CIS 41B - Lab 4: Web API access, multithreading, multiprocessing, review of iterables and GUI

In this current time of ever changing orders and recommendations, it's good to occasionally check for the latest news. Write an app that's a personalized news aggregator. The app lets the user choose their news sources, get the latest headlines from their news sources, and read the news article online.

In a quarter during normal time, lab 4 is the last lab that students use to qualify for the team project, which is relatively independent work, so lab 4 has opportunities to show that you're an independent learner, just as if you're at work at a company.

Step 1: Fetch data

The input data is from <https://newsapi.org/>. You can go to the website and click on the "Get Started" tab to get your personal API key, and to see examples (in Python!) of how to use the API to get the data for the lab.

There are 2 kinds of API calls that you need to make:

1. An API call to get the names (and other info of your choosing) of all *news sources* that are *in English* and are *from the US*.

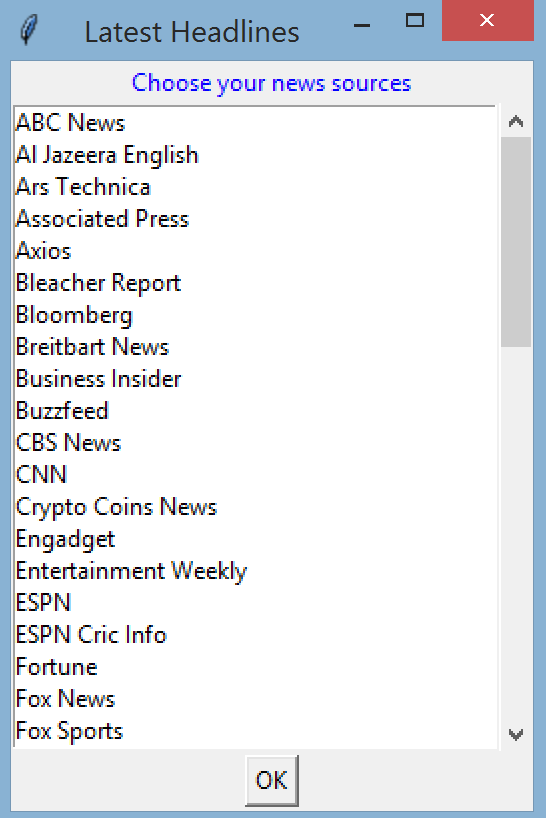
2. An API call to get the *headlines* of a chosen news source.

The info you need are at the "Get Started" tab and under the "Documentation" links on the left side of the page. The Python examples are especially helpful.

Step 2: GUI

Here is the general flow of the GUI, which has 2 classes for the 2 windows.

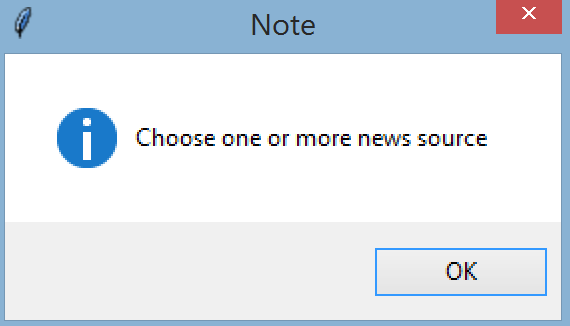
A. The application starts with a main window that contains:



* a title
* a line of instruction to tell the user to choose their news sources
* a listbox of 20 items with a scrollbar
* each item in the listbox is a US news source, listed in alphabetical order
* an OK button for the user to lock in their choice(s)

B. The user can click on one or more items to choose the news source.  
 Then the user can click OK to submit their choices.

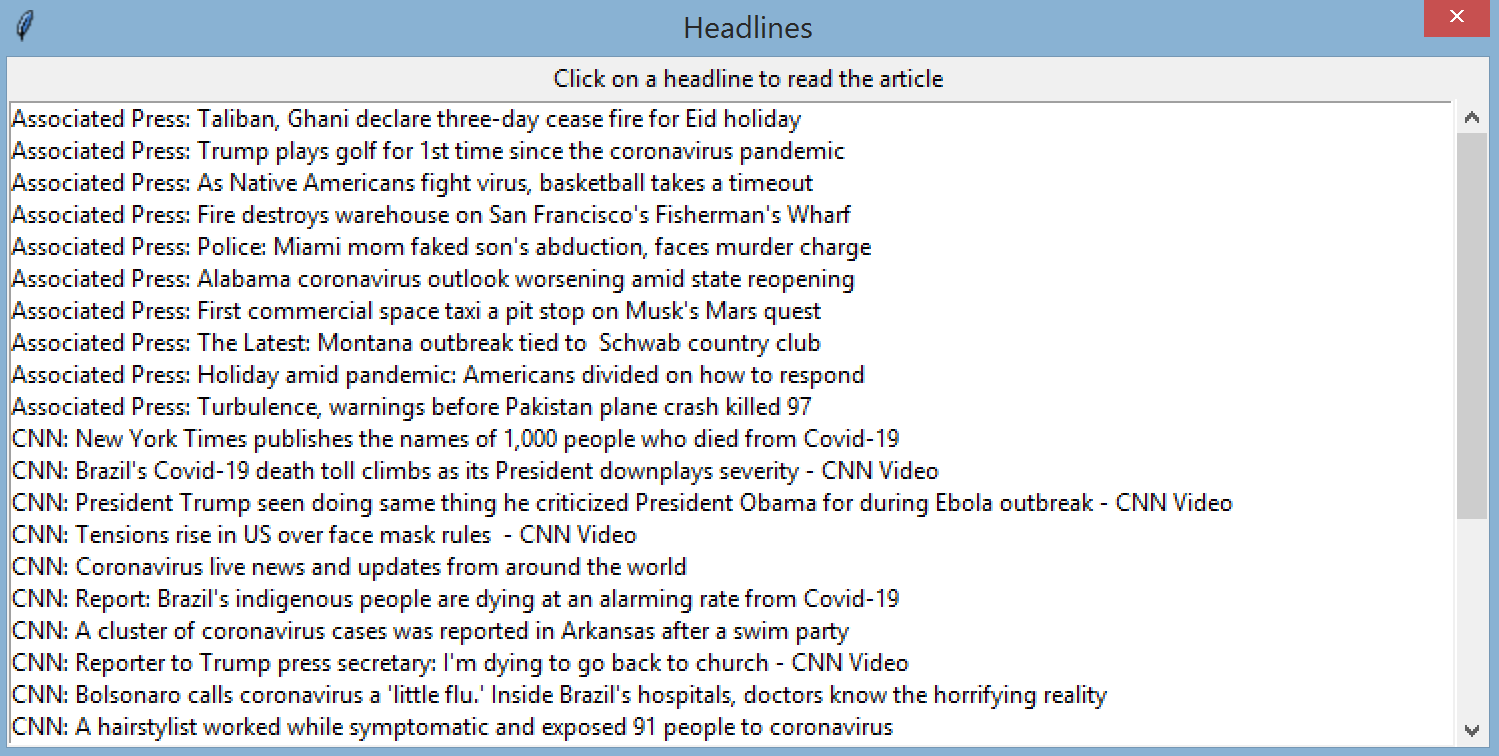
C. If the user doesn't make any choice and clicks OK, a messagebox window  
 pops up to let the user know they need to select a news source.



D. If the user makes at least one choice and clicks OK, then a display window shows up.

E. The display window contains:

* a title
* a line of instruction to tell the user to click on an item to read the news article
* a listbox of 20 items with a scrollbar. Make the listbox wide enough to show the complete text strings (if one string is extraordinarily long, then it can be partially shown).
* each item in the listbox is a headline from the news source that the user chose. The item has 2 parts:
  + the source name, followed by a colon
  + the headline from the source



The above is a the display window after Associated Press, CBS News, and CNN are chosen.  
CBS News is outside of the listbox view. (This was on 5/23 so the headlines you have will be different)  
The news source headlines could be displayed out of sequence, as shown above. In sequence would be in alphabetical order: Associated Press, CBS News, CNN

F. The listbox is implemented such that as soon as the user clicks on a listbox item, a browser tab opens on the user's computer to show the web page for the news article.

To open up a browser with a URL: 1. import webbrowser  
 2. webbrowser.open(url)

G. The user can continue to go back to the display window and click on another listbox item to see another webpage.   
While the display window is opened, the main window is disabled, so the user cannot create a 2nd display window.

H. When the user clicks 'X' to close the display window, then the user is back at the main window. Any previously selected items in the main window listbox should be cleared. This includes the case when the user does not choose to view any web page from the list of headlines.

To clear the listbox selections: listbox.selection\_clear(0, tk.END)

Step 3: Multithreading

* After the user chooses their news sources , use threads to fetch the headlines from each source in order to display them in the display window.
* Use one thread for each news source. Each thread is a child thread, and the main thread is the GUI.
* Each child thread should pass data back to the GUI so that the GUI can display the data. The child thread should *not* insert data in the listbox.

Step 4: multiprocessing

* Make a copy of your py file with threads and use the copy to work with processes.
* Change all threads to processes.
* Change the method that the threads run into a function (not part of a class) for the processes to run.  
  Each process is for one news source, and each process sends the headline data back to the main process. The main process displays the data in the listbox.
* Don't forget to check for \_\_name\_\_
* The application should still work the same way as with threads.

Extra credit (EC): 3pts:

* Make a copy of your .py file with processes and use the copy for the EC.
* Only use the concepts discussed in the class notes, find a second way for the processes to send data back to the main process.
* The application should still work the same way.

Turn in:

lab4threads.py and lab4processes.py (and optionally lab4processes2.py)