non-whitespace characters
“ ending with a double quote

By excluding double-quotes inside our quotes, regex returns each quote separately. We go from 13 to 25 matches by being more specific.

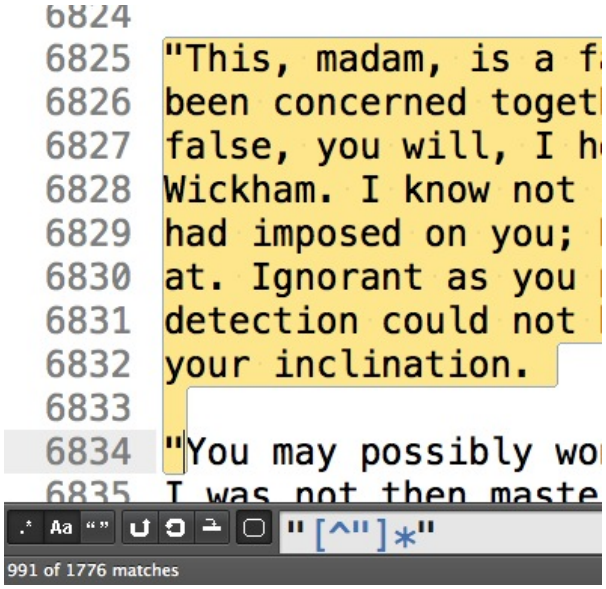


Multiline quotes

If we click “find next,” it turns out that our new regex will also find multiline quotes.



newlines can now appear between quotes

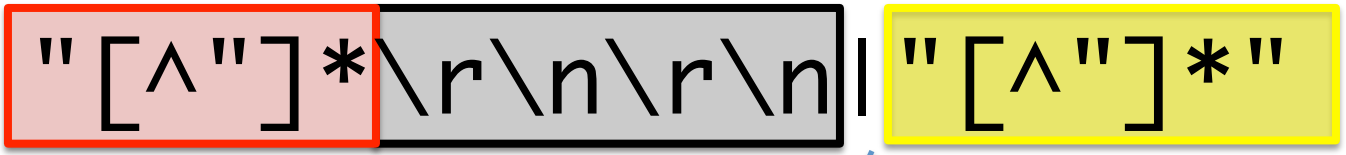


But in Pride & Prejudice, mulitline quotations are split. They start with a quotation mark, have some text with NO quotation mark, then a quotation mark.

So we want to count these as separate quotations.

Split multiline quotes

A special sequence indicating a paragraph break in these files:
a carriage return `\r`, followed by a newline `\n`,
and then a blank line (another `\r\n`)

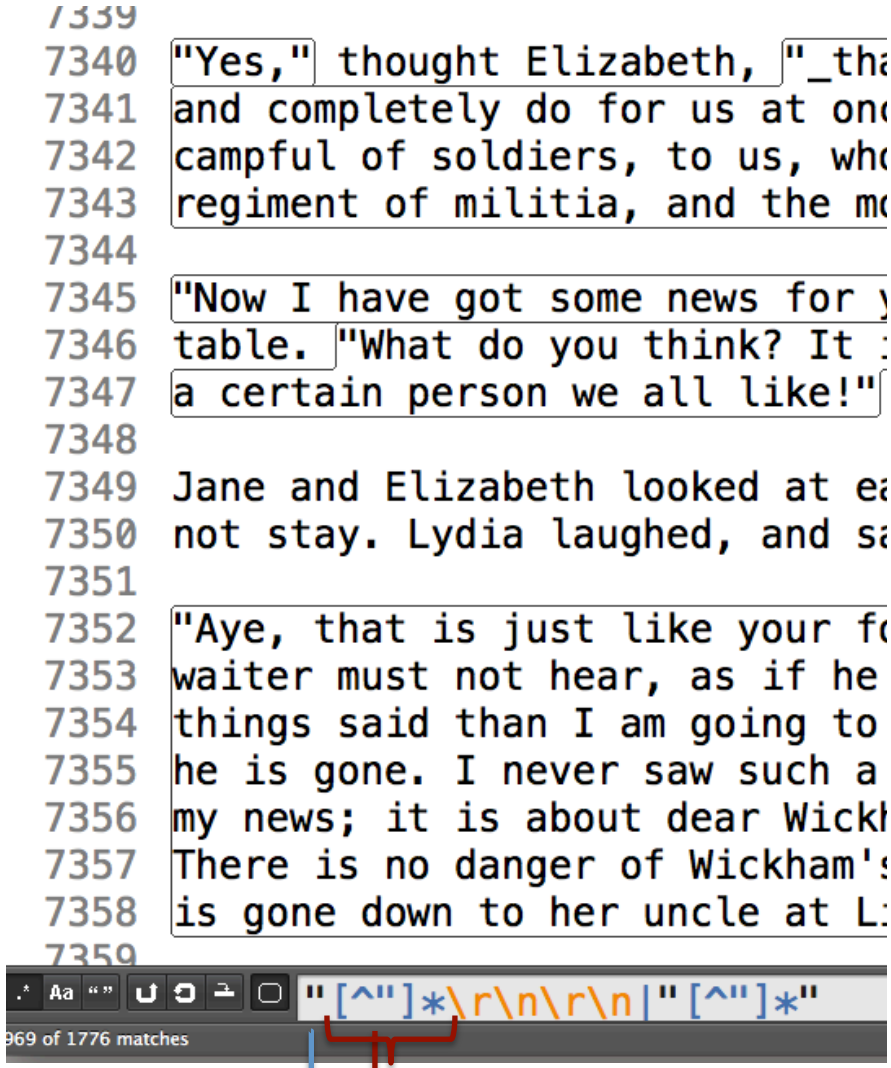


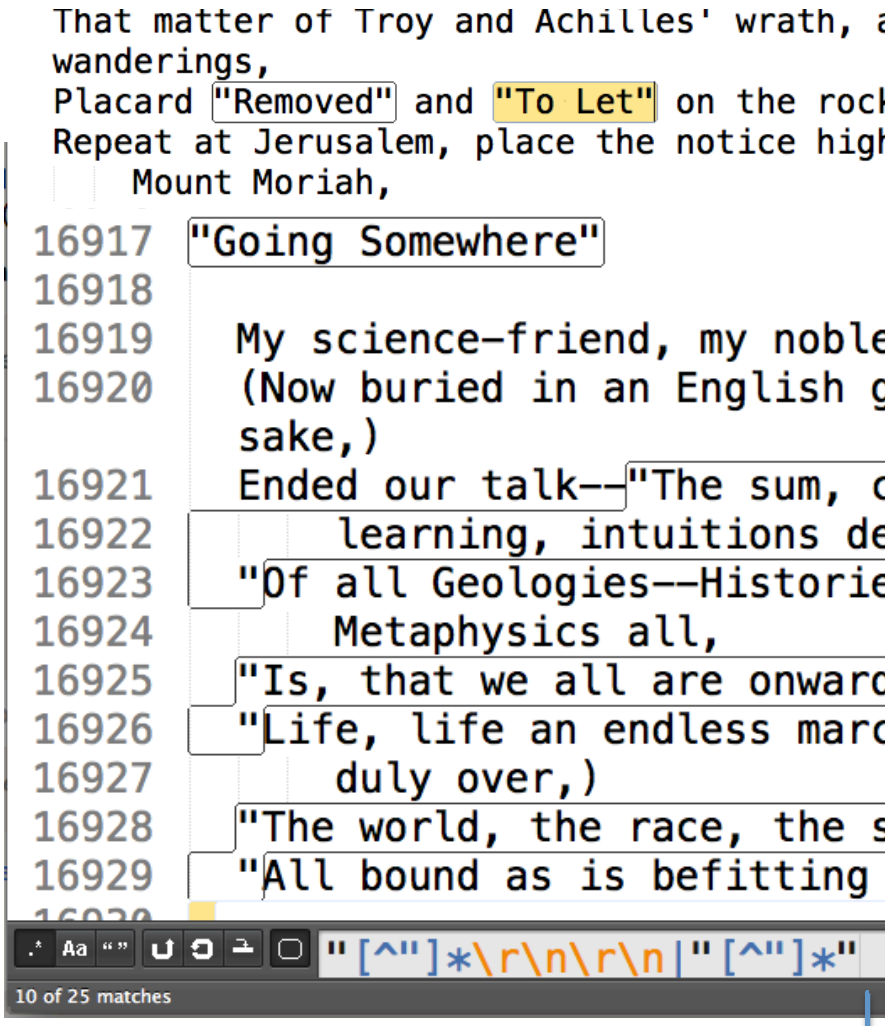
Multiline quotes consist of one or more paragraphs that begin with one quotation mark and have no closing quotation mark

…followed, finally, by a paragraph that has quotation marks at the beginning and the end.

| (The pipe symbol) combines two regex expressions – it's a logical "OR"

*We want to have both the split multiline **and** short quotations.*





Match any text beginning with a double quotation mark,
consisting of zero or more occurrences of a non-quotation mark characters,
or of a newline followed by a quotation mark,
ending with a non-whitespace character
followed by a double quotation mark.

Leonard Richardson

Swapping the dialogue between two different texts

ALICE'S ADVENTURES IN THE WHALE

Leonard Richardson

Alice's Adventures in the Whale is a generated novel created by Leonard Richardson. His script (accessible here for the code-savvy and adventurous) replaces all dialogue in one novel with dialogue from another — in this case, *Alice's Adventures in Wonderland* and *Moby-Dick; or, The Whale*. We present this excerpt for your enjoyment.

CHAPTER V.
"Advice from a Caterpillar"

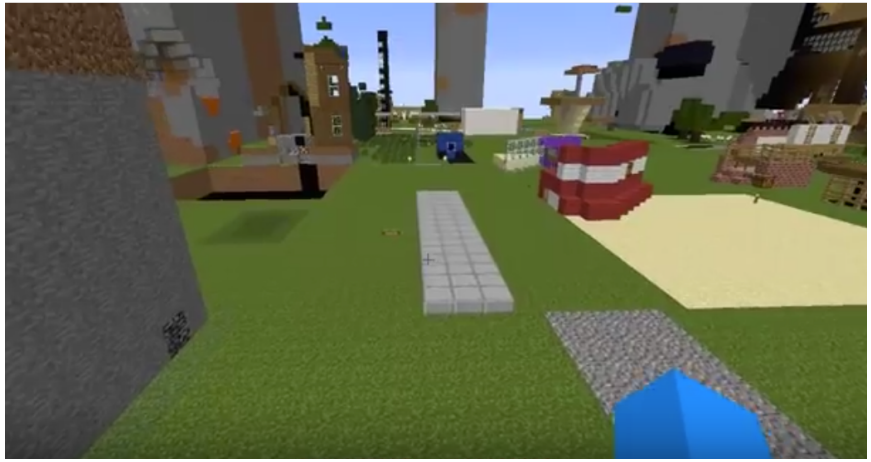
The Caterpillar and Alice looked at each other for some time in silence: at last the Caterpillar took the hookah out of its mouth, and addressed her in a languid, sleepy voice. "I say, pull like god-dam," said the Caterpillar.

This was not an encouraging opening for a conversation. Alice replied, rather shyly, "There she slides, now! Hurrah for the white-ash breeze! Down with the Yarman! Sail over him!"

https://github.com/leonardr/In-Dialogue/

http://wagsrevue.com/20/fiction/richardson1

The Reef -- Minecraft books linked inside map objects



https://www.youtube.com/watch?v=gpkdtv68DYU

https://nanogenmo.github.io/

Essential Concepts

- Regular expressions added to your tools

`.`	Any character except newline.
`\.`	A period (and so on for `\*`, `\(`, `\\`, etc.)
`^`	The start of the string.
`$`	The end of the string.
`\d,\w,\s`	A digit, word character `[A-Za-z0-9_]`, or whitespace.
`\D,\W,\S`	Anything except a digit, word character, or whitespace.
`[abc]`	Character a, b, or c.
`[a-z]`	a through z.
`[^abc]`	Any character except a, b, or c.
`aa|bb`	Either aa or bb.
`?`	Zero or one of the preceding element.
`*`	Zero or more of the preceding element.
`+`	One or more of the preceding element.
`{n}`	Exactly *n* of the preceding element.
`{n,}`	*n* or more of the preceding element.
`{m,n}`	Between *m* and *n* of the preceding element.
`??,*?,+?, {n}?`, etc.	Same as above, but as few as possible.
`(expr)`	Capture *expr* for use with `\1`, etc.
`(?:expr)`	Non-capturing group.
`(?=expr)`	Followed by *expr*.
`(?!expr)`	Not followed by *expr*.

RegExExpress
a cool ASCII tutorial game
https://pyweek.org/e/RegExExpress/

Stack Overflow
http://stackoverflow.com/

Google
http://google.com

Trevor Payne's Regex Tutorial
on youtube

http://regexpal.com

Regex, the 2nd coming

When you read a line of code like this:

```
name.match(/^[\da-z]+\d$/);
```

You probably actually see this:

```
name.match(justabunchofrandomjunkallsmashedtogether);
```

```
name.match(/
        ^     # from the beginning
        [     # a set of chars containing
          \d  # any digit
          a   # the letter a
          -   # through
          z   # the letter z
        ]     # end class
        +     # one or more
        \d    # any digit
        $     # to the end
        /);
```

"Match from the start a set of one or more digits or a-z followed by any digit to the end of the string." You could even get more succinct and say, "Match completely one more digits or lowercase letters followed by a digit."

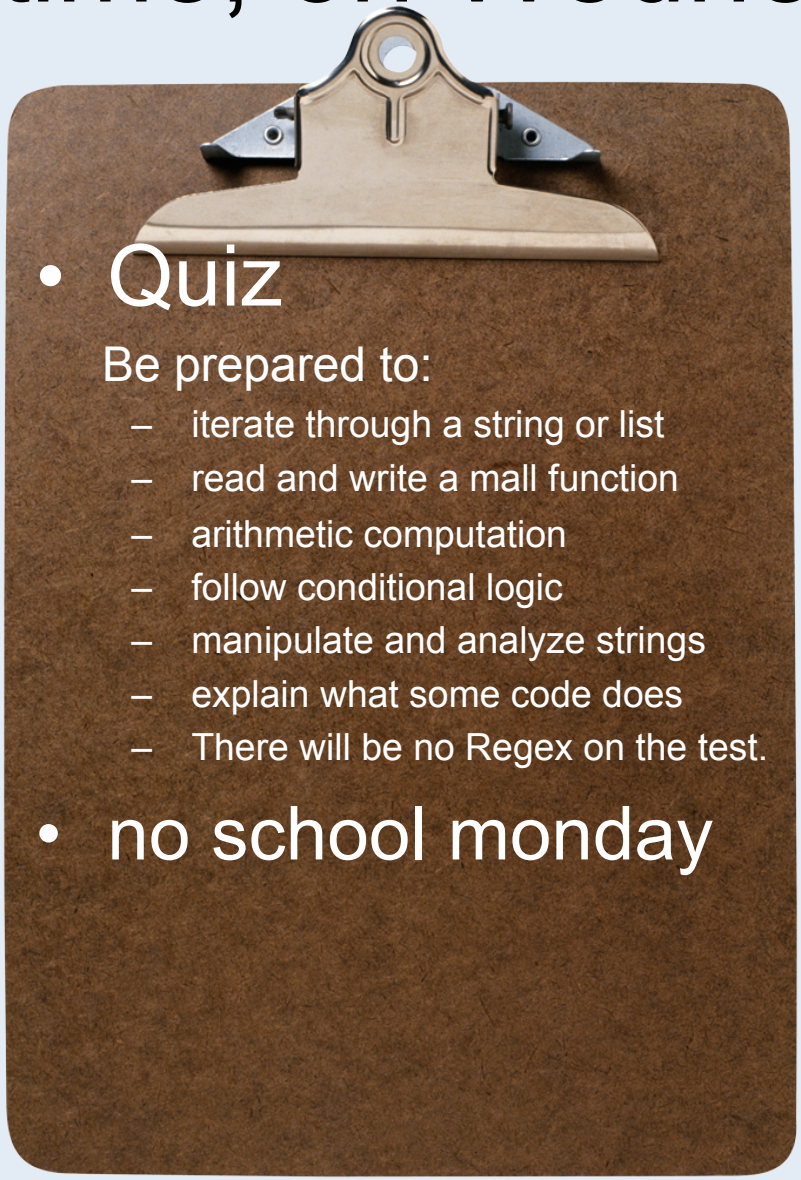
Image from RegexTheHardWay

Homework

1 Analyze and compare a fiction book with a book of poetry in any way you see fit, using your existing knowledge of regular expressions and expanding that knowledge if you like. You can try to lexically determine the length of sentences (e.g. references to “Mr.” Bennet may make that difficult), for instance. Or you can look for certain words, or sets of words, and compare what percentage of the text they take up in each case. Some of the simplest and most commonly overlooked words (articles or pronouns, for instance) may actually be very interesting to consider; avoid assembling large sets of nouns, verbs, or adjectives. You may use code (but its not required) to calculate these metrics.. Upload a file (could be a infographic or powerpoint slide) demonstrating your findings, including the regex you used to Canvas.

2 Watch a tutorial video on python.
https://www.youtube.com/watch?v=ZdDOauFIDkw
Follow along and extend the online tutorial to create custom expressions to grab the title, headers, and physical addresses (e.g. 120 Boylston St.) from a web page of your choosing*.
Upload your ipython notebook from the video, along with useful regex.
*Feel free to choose a plain web page.

Next time, on Wednesday



- Quiz

  Be prepared to:
  - iterate through a string or list
  - read and write a mall function
  - arithmetic computation
  - follow conditional logic
  - manipulate and analyze strings
  - explain what some code does
  - There will be no Regex on the test.
- no school monday

Summary of today

- Technical practice
  - Textual analysis of large files using regex
  - Grabbing source text off the web
  - Typing in your first regex expressions
- Other python tutorials that are good

  Trevor's "Let's Learn Python" tutorials are also good:
  https://www.youtube.com/playlist?list=PL82YdDfxhWsDJTq5f0Ae7M7yGcA26wevJ

  Lynda.com Programming fundamentals in the real world
  https://www.lynda.com/Python-tutorials/Programming-Fundamentals-Real-World/418249-2.html