Python Data Products
Course 1: Basics

Lecture: text and string processing in Python
Learning objectives

In this lecture we will...
- Perform simple manipulations of string data in Python
- Discover a few useful library functions for string processing
In this lecture we'll look through a few functions to manipulate string data in Python:

- `string.split()` and `string.join()`
- List operations on strings
  - `index()` and `find()`
- The "string" library
First let's read in a review from the Yelp dataset:

```python
In [1]: import json
    import string

In [2]: path = "/home/jmcauley/datasets/mooc/yelp_data/review.json"

In [3]: f = open(path)

In [4]: d = json.loads(f.readline())

In [5]: d

Out[5]: {'business id': '0W4lkclzZThpx3V65bVgig',
        'cool': 0,
        'date': '2016-05-28',
        'funny': 0,
        'review id': 'v0i_UH3Mo_hPBq9bxWvW4w',
        'stars': 5,
        'text': "Love the staff, love the meat, love the place. Prepare for a long line around lunch or dinner hours. 
        They ask you how you want you meat, lean or something maybe, I can't remember. Just say you don't want it too fatty. 
        Get a half sour pickle and a hot pepper. Hand cut french fries too.",
        'useful': 0,
        'user id': 'bv2nCi5Qv5vroFiyK6opiw'}

In [6]: review = d['text']
```
• We saw string.split() previously when reading CSV/TSV files
• Here, .split() can be used to convert a string to a list of words (or we could split it based on another character)
• This process is known as **tokenization**
Code: String.join()

In [9]: ' '.join(reviewWords)
Out[9]: "Love the staff, love the meat, love the place. Prepare for a long line around lunch or dinner hours. They ask you how you want you meat, lean or something maybe, I can't remember. Just say you don't want it too fatty. Get a half sour pickle and a hot pepper. Hand cut french fries too."

- String.join() is like .split() in reverse: it takes a list (here the list of words in the review), and converts them to a string, by placing the same token (here a space character) in between each one
• String.lower() converts a string to lower case
• This operation can be useful before we compute statistics on strings – it allows for easier comparison between different variants of the same word
• Similarly `string.upper()` converts a string to upper case
Code: List operations on strings

- Regular python list operations will work on strings

```
In [11]: len(review)
Out[11]: 289  Note: # characters

In [12]: len(reviewWords)
Out[12]: 56   Note: # words

In [13]: review[:18]
Out[13]: 'Love the s'

In [14]: reviewWords[:10]
Out[14]: ['Love', 'the', 'staff', 'love', 'the', 'meat', 'love', 'the', 'place', 'Prepare']

In [15]: reviewWords.index("pickle")
Out[15]: 46   Note: word position

In [16]: review.find("pickle")
Out[16]: 238  Note: # characters into review that the word appears

In [17]: review.find("cucumber")
Out[17]: -1

In [18]: review.count("love")
Out[18]: 2

In [19]: review.lower().count("love")
Out[19]: 3
```
We can remove punctuation by performing a list comprehension on the string.
The "string" library contains utility functions that we can use (for e.g.) to get the list of punctuation tokens.
Finally, we have to use `join` to convert the output back to a string.
Strings in Python

These are just a few of the most basic operations, see also:

- `string.startswith()` (etc.)
- `string.isalpha()` (etc.)
- `string.strip()`
- Other operations in the string library
- (later) the NLTK library
Summary of concepts

• Understand a few of the basic Python string operations
• Apply list operations to strings
• "Tokenize" strings into lists and vice versa

On your own...

• Try computing simple statistics from string data, e.g. how often does a particular word appear among Yelp reviews, and which words are the most common?