In this assignment you are given 26 text (json) files, each containing TripAdvisor reviews for a particular hotel together with some information about the hotel. The information in each text file is stored in the *json format*. JSON stands for Java Script Object Notation. Like xml, it is a format for storing and exchanging data. A *json* object looks very much like a python dictionary in which the keys are strings. In fact, you are going to be using the load method in the Python json library to read the contents of each file as a Python dictionary and assign it to a variable for later use.

Consider the code below for reading the *72572.json* file into the variable jsondat.

```
import json
with open('72572.json') as input_file:
    jsondat=json.load(input_file)
```

By the way, note the JSON file extension. That is just a convention. The load method ignores that and just looks at the *contents* of the text files. As long as it is in the JSON format we are okay. We can verify that jsondat really is a dictionary and that it has two keys.

```
type(jsondat)
Out[1]: dict
jsondat.keys()
Out[2]: [u'Reviews', u'HotelInfo']
```

The value associated with the ***HotelInfo*** key is itself a dictionary with *four* (4) keys:

```
jsondat['HotelInfo']

{   'Address': '<address class="addressReset"><span rel="v:address"><span '
               'dir="ltr"><span class="street-address" '
               'property="v:street-address">2420 West Thomas Rd</span>, <span '
               'class="locality"><span property="v:locality">Phoenix</span>, '
               '<span property="v:region">AZ</span> <span '
               'property="v:postal-code">85015</span></span> '
               '</span></span></address>',
    'HotelID': '73768',
    'HotelURL': '/ShowUserReviews-g31310-d73768-Reviews-Days_Inn_I_17_Thomas-
Phoenix_Arizona.html',
    'ImgURL': 'http://media-
cdn.tripadvisor.com/media/ProviderThumbnails/dirs/b4/15/b4153483cffba437d2123bda08b8fe154
large.jpg',
    'Name': 'Days Inn I-17 & Thomas',
    'Price': '$50 - $66*'}
```

The value associated with the ***Reviews*** key is a list of 233 reviews, in this case, each review stored as a dictionary.

```
type(jsondat['Reviews'])
Out[3]: list
len(jsondat['Reviews'])
Out[4]: 233
```

Let us look at the first review (dictionary) for this particular hotel.

```
jsondat['Reviews'][0]

{   'Author': 'Ricardo G',
    'AuthorLocation': 'Harlingen, Texas',
    'Content': 'One of the best things that I encountered at this Days Inn was '
               'the attentiveness of the front desk staff. From there on it '
               'was unsatisfactory for me. The building is detached from the '
               'front desk area and is old, has no elevator and not in the '
               'best neighborhood. The room that I stayed in was ok and could '
               'have been a little cleaner. The bed sheets looked rather worn '
               'out, but, the bed was comfortable. The TV is not flat screen '
               'and the reception was not all that great. One of the big '
               'pluses is that I had a microwave oven and small fridge which '
               'came in handy. The A/C is rather noisy, but, works just fine '
               'and kept my room comfortable. Would I stay at this hotel '
               'again? The answer is no. There are a lot more hotel choices '
               'around this area that look more promising although a bit more '
               'expensive. Hope this helps.',
    'Date': 'March 25, 2012',
    'Ratings': {   'Cleanliness': '3',
                   'Location': '2',
                   'Overall': '3.0',
                   'Rooms': '2',
                   'Service': '5',
                   'Sleep Quality': '3',
                   'Value': '2'},
    'ReviewID': 'UR126635779',
    'Title': '“Not to my Satisfaction”'}
```

Each of the 26 files has a similar structure with the number of reviews varying from hotel to hotel (and you might consider the possibility that a particular hotel's list of reviews is empty, i.e. that there are no reviews for that hotel). Note that each review is a "nested dictionary", i.e. you have a dictionary with 'Ratings' key inside of it. The value of this key are the ratings by that reviewer. Note that one or more "rating category" key/value might be "missing" from a particular review—only the key/value pairs for the ratings categories a particular review assigned a rating to will appear in a reviewer's "ratings" dictionary.

In the *first part of the assignment*, you want to combine the data in *all* the hotel files to obtain a "hotel ratings" DataFrame for *all* the hotels put together, say ***hotelRatingDF***. This DataFrame should contains the Hotel ID, Hotel Name, Review Date, Review ID, Author Name and each rating given by each reviewer in its own column:

```
hotelRatingsDF.columns
Index(['HotelID', 'HotelName', 'Author', 'Date', 'Business service',
       'Business Service', 'Check in / front desk', 'Cleanliness', 'Location', 'Overall',
'Rooms', 'Service', 'Sleep Quality', 'Value', 'ReviewID','Title'],
      dtype='object')
```

The hotelRatingsDF frame should contain 2485 records since there are that many reviews in *all* the json files put together. In other words, that is the total number of reviews for *all* of the hotels. Using this frame, you then want to compute some "statistics" for each of the ratings columns. Note that you are computing statistics for all the hotels at once (for each rating category).

By the way, you are required to write a loop to open each of the 26 json files and "do something with them". There are various ways of doing this, e.g. https://www.tutorialspoint.com/python/os_listdir.htm. Inside you loop you can obtain the list of reviews from each file you open, convert the list to a DataFrame and then add that DataFrame to (growing) list of DataFrames (26 when all is said and done). After the loop, you can concatenate the list of (26) DataFrames to obtain a DataFrame containing *all the reviews for all the hotels*. Notice what I am leaving out here: the hotel information. You could add the hotel info to each "ratings" DataFrame in the loop but that is not the only way to go about it. I would also suggest saving the comments (content key values) in the reviews for each hotel in the loop while you are at it, say in a dictionary with the "hotelID" as keys. You will need to work with these comments in the *second part of the assignment*. You could instead keep the ratings and comments together in a DataFrame and then separate them later (after concatenating them, say). It is up to you. I suggest before you write a single line of code that write a plan for how you are going to be processing the data.