# Assignment 2

deadline: Feb 27, 11:59 PM (Thursday)

# Your goal

- Practice what you just learned in class.
  - ✓ How to read local files and save as local files
  - ✓ How to find useful information from raw data.
  - ✓ How to create function.
  - ✓ How to use comprehensions and lambda
  - ✓ How to create basic graphic. (optional)

# Zipf's law

**Zipf's law:** $r * freq = A * N$

r = word rank    freq = word frequency    A = constant.   N = total number of words in collections

**References**

>https://isquared.wordpress.com/2010/11/08/zipfs-law-how-can-something-so-simple-explain-so-much/

>http://www.ccs.neu.edu/home/ekanou/ISU535.09X2/Handouts/Review_Material/zipfslaw.pdf

>https://www.youtube.com/watch?v=fCn8zs912OE

# Question 1

- Learn and understand Zipf’s Law.
- Pick any file (or files) in NLTK package.
- Prove if Zipf’s Law works in Natural Language or not.
- REQUIREMNTS: (Must Do)
  - ✓ Use ’glob’ to open and read file. If read multiple files use LOOP to achieve.
  - ✓ Create at least one function and one lambda function.
  - ✓ Save your data as an CSV file contains (Word, Rank, Frequency) with TITLE and Sort it by RANK.
  - ✓ Open your CSV file, generate a log-log figure according to the data you saved.
  - ✓ Write a conclusion at the end of your code base on your figure.

# Question 1 continue…

- (Optional) Use ‘matplotlib’ to generate a log-log figure in your code.
- Before Submission Please Check:
  - ✓ Do I have Markdown TITLE at very beginning?
  - ✓ Do I use the required method to open and read files?
  - ✓ Do I use 'def' and ‘lambda’ in my code?
  - ✓ Do I write your code in small blocks instead of in one big block?
  - ✓ Do I have proper outputs or comments to show your progress?
  - ✓ Do I add TITLE in your CSV file? Is the data well sorted?
  - ✓ Do I create the figure with right form(log-log) in your csv file?
  - ✓ Do I give the conclusion at the end of your code?

ALL DONE? Submit your .ipynb file and your CSV file. DO NOT ZIP!

# Question 2

**Part 1**

Read all the json files in the folder called Data.

- There are three categories of json files in this folder. They are identified by the key called “term” in each of the json file.
- Create a folder structure to read all these json files and store them into these separate folders. You are expected to create a hierarchy of folder structure.
- Example:
- You can place all restaurants json files in a particular country (say Australia) in the same folder. How you group the json files and create a folder structure is your choice. Your task is to identify criteria by which you can group all these json files and store them.
- (You could use these keys to create hierarchy and store json files: Country, city, categories)

# Question 2

- Output Format:
- Create a folder (Name: Data Processed)
- In this folder you should have a hierarchy of folder structures

(Example: Data Processed/Australia(AU)/……..)

A good idea is that you can classify json files on Country name or code (You can create a hierarchy of folder structures to effectively sort and store the files). The original json files in the folder "Data" have the name of the "id" key in the file. You can even think of changing the name of the json file when you read and store them.

**Submission**: You will be submitting the Folder (Data Processed )which is storing the json files and the jupyter notebook (code) that you used to create this hierarchy of folder structures.

# Question 2

**Part 2**

Read all the json files in the folder called Data.

- Read only the json files which contain the key called “restaurants”
- Each (or most of the json files) contain a key called “open” which contains the details of the operation (timings) of the restaurants. For each json file, read the timings of the restaurants.
- Data of the operation timings of the restaurants is present for each day of the week. I want you to extract each of this data and write it in an excel sheet.
- Example:
- For a particular restaurant named “The Coffee Grounds”, the excel sheet should look like this:

# Question 2

**Sample Output:**

| Name of restaurant | City | Country Code | Day of week | Start Time | End time |
|---|---|---|---|---|---|
| Coffee Grounds Acton | Acton | AU | 0 | 700 | 1830 |
| Coffee Grounds Acton | Acton | AU | 1 | 700 | 1830 |
| Coffee Grounds Acton | Acton | AU | 2 | 700 | 1830 |
| Coffee Grounds Acton | Acton | AU | 3 | 700 | 1830 |
| Coffee Grounds Acton | Acton | AU | 4 | 700 | 1830 |
|  |  |  |  |  |  |

The following rows will have the next restaurants details and so on.

# Question 2

**Bonus:** Split “Start Time” column into two different columns having the hour and minute in each of them

Split “End Time” column into two different columns having the hour and minute in each of them

**Sample Output:**

| Name of restaurant | City | Country Code | Day of week | Start Time Hour | Start time Minute | End time Hour | End Time Minute |
|---|---|---|---|---|---|---|---|
| Coffee Grounds Acton | Acton | AU | 0 | 7 | 00 | 18 | 30 |
| Coffee Grounds Acton | Acton | AU | 1 | 7 | 00 | 18 | 30 |
| Coffee Grounds Acton | Acton | AU | 2 | 7 | 00 | 18 | 30 |
| Coffee Grounds Acton | Acton | AU | 3 | 7 | 00 | 18 | 30 |
| Coffee Grounds Acton | Acton | AU | 4 | 7 | 00 | 18 | 30 |
|  |  |  |  |  |  |  |  |

# Question 2

**Submission:**

- You will be submitting the output excel file (CSV) in the desired format along with the jupyter notebook file containing the code.