**SCRIPT1**

Contributors:

#### Anika Berger (anberger@clarku.edu)

#### Qi Zhang (qizhang3@clarku.edu)

**PURPOSE**

We used online Python compiler (https://repl.it/languages/python) to write the code. This code calls the NATIONAL WEATHER SERVICE website ([https://forecast.weather.gov](https://forecast.weather.gov/)) by entering the latitude and longitude of the target city. The output is a five-day weather forecast for the target city.

**REQUIRED INPUTS**

Longitude and latitude of the target city in decimal degrees.

#### **EXPECTED OUTPUTS**

The script returns the 5-day forecast for the target city, in uppercase letters.

#### **DETAIL SUPPLEMENT**

The script extracts information from multiple elements listed under the same class name using the BeautifulSoup library. First, the user enters the latitude and longitude of the target city. The code converts the latitude and longitude values to strings to generate the URL for the selected location. The script outputs a five-day weather forecast for the target city.

In order to display the weather more clearly, we manipulated the strings by adding spaces and commas where necessary. We observed the characteristics of the output text and used replace() to replace the parts without spaces and added commas in "high" and "before". (E.g: day = day.replace('Low', ', Low')) We used the .upper() function to capitalize all letters.

**PROCESS**

Our final version of the script functions as expected, though we tried another method for the string manipulation aspect which did not work in the end. We tried to use a loop to find every uppercase letter, put a space before the uppercase letter, and replace the two spaces with a single space. This process, however, cannot be implemented with replace(), and replace() cannot change the original string and list. We should create a new list or string to hold the changed text.

**FUTURE USE**

Though we may not need to scrape weather data in the future, the ability to web-scrape will be valuable to have in our skill set. Other aspects of the script will also be helpful for future coding ventures, such as string manipulation and knowing how to capitalize the letters of a string. As we learned in class, the .upper and .lower functions can be beneficial when dealing with big data that has inconsistent capitalization. Exposure to these functions now could benefit our future coding careers.

**SCRIPT2**

#### Contributors:

#### Anika Berger (anberger@clarku.edu)

#### Qi Zhang (qizhang3@clarku.edu)

**PURPOSE**

We generated two ideas based on script1 and lab3. The first was to add a temperature function based on script1 to determine whether the temperature is cold or hot, then give some suggestions for how to dress for the weather. Another idea was to use python3.5, PYQT5, and Pycharm to create a window, linkl the “openweathermap “website ([https://openweathermap.org](https://openweathermap.org/))via api and url, and use the window to display the weather forecast and dressing suggestions for the target city.

**REQUIRED INPUTS**

Name of the target city.

Longitude and latitude of the target city in decimal degrees.

#### **EXPECTED OUTPUTS**

The script2 returns the forecast for the target city and suggestions for how to dress.

#### **DETAIL SUPPLEMENT**

In the file named “script2.py”, we added a function based on script1 to classify the temperature into four categories: hot, warm, cool, and cold. The user inputs the longitude and latitude of the target city, and the script returns the weather information for each day, and displays the dressing suggestions according to the temperature. For example, when the temperature is 100 degrees Fahrenheit, the interface displays "It is hot. We recommend you to wear T-shirts, shorts and skirts. Be careful not to have heat stroke!"

The file “another\_try.ui” contains the window information I created using PYQT, which contains the controls I added. The file named “another\_try.py” contains the code that the ui file converted into code. First, the user enters the city name and then clicks the OK button. The display window will show the weather and dressing recommendations for the day.

**PROCESS**

For the file named “script2.py”, we created a string to store the dressing suggestions. At first, we used the print function to output a string, but the result displayed "None". We used the return function to make the result correct.

We encountered challenges when trying to implement the window idea. One issue was that many of the apis we tried cost money to use. In the end, we chose Openweathermap to get information, but we also encountered challenges here. There is a problem with using the city name to get information; if there are multiple cities with the same name as the target city, it will not display successfully. To solve this problem in the future, we could use the zip code to get weather information instead of the city name.

In this project, we both learned new things and gained new skills. Encountering challenges caused us to do our own debugging and problem solve, which left us with more experience in the end.

#### **Resources**

1.<https://github.com/jareddalessandro/Weather-openweathermap>

2.<https://github.com/arjun-thakor/python-weather/blob/master/weather.py>

3.<https://github.com/clement-mwendwa/WeatherByCity/blob/master/interact.py>

4.<https://openweathermap.org/api>

5.<https://www.tutorialspoint.com/pyqt/index.htm>

6.<https://www.runoob.com/python/python-tutorial.html>