**1. Project Overview [~1 paragraph] What data source(s) did you use and what technique(s) did you use analyze/process them? What did you hope to learn/create?**

I used Wikipedia as the data source and mediawiki package to search Wikipedia, get article summaries, get data like links and images from the requested page. I analyze the word frequencies of the requested page, most common words, and words that in one page that the other one does not have. I also designed some simple user interfaces that users can type in words that they want to know and they can get summaries, categories, most common words, coordinates and logos of the words.

**2. Implementation [~2-3 paragraphs] Describe your implementation at a system architecture level. You should NOT walk through your code line by line, or explain every function (we can get that from your docstrings). Instead, talk about the major components, algorithms, data structures and how they fit together. You should also discuss at least one design decision where you had to choose between multiple alternatives, and explain why you made the choice you did.**

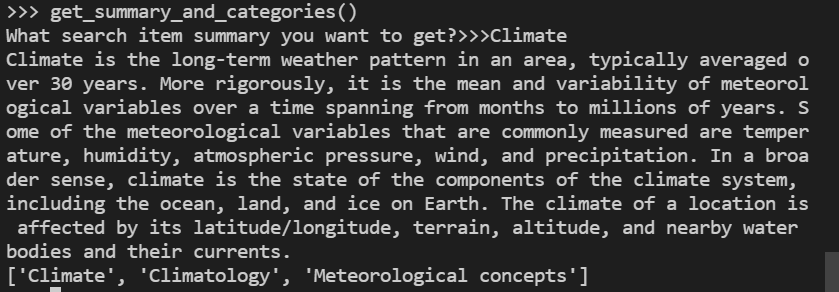
I designed functions to accomplish my analysis of the text. I mainly use the mediawiki package and codes we write to analyze books during the class. I first parse the text from Wikipedia using one function, then use this function to create a dictionary that contains each word and its frequency. Then using the dictionary to analyze the total words and different words on each wiki page. I also want to get the most common words on a page but not stopwords. As a result, I use the stopwords text in our class and exclude it when analyzing the most common words. Another important feature I want to include is to analyze what are the words that appear in each text that don’t appear in other texts. In order to achieve this, I create another dictionary that takes another requested page that users want to compare to. Returning the key that does not on the other page. I also make several functions that users can type in the keyword and get the coordinates and main images of that word from Wikipedia.

When analyzing the frequency, I need to access the text of the page. I have a few options including importing the text to a file and process that file, or directly creating a function that uses the library to get the page. And when I write the frequency function I could directly use the function to access the content of the page. I choose the second one since I think it is more precise and concise. The code also takes up less space to run. Also, we comparing two different pages, I do not want too many words listed out. So I plan only to display the first ten words that one page has and the other does not. I could use while or add a count under if condition. In the end, I choose to add a count under the if condition since I find it is less confusing and the revision is also less than the while loop. I run into problems when I try to use while because the place to put while is always a challenge to me.

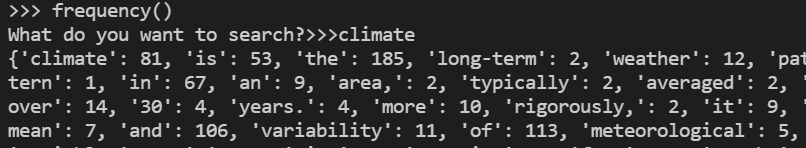
**3. Results [~2-3 paragraphs + figures/examples] Present what you accomplished:**

* **If you did some text analysis, what interesting things did you find? Graphs or other visualizations may be very useful here for showing your results.**
* **If you created a program that does something interesting (e.g. a Markov text synthesizer), be sure to provide a few interesting examples of the program's output.**

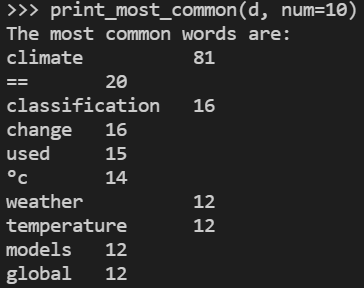
I find several interesting things after I create the program. First, users could search for anything they want and get back the summary and category using Wikipedia by just typing in the keywords:



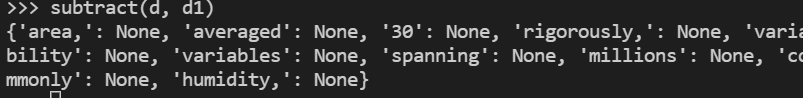
They could also get the frequency of the searched item page:



Users could also get common words excluding stopwords:

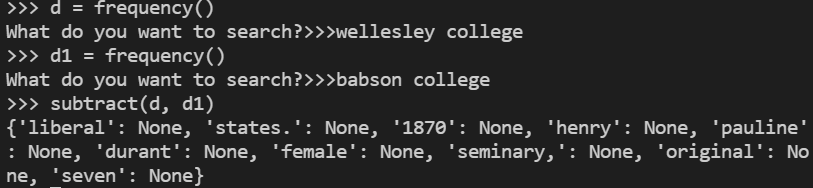


Another interesting example is that users could compare confusing words to see their difference by seeing the words that one has and the other does not. Here is the example of the first ten words that climate has but the weather does not:



From the output, we can get the idea that compared to the weather, the climate is a long-term and spanning system that is common and averaged.

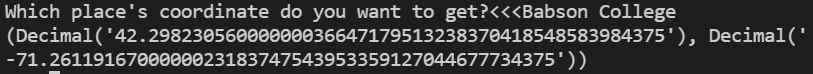
Another example would be Babson College and Wellesley College:



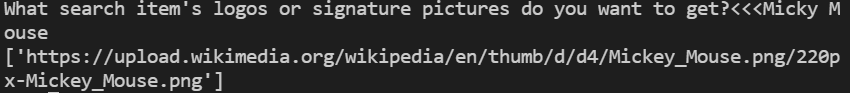
We can see that compare to Babson, Wellesley is a seminary and liberal college at the beginning, and it is a female college and has a longer history than Babson.

Users could compare anything they want using the program and they could get a quick general idea of what are the major differences between the two search items.

There are other interesting features that users could use is get coordinates of the searched place:



Main pictures or logos of the searched item, for example, Micky Mouse:



Click on the URL you will get:



This is especially useful when you want to use the logo of an item and this would save you time by searching on google.

**4. Reflection [~1 paragraph] From a process point of view, what went well? What could you improve? Other possible reflection topics: Was your project appropriately scoped? Did you have a good plan for testing? How will you use what you learned going forward? What do you wish you knew before you started that would have helped you succeed?**

After finishing the assignment, I think the planning process went well. But there are unexpected minor errors that take up an unproportioned amount of time. For example, the code could not read the import library and I used two hours to fix the problem. I think next time I would start the assignment even earlier to guarantee I have enough time to deal with unexpected errors. Besides that, I also feel the joy of creating a program and making it run successfully. This is also the first time I deeply realize the practical use of python code and how we could use programming to improve our life quality.