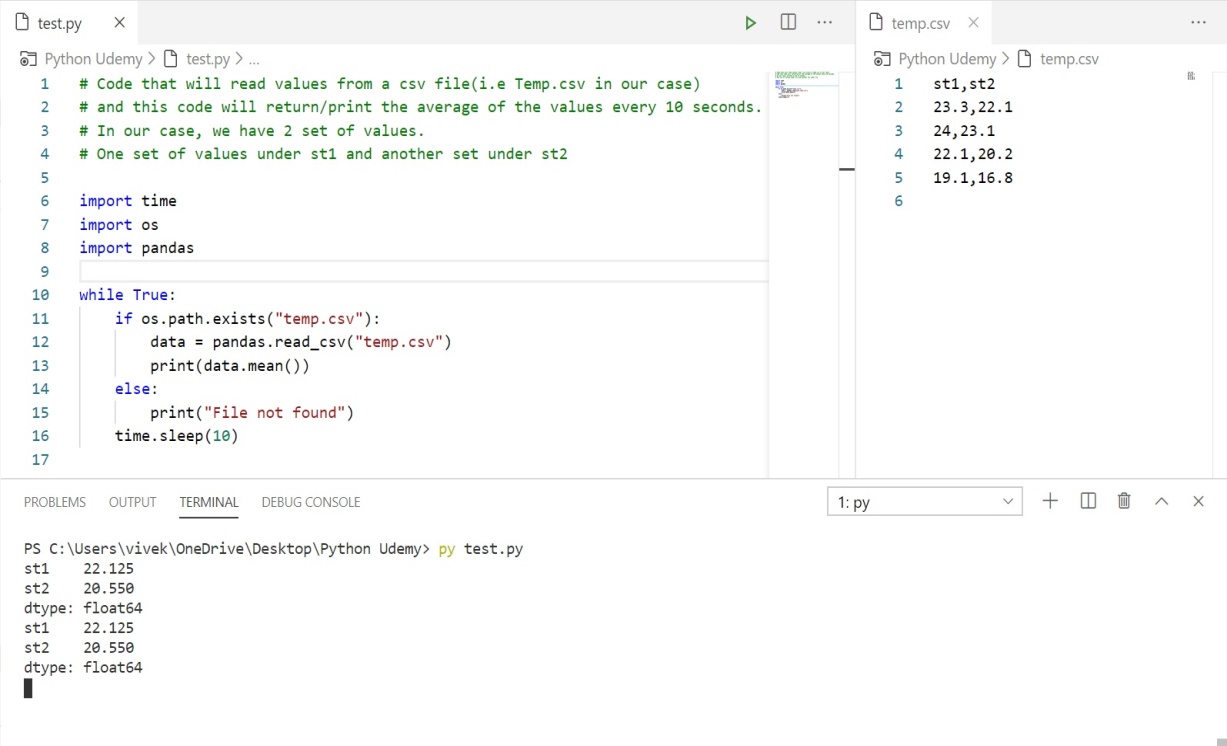
1. About "Pandas Lib in Py" - <https://towardsdatascience.com/a-quick-introduction-to-the-pandas-python-library-f1b678f34673>

According to the Wikipedia page on Pandas, “the name is derived from the term “[panel data](https://en.wikipedia.org/wiki/Panel_data" \t "_blank)”, an [econometrics](https://en.wikipedia.org/wiki/Econometrics" \t "_blank) term for multidimensional structured data sets.” But I think it’s just a cute name to a super-useful Python library!

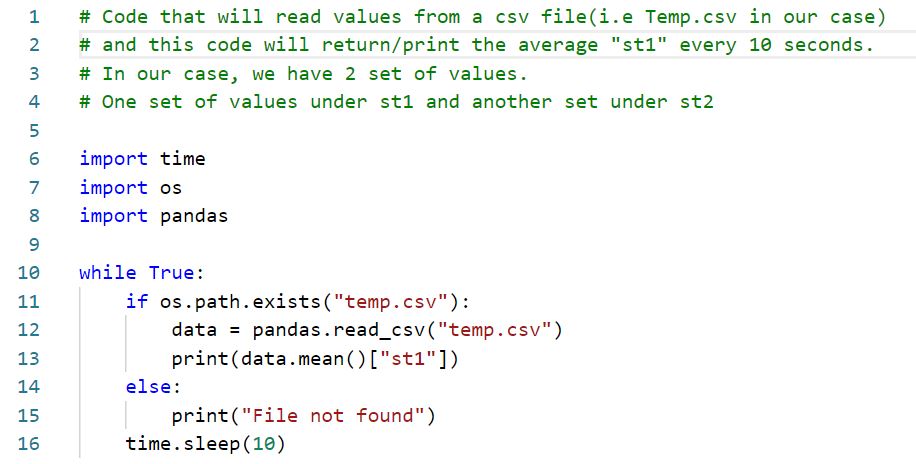
What’s cool about Pandas is that it takes data (like a CSV or TSV file, or a SQL database) and creates a Python object with rows and columns called **data frame** that looks very similar to table in a statistical software (think Excel or SPSS for example. People who are familiar with R would see similarities to R too). This is so much easier to work with in comparison to working with lists and/or dictionaries through for loops or list comprehension (please feel free to check out one of my [previous blog posts](https://medium.com/@adi.bronshtein/the-first-week-of-general-assembly-dsi-and-some-basic-python-6148099be7c0" \t "_blank) about very basic data analysis using Python. It would have been so much easier to do what I did there using Pandas!).



* The above code reads a .CSV file containing a few of the day’s temperatures. The .CSV file is containing two set of values, one set under “st1” and the other under “st2”. The code calculates the mean of st1 and st2 separately and gives the output as we see in the above screen shot.

Pandas lib allows us to do this function of accessing the CSV file and recreating it or taking actions on the file.

* What do we do if we want the mean of only “st1” or “st2”?
* The below given code will do that:

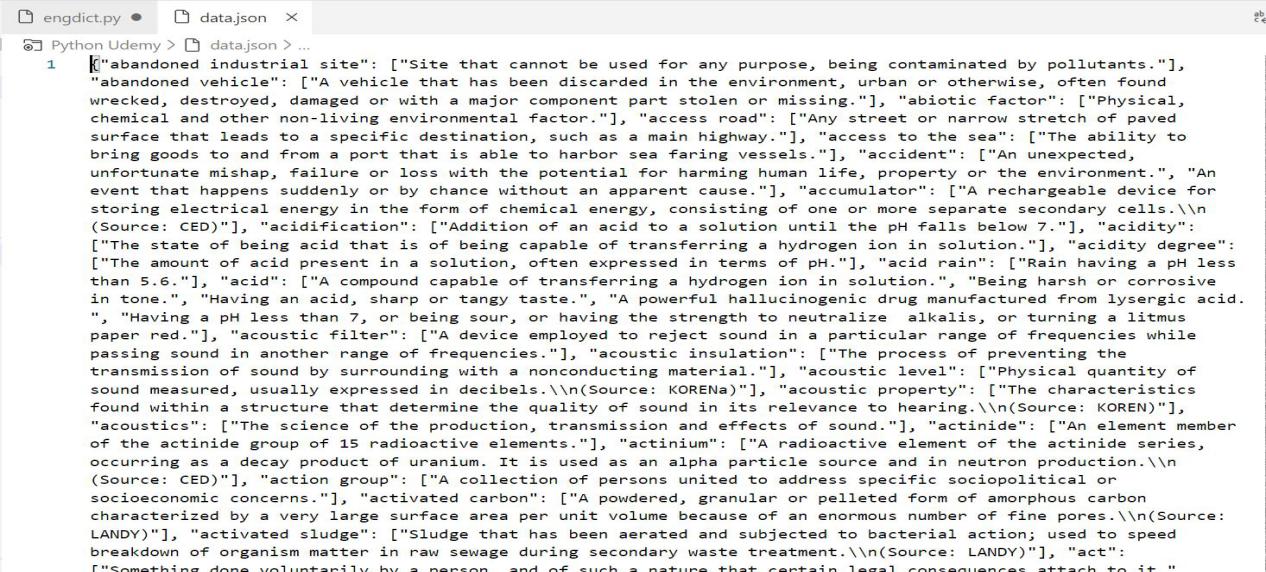


Application development

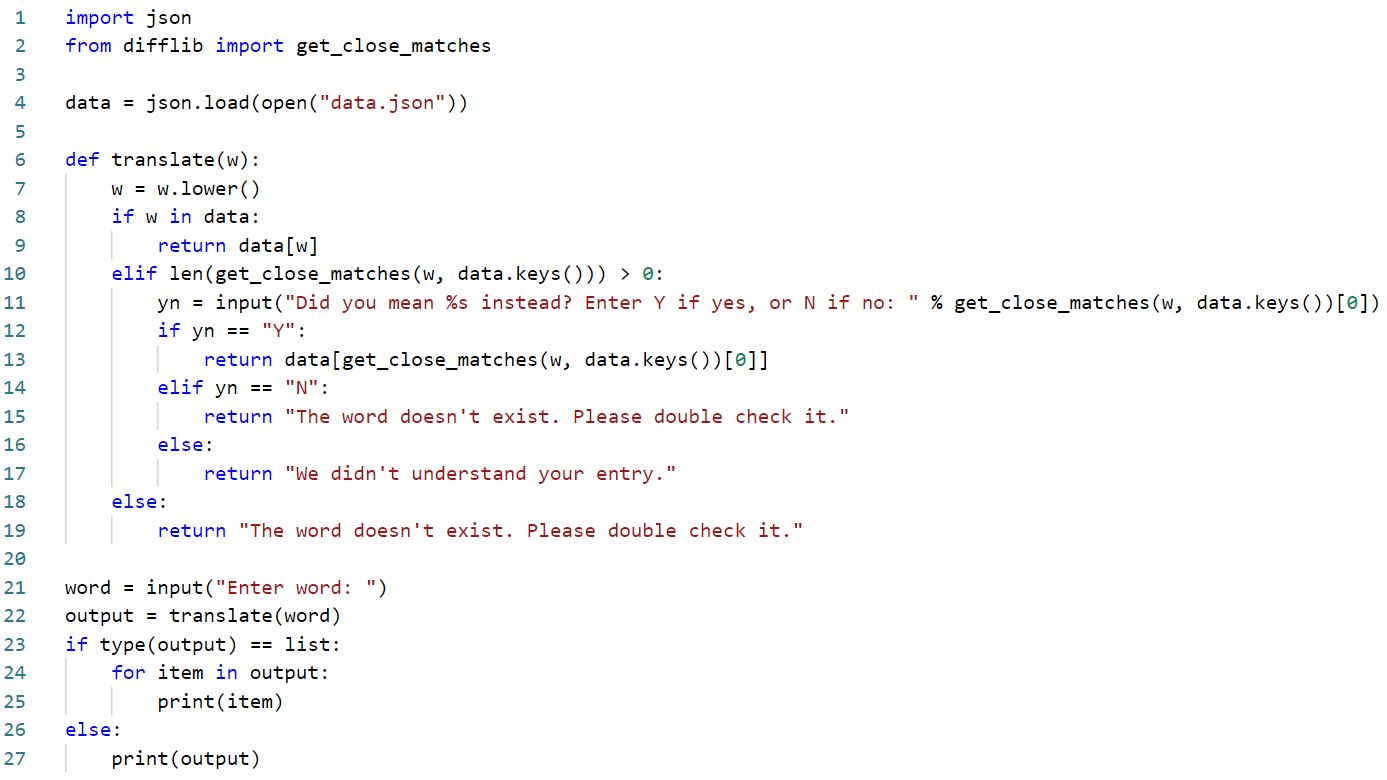
1. **Interactive English Dictionary** -
2. Version 1.0: The Basics….

* We first download a “.JSON” file which contains the words and the meanings.

Below image is to show how a JSON file looks:

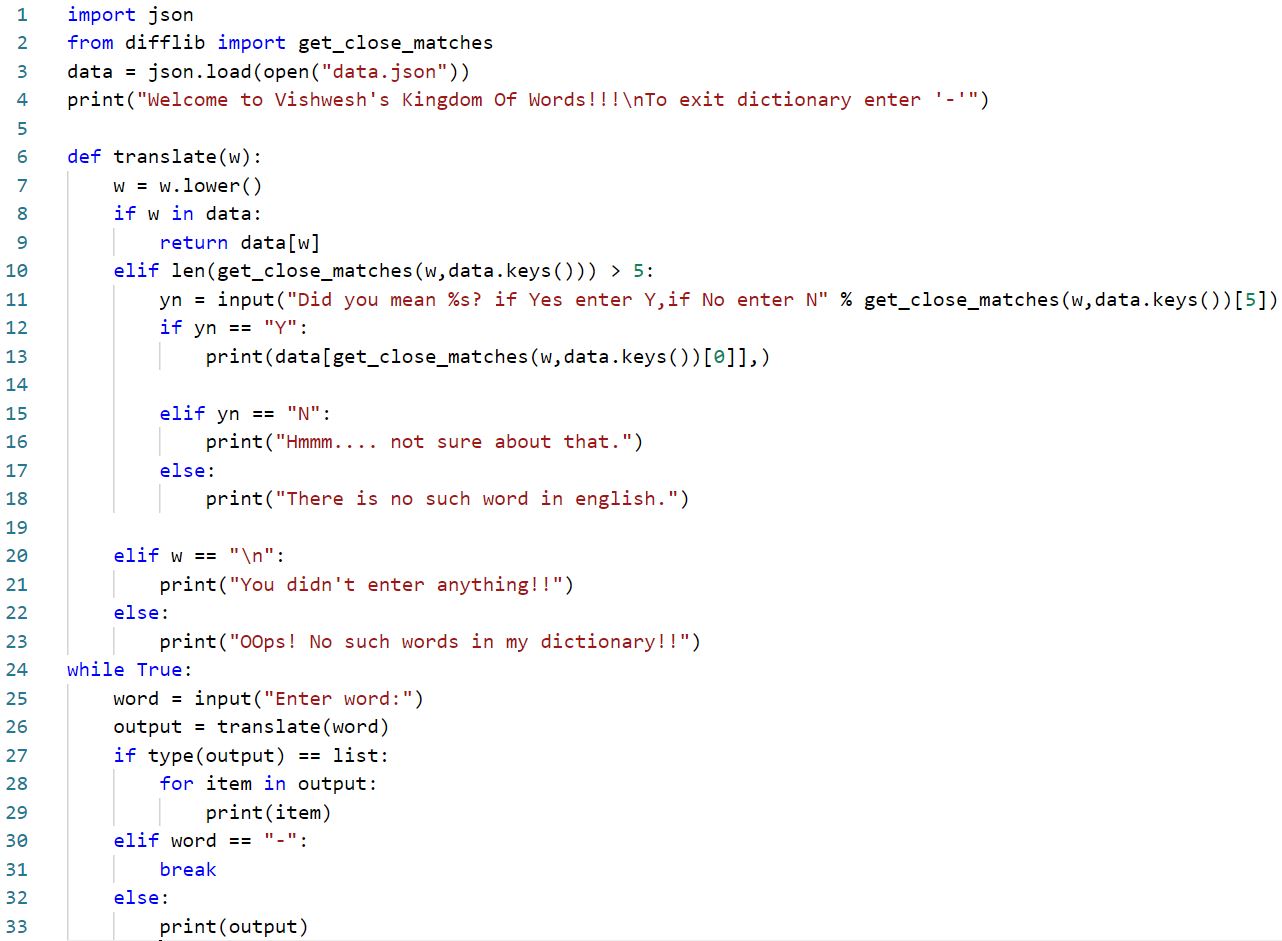


* We will then load this JSON file into the Python dictionary. Data = json.load(open(“data.json”))
* And then we go on to write the code that can read the JSON file, find out the meaning of user entered word from the JSON file.



Other important things to consider are:

* User input can be in Capitals/small letters/ combination of both. The letter case must not cause error. - (line 7 of code).
* If the user enters a word that is not there in the dictionary, the code must not give out an error to the user. (line 10 of code - uses “get\_close\_matches” function under the library of difflib).



My Code

|  |
| --- |
| **PROBLEM STATEMENT:**   1. **The dictionary provided as resource in the Udemy course did not fetch meanings for many words like “Grade”, “Bengaluru”, etc… as per user experience.** 2. **The .JSON file for the first dictionary was provided as resource in the Udemy course.**   **I downloaded another .JSON file of compact Webster’s English dictionary and included this new data as data1 in the code. It was structured in such a way that the entered word would first be searched for in the 1st dictionary and then if not found would go for the second dictionary. So I now have 2 dictionaries in the code. And the dictionaries cover all the words.**   1. **After searching for the meaning of a word, the code would terminate and we had to execute the code again to access dictionary. This would kill users time. So I included a while loop in the code so that the code would run until the user wanted to terminate the code. To terminate the code I initialized “-”, so when ever the user typed “-” while searching the word, he/she will terminate the code.** 2. **As the code asks for clarification whenever there is a typo by the user, by adding another dictionary to the code, we can have a larger range of close matches / similar words.**   **Improvised code:**  **123**  **. .**  **. .**  **. .**  **. .**  **456**  **In addition:**  **If we wanted the code to also recognize and give meanings of acronyms like “NATO”, “USA”,”ASEAN”……so on. We could use:**  **12345**  **(Using this after line 11 of above code)**  **The Biggest problem now, at least the one I can see is that - .JSON files are very large and can affect the functionality of the application significantly.**  **An alternative to accessing all the large amount of data from a .JSON file is to use a separate “DATABASE”.**  **We could make use of a MySQL database. So this idea takes us one more level higher and closer to making our dictionary application more user friendly!! Let’s move on then!!**    **mysql_database_connection_in_python-300x180** |