CIS 41B - Lab assignment 3: web scraping, data storage, review of regex, iterables, GUI

Write an application that downloads from Wikipedia the list of most popular names per country, then presents the user with a GUI to look up popular names for a country.

The application has 2 parts: lab3back.py (the backend to get data) and lab3front.py (the frontend with the GUI)

The 2 parts do *not* work with each other (no importing of data or methods from one file to the other).

Instead: - the lab3back.py will produce a JSON file and an SQL database file

- the lab3front.py will read from the database to present data to the user

lab3back.py description

Part A

1. The Wikipedia URL is: <https://en.wikipedia.org/wiki/List_of_most_popular_given_names>
2. Fetch data from the URL and extract each country name and its associated popular names.

* Some of the names are in non-Western language Unicode, so you might run into Unicode error when you store them as Python strings and display them on your system (unless your system is configured for these characters.)
* To work around this problem, when you use get\_text() to get the text data inside a tag element, use:

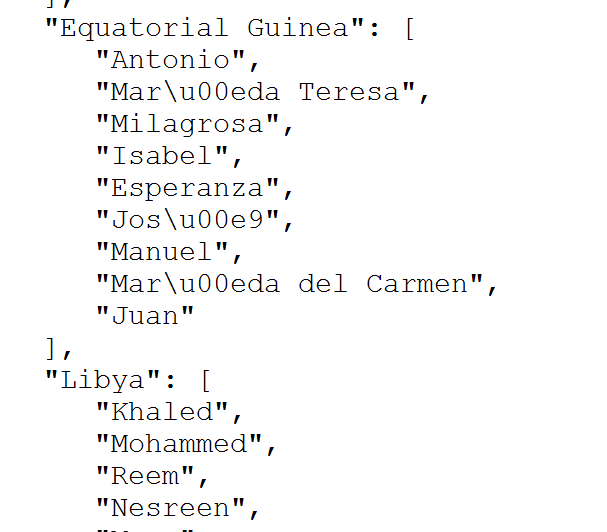
tag.get\_text().encode(sys.stdout.encoding, errors='ignore').decode(sys.stdout.encoding)

* The encoding and decoding are parts of the Python [codec](https://docs.python.org/3.7/library/codecs.html) module.
* The encoding tells Python to ignore Unicode errors when translating the string to a byte string
* Then the decoding turns the byte string back into a Python string, without the characters that your system can't display.

(The reason the browser can display all these characters is because modern browsers have very sophisticated language detection code -- a reason why Google and other international companies are looking hard for computational linguistics majors, in case this is an interesting topic for you)

1. The country name is in the first column of each table on the webpage. The country name is considered the part that comes before () or before comma.  
   Example: Japan (2015)[37]  means the country name is Japan  
    Israel, Muslim boys (2015)[30][31][32] means the country name is Israel
2. All popular names under the same country name should be combined together, under one country name.  
   Store all data into a JSON file.

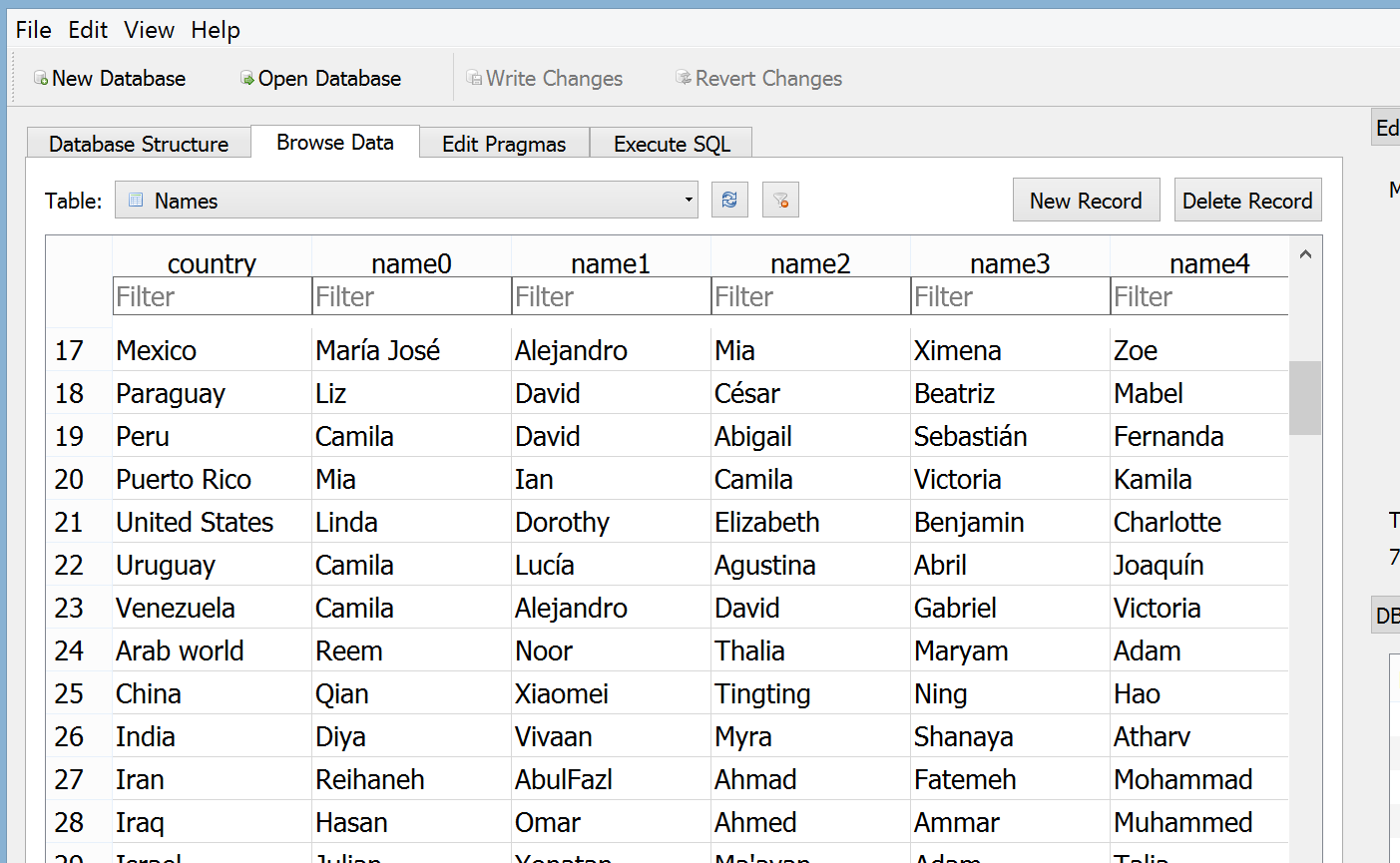
Sample of the JSON file (any \u00 string is a Unicode character that's outside the ASCII range, showing that Python strings are Unicode strings, not ASCII strings):



Part B

1. When you're done creating the JSON file, comment out all the code that you wrote for Part A. You don't need to go to the Wikipedia page any more.
2. Read data from the JSON file and store into an SQLite database.
3. Each row of the database is for 1 country record.   
   For each record, the first column is the country name, the subsequent columns are for the popular names, one name per column.

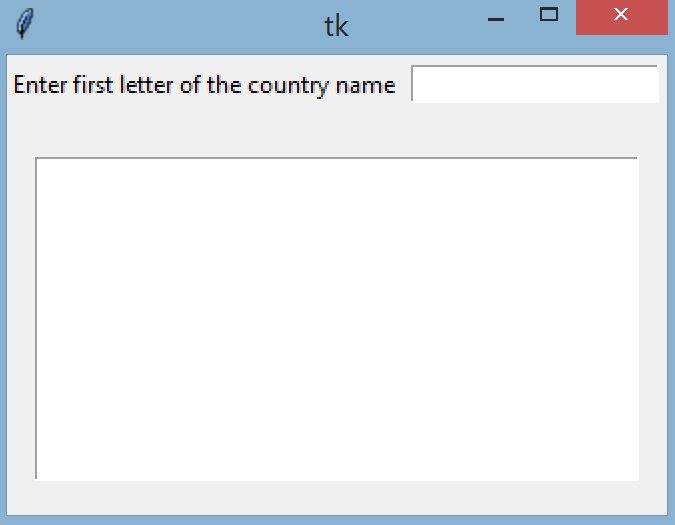
Sample of the database, from the DB Browser tool:



lab3front.py description

1. Write a GUI with 2 window classes: a main window and display window.
2. The main window has:

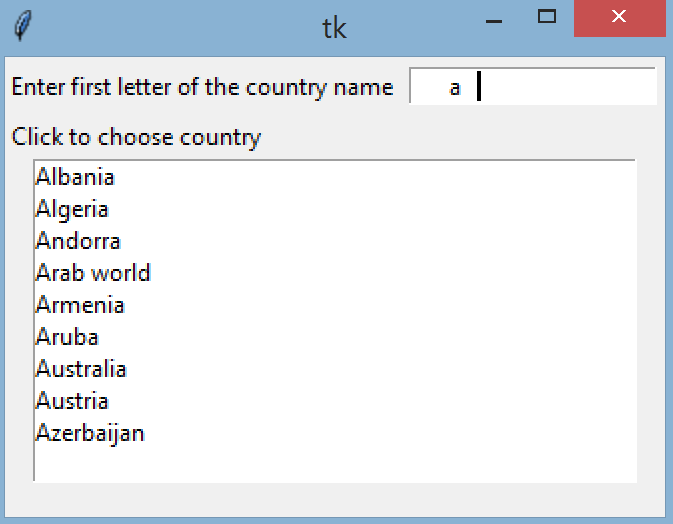
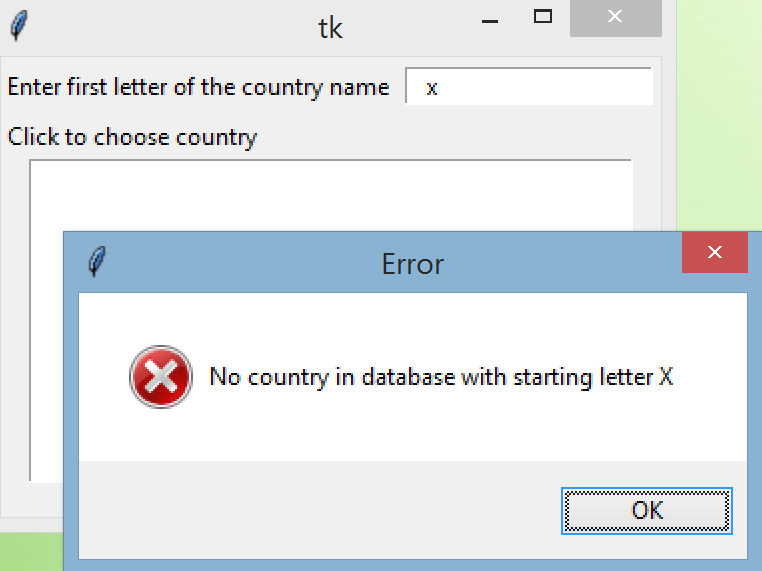
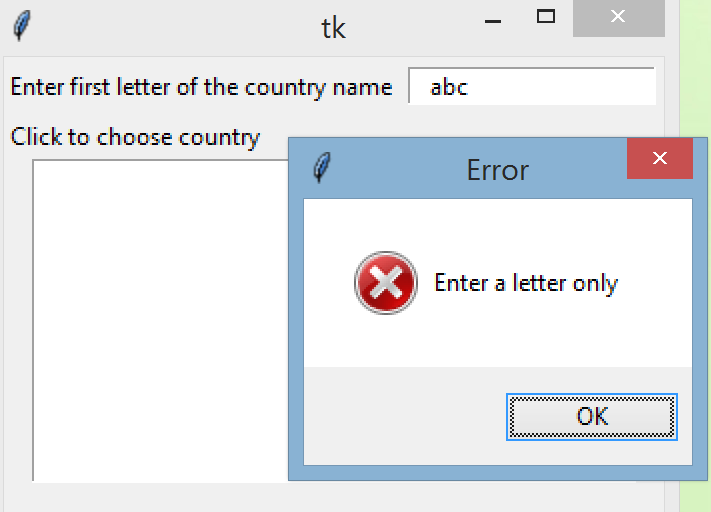
* A prompt and entry object for the user to enter a letter, which is the first letter of the country name that they want to look up.
* A listbox to display the matching country names. The listbox allows only *one selection* from the list.
* The listbox should be able to show around 10 lines of text (use your choice of exact number).



Main window at start

1. When the user enters an input, check that the input:

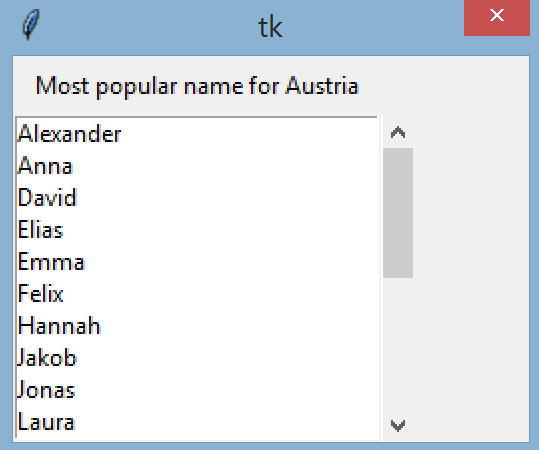
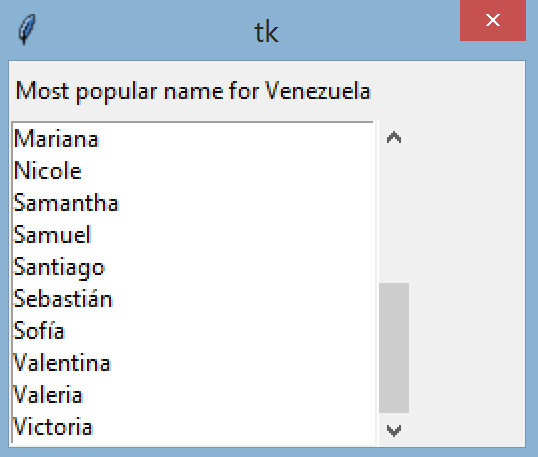
* is exactly one letter, uppercase or lowercase, and space in front or after is okay, but no other characters
* is within the list of letters that have matching country names
* If the letter matches both conditions, show the list of matching country names and show the text to tell the user to choose one country.
* If the input is not valid or has no country match, pop up an error message window with a descriptive error message. The user can click OK to acknowledge and enter in a different letter.

Note spaces in front and after 'a' Non-matching letter Not a letter  
(The vertical bar is the cursor)

1. If the user clicks to select a country name from the listbox, open a display window with the popular names for that country. The display window has:

* An explanation with the chosen country name.
* A listbox which can display 10 lines of text, to display the popular names for the chosen country.
* A vertical scrollbar for the listbox, because most countries have more than 10 names.
* The window should get the focus in the GUI, but the user can still go to the main window and select another country from the listbox or enter a new letter.
* The popular names are displayed in alphabetical order in the listbox.

Display window with scrollbar at top position Display window with scrollbar at bottom position

1. Some tips on handling data efficiently:

* lab3front.py should get data from the database file. Do not go to the website or import anything from lab3back.py or use the JSON file.
* Since the list of country names won't change, it's a good idea to generate a list of letters that have matching country names at the start of the program, then use this list to check the user input. It will be faster than checking the database every time the user enters a letter.
* Don't extract all data from the database to store in memory. Only fetch the data that the user requests.

When done, turn in 4 files: lab3front.py, lab3back.py, the db file, and the json file.