CS217 HW 6 – JSON

To get started:

Step 1. Go to https://rapidapi.com/axisbits-axisbits-default/api/covid-19-statistics/endpoints. This is the data API that we will use in this assignment. You will need to register a free account before using it (click sign up on the top right).

Step 2. On the left side, choose "GET reports"; on the right side, you should change from Nodejs to **Python** with **Requests** library. This will show a sample code snippet on how to use their API. The code snippet uses the package requests in python. There are four variables:

`url` – this defines the url links to download data.

`querystring` – this gives a temple to query to subset of data you want to download. It allows us to specify what kind of data we want from the server. It can be e.g, city-level, state-level, etc.

`headers` – this defines the host and key to access data. Each account should have a different key.

`response` – this is the result of the request. `response.text` is a string in the JSON format that you can convert to a python object. It provides various data attributes of COVID-19. In this assignment, we will mainly focus on two values:

- `deaths` – an integer referring to the cumulative death count
- `confirmed` – an integer referring to the cumulative confirmed count

Q0: Copy the sample code snippet below Q0 in the py file. When executed, it should output a JSON string with COVID-19 information of Alabama on 2020-04-16. (Please make sure you understand the structure of this JSON string.)

Q1 (1pt): Write a function `death_count_il` to extract the cumulative death count of Illinois recorded till 2020-11-16. The function takes no input and returns an integer.

Q2 (2pt): Write a function `get_death_count` that returns the cumulative death count of a given state till a given date. The function takes two string variables `state` and `date` and returns an integer.
Hint: there are some cases where no data can be retrieved. It means that the death count is 0 for this state and date. We should return 0 in this case.

Some samples for you to debug/confirm your code:

```
print(get_death_count("Illinois", "2020-04-07"))
```

output: 308

```
print(get_death_count("Illinois", "2019-04-07"))
```

output: 0

Q3 (2pt): We want to explore the cumulative confirmed cases in a given state and set of dates.

1. First, implement the function `get_confirmed_count`, it is similar to the function `get_death_count` in Q2 (two string inputs and returns an integer), but returns the cumulative confirmed cases instead of death cases.
2. Then implement the function `get_from_dates`. It takes the name of a state in a string and a list of string as input (each string in the list is for a date in the form of "xxxx-xx-xx"), and it returns a list of integers, each showing the confirmed cumulative cases of a given date. You can make use of the function `get_confirmed_count`.

Some samples for you to debug/confirm your code:

```
week1=['2020-03-01','2020-03-02','2020-03-03','2020-03-04','2020-03-
05','2020-03-06','2020-03-17']
print(get_from_dates('California', week1))
```

output: [0, 0, 0, 0, 0, 0, 698]

```
week2=['2021-03-12','2021-03-13','2021-03-14','2021-03-15','2021-03-
16','2021-03-17','2021-03-18']
print(get_from_dates('New York', week2))
```

output: [1724425, 1734676, 1741363, 1748482, 1755193, 1761775, 1767290]

Q4 (2pt): Write a function `get_us_confirmed_count` to obtain the cumulative confirmed cases till a specific date across the US. It takes a date in the string format and returns an integer of the confirmed cases in the US by that date.
Hint: we are looking at country-level data now. So in your `querystring`, you should specify the `iso` and `date` variables only. After you get your json object, you will need to analyze the structure and obtain the sum of confirmed cases in each state.

Some samples for you to debug/confirm your code:

```
print(get_us_confirmed_count('2020-03-01'))
```

output: 76

```
print(get_us_confirmed_count('2021-03-01'))
```

output: 28663108

Q5 (2 pt):
We define the surging factor as (new_death * new_confirmed). Write a function that return the surging factor in a given state for a given period. The function name is `surging_factor` and should take three strings as input: `state`, `start_date` and `end_date`. You can write helper functions. Note: the "new death" count should be the difference between the death count of the last day and the death count of the first day of a given period.

Some samples for you to debug/confirm your code:

```
print(surging_factor("California", "2020-05-16", "2020-05-16"))
```
output: 0

```
print(surging_factor("California", "2020-05-16", "2020-05-17"))
```
output: 46112

Q6 (1pt): With the five functions defined in the previous questions, please do some analysis and write two or three paragraphs to briefly summarize your findings. For example, you can discuss how the pandemic has changed over time in a specific area (like your home state) and the whole country.

Submission:

You should create a folder and name it as hw6_yourNetid. For example, if your netid is abc1234, your folder name should be `hw6_abc1234`. Then put the following files in the folder:

- **Q0 to Q5**: The given py file (hw6.py) includes the template for Q0 – Q5. You need to implement the functions as required. Please do not change any function names or your code will not pass our tests. Please do not include extra printing statements, you py file should only include the copied snippet code and the functions.
- **Q6**: please put your summary in a pdf file.

Package you folder as a zip file. After unzipping your submission, the directory structure should look like the following.

```
hw6_netid/
hw6_netid/hw6.py
hw6_netid/summary.pdf
```