We will be examining data from health inspections for restaurants, nursing homes, and schools in Prince George's County. Food_inspection.csv.

Source: https://data.princegeorgescountymd.gov/Health/Food-Inspection/umjn-t2iz

Exercise #1 - Data Cleaning

Step 1.1 - Read in the data. Create a table of the unique values of Category. Which values do you believe represent a restaurant? Explain which categories you chose in a markdown cell. Create a single dummy variable for restaurant that combines multiple values from Category.

Step 1.2 - Convert the Inspection_date column into a datetime column. Create a new column for the year of the inspection. Create a new column for the month of the inspection. Create a column for the year and month.

Step 1.3 - For each column with the type of compliance, e.g. "Rodent and Insects," create a dummy variable that is 1 if the establishment is out of compliance and 0 otherwise. Use np.nan for not applicable. Drop the string columns (retain only the dummy variables).

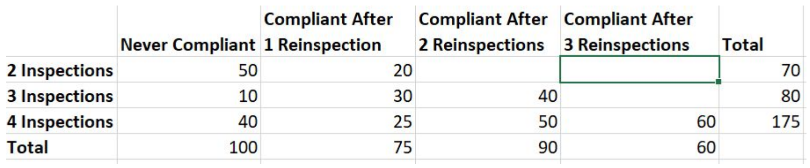
Step 1.4 - Create a new column that contains the number of violations for that inspection (the number of categories where the establishment was not in compliance). Create a dummy variable that is 1 if the establishment is out of compliance in any category.

Step 1.5 - For establishments with multiple inspections, create a new DataFrame in wide format. Keep only the establishment ID, Category, Inspection_date, and number of violations. Make sure category is consistent within ID and resolve any discrepancies if necessary (i.e. each establishment has only one category). Reshape from long to wide (pivot) such that each establishment is a row and you have a column for the date and number of violations for inspection 1, inspection 2, inspection 3, etc.

Exercise #2 - Summary Statistics/Grouped Data

Step 2.1 - What is the most common type of violation? The compliance categories are not mutually exclusive because one restaurant can have multiple violations. Create a table with the number of violations by violation type. Sort the table from the most common to least common violations.

Step 2.2 - For establishments with multiple inspections, how many reinspections does it take for an establishment to become compliant? Create a table where each row is the number of inspections a restaurant has had and the columns are the number of reinspections until the establishment becomes compliant. Write 2-4 sentences with your observations. A mock up of this table is below (you will have more rows).



	Never Compliant	Compliant After 1 Reinspection	Compliant After 2 Reinspections	Compliant After 3 Reinspections	Total
2 Inspections	50	20			70
3 Inspections	10	30	40		80
4 Inspections	40	25	50	60	175
Total	100	75	90	60	

Exercise #3 - Data Visualization

For the data visualization tasks below, you may choose to use any Python visualization package you wish. Make sure all graphs are labeled appropriately. Limit your analysis to restaurants using the dummy variable indicator you created in 1.1.

Step 3.1 - Create a bar graph showing the results of 2.1.

Step 3.2 - Create a line graph that shows the percent of restaurant inspections that have at least one violation by month and year. Are inspections getting harder or easier over time? Is there a particular month where more restaurants pass? Write 2-4 sentences with your observations.

Step 3.3 - Create a map that shows all of the restaurants. Color the restaurants with at least one violation in red. Are there particular areas with more violations? If there are clusters of violations, either through interactive visualization or by manually inspecting the data, look at the types of violations where there are clusters. Are there any trends? Write 2-4 sentences with your observations. If you did not use interactive visualization, explain how you explored trends in violation type by area. You may also create a second map showing violation types.

Exercise #4 - All the photos! Sort of...

This assignment builds off of previous Exercise from weeks ago, scraping data from https://pokemondb.net

Step 4.1: Start with your deduplicated pokedex data. Just like in **Step 2.4 from HW #4**, create a new column in the DataFrame called *sample2* that tags every 12th pokemon. (This will reduce our sample size.)

Step 4.2: Scrape the photo URLs for each pokemon in sample2 like in **Step 3.1 from HW #4**.

Step 4.3: Display the photos for all of the sample2 pokemon in your Jupyter notebook. Example code for how to display multiple images in a loop is below.

```
Display(Image(url=_____, unconfined=True))
```

Exercise #5 - Rescrape and Clean the Location Table

Step5.1: Using BeautifulSoup, scrape the Location table **for Bulbasaur** properly and add it to a DataFrame. When multiple columns are combined, separate the columns and duplicate the location information. For example, the column "Red Blue" that contains "Pallet Town" will become two columns-- "Red" with location "Pallet Town" and "Blue" with location "Pallet Town." Transpose the DataFrame so the video game (e.g. "Red") is the column name.

Step 5.2: Generalize Step 5.1 for all pokemon in the original sample (the sample from **Step 5.1 from HW #4).** Append all the data together. If information is not available for a pokemon in a particular game, the column should contain NaN. Display the entire table.

Exercise #6 - Location Analysis Done Right

Step 6.1: Join the pokedex data to the location DataFrame created in Step 5.2 above. (Exclude pokemon that are not in the sample.) For the locations in pokemon game X, calculate the average total points for each location. Which location has the highest average total point score?