Homework Assignment 1: Due 01/29/2023 at 11:59 PM

You may use whatever IDEs Platforms / text editors you like, but you must submit your responses and source code files (Jupyter notebooks are considered as such) on iCollege.

Note: Please refer to the Jupyter notebooks created in class. They provide useful hints for solving the following problems. In the submitted Jupyter notebook(s), your code cells' output (if any) should be visible without executing the notebook.

1) (25 points) Download the resource file "sw-security-urls.txt". Create a Jupyter notebook (in Python) that performs the following tasks:
   a) (4 points) Write and call a function that downloads the contents of all URLs in the resource file (Wikipedia articles) one by one to a local directory. Afterwards, this local directory should contain 60 HTML-files.
   b) (9 points) Using the library BeautifulSoup, extract from each downloaded HTML-file its title, its main text found under the element <div id="bodyContent" …>, and the following JSON-LD formatted metadata (if existing): "name", "url", "datePublished", "headline". Store this data for each file in a dictionary and add this dictionary to a list. This list should then contain 60 dictionaries. Output this list.
   c) (3 points) At the same time, create a string that contains the main text of all the Wikipedia articles. To do that, you can concatenate the individual main texts extracted in the previous step (leave a whitespace between each text). Finally, save the content of this string in a local text file. Output this string.
   d) (6 points) Create a pandas dataframe from the list obtained in step 1b) and set its index to the article's title. Output this dataframe. Which Wikipedia articles do not provide "headline" information in their JSON-LD formatted metadata? (You can use a Documentation cell in your Jupyter notebook to provide your answer.)
   e) (1 point) Replace missing data (NaN) in the pandas dataframe with "#empty#".
   f) (2 points) Store the obtained pandas dataframe (set index_label to "title") in a local sqlite3 database.

2) (15 points) Extend the Jupyter notebook (in Python) created in 1) by providing solutions to the following tasks:
   a) (4 points) Create an NLTK-corpus using PlaintextCorpusReader from the local text file created in 1c). For the tokenization of sentences please use the RegexpTokenizer with the pattern: r'[^.!?]+' . Using NLTK's method words(), output at most 200 words of the corpus together with their absolute frequency in descending order (regarding frequency). Note: this output might contain elements like punctuation marks and stop words.
   b) (6 points) Filter out the following elements from the word list obtained in the previous step:
      - any stop words found in NLTK's English stop word corpus
      - any elements that do not contain the letters A-Z and a-z (Regular expression pattern: r'[^A-Za-z]')

      Output this filtered list of (at most 200) words together with their absolute frequency in descending order (regarding frequency). Also, count in a defaultdict how often a word occurs that is still present after the filtering is complete.
   c) (5 points) Create and output a word cloud using the defaultdict obtained in 2b).