December 12th, 2022
SI 206 Fall 2022
Lindsey Bellowe, Leah Feilbogen, Kiran Krishnan
GitHub Repository: https://github.com/lbellowe/206-Final-Project

206 Final Project Report ~Covid Weather Women

Project Goals

Our overall project theme was factors that impact the spread of Covid-19. We were interested in identifying a possible relationship between United States Covid-19 and weather data. To accomplish our goal, we gathered data on the total number of Covid-19 deaths and confirmed cases per month per state, the average high and low temperatures per month per state, and the overall community risk levels (how likely you are to contract Covid-19 in a particular area) per state. Below is a specific breakdown of the APIs and Website we used to collect the data.

1. Leah used Beautiful Soup to collect the high and low temperatures for each state for each month from Extreme Weather Watch's website. Leah then performed four calculations based on the data she collected for each month; (1) the state with the max high temperature, (2) the state with the min high temperature, (3) the state with the max low temperature, (4) and the state with the min low temperature.
2. Kiran used an API to collect the number of confirmed cases and deaths that occurred on each day starting April 1, 2020 until March 31, 2021 for each state. Kiran then performed four calculations; (1) month with the most confirmed cases per state, (2) month with the least confirmed cases per state, (3) month with the most deaths per state, (4) and month with the least deaths per state.
3. Lindsey used an API to collect the overall risk level by state which is rated either a 1, 2, 3, or 4 depending on if the risk is low, medium, high, or extreme (the risk level determines how likely you are to contract Covid-19 in a particular area). She ultimately found the average risk values from the top 3 most populous counties in each state. Lindsey then calculated the average risk levels per state.

Goals we Achieved

Each member of the group was able to successfully obtain meaningful data (specified above) from their respective API or Website/Beautiful Soup. After a lot of hard work, we were able to store 25 data points at a time in each of our tables in the database. Eventually, after running the program a total of 25 times, each table with the exception of the "Month" table has at least 100 rows worth of data. Please note that the "Month" table does not contain any of the data we collected using the APIS or Beautiful Soup which is why it does not contain 100 rows, and it is the table we used to accomplish the

join portion of the project. We were also able to perform calculations (specified above) and begin exploratory analysis into a relationship between various aspects of Covid-19 and weather in different US states. Additionally, we wrote our calculations into csv/txt files and created visualizations that reflected this data. Finally, the group learned how to obtain data and begin a project from start to finish; this project will be useful especially as all three group members are on the Information Analysis pathway and will continue to utilize the skills developed throughout our future.

Problems we Faced

The main problem our group faced was limiting the number of data points being stored to 25 each time the program runs when using Beautiful Soup. In the case of the Website where we used Beautiful Soup, the way we were calling the functions (one html at a time) made it challenging to limit the number of data points being stored. After many office hour sessions, we were able to solve this challenge, but this was by far the most difficult portion of this project for our team. Additionally, there was some confusion when it came to the project's directions and the actual requirements of the project. For example, we did not know that it was required to use requests.get for the APIS until we got that feedback after our in-class presentation / grading session. Lastly, we found it challenging to identify a fully developed conclusion based on the data we collected. There are many factors that affect the number of Covid-19 cases and deaths besides the weather. For example, population size and density, the availability of Covid-19 tests, the quantity of Covid-19 vaccines etc. might also be affecting factors. We realized that isolating one of several possible factors makes coming to a definite conclusion about our overall topic, factors that affect the spread of Covid-19, extremely difficult.

Files that Contain Calculations from the Data in the Database

The files that contained calculations from the database are included in our zipped files, but we have attached screenshots of the three files below as well as hyperlinked the specific GitHub page for each file.

weather.csv

```
Month,Max High Temp State,Max High Temp,Min High Temp State,Min High Temp,Max Low Temp State,Max Low Temp,Min Low Temp State,Min Low Temp
January,Hawaii,77,Alaska,10.7,Hawaii,62.5,Alaska,-2.4
February,Hawaii,76.8,Alaska,16.2,Hawaii,62.5,Alaska,1.2
March,Hawaii,77.3,Alaska,21.2,Hawaii,63.5,Alaska,3.5
April,Florida,82.1,Alaska,35,Hawaii,65.1,Alaska,17.4
May,Florida,87.5,Alaska,48.9,Hawaii,66.5,Alaska,31.2
June,Arizona,93,Alaska,59.9,Florida,70.9,Alaska,41.9
July,Arizona,95.2,Alaska,62.3,Florida,72.9,Alaska,45.9
August,Texas,94.6,Alaska,57.9,Florida,73.1,Alaska,42.8
September,Florida,89,Alaska,48.3,Florida,71,Alaska,34.5
October,Florida,83.6,Alaska,33.2,Hawaii,68.8,Alaska,21.1
November,Hawaii,80.2,Alaska,19.4,Hawaii,67,Alaska,7.3
December,Hawaii,78.1,Alaska,13.2,Hawaii,64.6,Alaska,0.7
```

first_api.txt

```
First API Calculations:
Month with the most confirmed cases per state:
[('Alabama', 'March', 15701638), ('Alaska', 'March', 1868257),
('Arizona', 'March', 25799613), ('Arkansas', 'March', 10140927),
('California', 'March', 112432256), ('Colorado', 'March', 13881333),
('Connecticut', 'March', 9132906), ('Delaware', 'March', 2815506),
('District of Columbia', 'March', 1321799), ('Florida', 'March',
61554055), ('Georgia', 'March', 31365608), ('Hawaii', 'March',
883054), ('Idaho', 'March', 5460659), ('Illinois', 'March', 37743564),
('Indiana', 'March', 20995935), ('Iowa', 'March', 10664455),
('Kansas', 'March', 9338079), ('Kentucky', 'March', 13036670),
('Louisiana', 'March', 13580984), ('Maine', 'March', 1472262),
('Maryland', 'March', 12271443), ('Massachusetts', 'March', 18804214),
('Michigan', 'March', 21170779), ('Minnesota', 'March', 15511379),
('Mississippi', 'March', 9323406), ('Missouri', 'March', 17646850),
('Montana', 'March', 3174117), ('Nebraska', 'March', 6374609),
('Nevada', 'March', 9280577), ('New Hampshire', 'March', 2460709),
('New Jersey', 'March', 26273580), ('New Mexico', 'March', 5848479),
('New York', 'March', 54454687), ('North Carolina', 'March',
27560439), ('North Dakota', 'March', 3140564), ('Ohio', 'March',
30773169), ('Oklahoma', 'March', 13408702), ('Oregon', 'March',
4966419), ('Pennsylvania', 'March', 30360800), ('Rhode Island',
'March', 4085755), ('South Carolina', 'March', 16594789), ('South
Dakota', 'March', 3565684), ('Tennessee', 'March', 24174360),
('Texas', 'March', 84649388), ('Utah', 'March', 11751648), ('Vermont',
'March', 531684), ('Virginia', 'March', 18535638), ('Washington',
'March', 10991162), ('West Virginia', 'March', 4227093), ('Wisconsin',
'March', 19431653), ('Wyoming', 'March', 1716641)]
Month with the least confirmed cases per state:
[('Alabama', 'April', 125166), ('Alaska', 'April', 8231), ('Arizona',
'April', 128668), ('Arkansas', 'April', 53571), ('California',
'April', 865585), ('Colorado', 'April', 264026), ('Connecticut',
'April', 467204), ('Delaware', 'April', 67480), ('District of
Columbia', 'April', 70950), ('Florida', 'April', 671158), ('Georgia',
'April', 453331), ('Hawaii', 'April', 14988), ('Idaho', 'April',
45937), ('Illinois', 'April', 802337), ('Indiana', 'April', 292473),
('Iowa', 'April', 83724), ('Kansas', 'April', 55555), ('Kentucky',
'April', 74780), ('Louisiana', 'April', 625639), ('Maine', 'April',
22358), ('Maryland', 'April', 327492), ('Massachusetts', 'April',
977699), ('Michigan', 'April', 818293), ('Minnesota', 'April', 65270),
('Mississippi', 'April', 109721), ('Missouri', 'April', 144460),
('Montana', 'April', 11538), ('Nebraska', 'April', 44916), ('Nevada',
'April', 97660), ('New Hampshire', 'April', 36795), ('New Jersey',
'April', 2186635), ('New Mexico', 'April', 49429), ('New York',
'April', 6328973), ('North Carolina', 'April', 168317), ('North
Dakota', 'April', 14825), ('Ohio', 'April', 287164), ('Oklahoma',
'April', 67883), ('Oregon', 'April', 50484), ('Pennsylvania', 'April',
822697), ('Rhode Island', 'April', 119603), ('South Carolina',
'April', 113247), ('South Dakota', 'April', 36231), ('Tennessee',
'April', 187332), ('Texas', 'April', 494349), ('Utah', 'April',
```

```
82277), ('Vermont', 'April', 21158), ('Virginia', 'April', 223913),
('Washington', 'April', 327552), ('West Virginia', 'April', 21315),
('Wisconsin', 'April', 119648), ('Wyoming', 'April', 10987)]
Month with the most deaths per state:
[('Alabama', 'March', 320023), ('Alaska', 'March', 9123), ('Arizona',
'March', 513727), ('Arkansas', 'March', 169225), ('California',
'March', 1747316), ('Colorado', 'March', 189756), ('Connecticut',
'March', 241425), ('Delaware', 'March', 46638), ('District of
Columbia', 'March', 32315), ('Florida', 'March', 1002705), ('Georgia',
'March', 550272), ('Hawaii', 'March', 13891), ('Idaho', 'March',
59611), ('Illinois', 'March', 719826), ('Indiana', 'March', 398621),
('Iowa', 'March', 174717), ('Kansas', 'March', 150082), ('Kentucky',
'March', 168065), ('Louisiana', 'March', 307155), ('Maine', 'March',
22405), ('Maryland', 'March', 250699), ('Massachusetts', 'March',
517635), ('Michigan', 'March', 520979), ('Minnesota', 'March',
210239), ('Mississippi', 'March', 213847), ('Missouri', 'March',
271622), ('Montana', 'March', 43507), ('Nebraska', 'March', 69535),
('Nevada', 'March', 158868), ('New Hampshire', 'March', 37340), ('New
Jersey', 'March', 742393), ('New Mexico', 'March', 119627), ('New
York', 'March', 1505538), ('North Carolina', 'March', 364351), ('North
Dakota', 'March', 46128), ('Ohio', 'March', 555880), ('Oklahoma',
'March', 146330), ('Oregon', 'March', 72516), ('Pennsylvania',
'March', 764787), ('Rhode Island', 'March', 79889), ('South Carolina',
'March', 275781), ('South Dakota', 'March', 59314), ('Tennessee',
'March', 358511), ('Texas', 'March', 1448057), ('Utah', 'March',
63056), ('Vermont', 'March', 6681), ('Virginia', 'March', 306755),
('Washington', 'March', 161026), ('West Virginia', 'March', 77397),
('Wisconsin', 'March', 222846), ('Wyoming', 'March', 21393)]
Month with the least deaths per state:
[('Alabama', 'April', 4044), ('Alaska', 'April', 173), ('Arizona',
'April', 4870), ('Arkansas', 'April', 1007), ('California', 'April',
30176), ('Colorado', 'April', 11715), ('Connecticut', 'April', 29752),
('Delaware', 'April', 1840), ('District of Columbia', 'April', 2651),
('Florida', 'April', 19495), ('Georgia', 'April', 18051), ('Hawaii',
'April', 274), ('Idaho', 'April', 1098), ('Illinois', 'April', 32642),
('Indiana', 'April', 16218), ('Iowa', 'April', 1947), ('Kansas',
'April', 2241), ('Kentucky', 'April', 3889), ('Louisiana', 'April',
32872), ('Maine', 'April', 843), ('Maryland', 'April', 13760),
('Massachusetts', 'April', 42952), ('Michigan', 'April', 59519),
('Minnesota', 'April', 3818), ('Mississippi', 'April', 3972),
('Missouri', 'April', 5056), ('Montana', 'April', 280), ('Nebraska',
'April', 866), ('Nevada', 'April', 4029), ('New Hampshire', 'April',
1009), ('New Jersey', 'April', 102708), ('New Mexico', 'April', 1472),
('New York', 'April', 425198), ('North Carolina', 'April', 4804),
('North Dakota', 'April', 296), ('Ohio', 'April', 12517), ('Oklahoma',
'April', 3692), ('Oregon', 'April', 1829), ('Pennsylvania', 'April',
30595), ('Rhode Island', 'April', 3459), ('South Carolina', 'April',
3284), ('South Dakota', 'April', 211), ('Tennessee', 'April', 3741),
('Texas', 'April', 12274), ('Utah', 'April', 722), ('Vermont',
'April', 958), ('Virginia', 'April', 7019), ('Washington', 'April',
```

```
16938), ('West Virginia', 'April', 515), ('Wisconsin', 'April', 5402),
('Wyoming', 'April', 82)]
```

second_api.txt

```
Second API calculations: Finding average risk level per state
{'AK': 1.3333333333333333, 'AL': 2.3333333333333335, 'AR': 2.0, 'AZ':
3.0, 'CA': 2.3333333333333335, 'CO': 2.0, 'CT': 2.0, 'DE':
1.3333333333333333, 'FL': 2.3333333333333335, 'GA':
1.6666666666666667, 'HI': 1.6666666666666667, 'IA':
2.6666666666666665, 'ID': 2.6666666666666665, 'IL':
2.3333333333333335, 'IN': 2.0, 'KS': 2.3333333333333335, 'KY': 2.0,
'LA': 2.0, 'MA': 1.0, 'MD': 2.0, 'ME': 1.6666666666666667, 'MI': 2.0,
'MN': 1.6666666666666667, 'MO': 3.3333333333333335, 'MS': 3.0, 'MT':
1.6666666666666667, 'NC': 2.0, 'ND': 2.3333333333333335, 'NE': 2.0,
'NH': 2.0, 'NJ': 3.0, 'NM': 2.6666666666666665, 'NV': 2.0, 'NY':
2.3333333333333335, 'OH': 2.0, 'OK': 2.6666666666666665, 'OR': 2.0,
'PA': 2.0, 'PR': 3.0, 'RI': 2.0, 'SC': 2.3333333333333335, 'SD':
2.6666666666666665, 'TN': 2.0, 'TX': 2.0, 'UT': 2.3333333333333335,
'VA': 2.0, 'VT': 1.3333333333333333, 'WA': 2.0, 'WI': 2.0, 'WV': 2.0,
'WY': 2.3333333333333335}
```

Our Visualizations

Note: Images 1 - 12 visualize the calculations written out in weather.csv

Image 1 - January Weather Data:

Image 2 - February Weather Data:

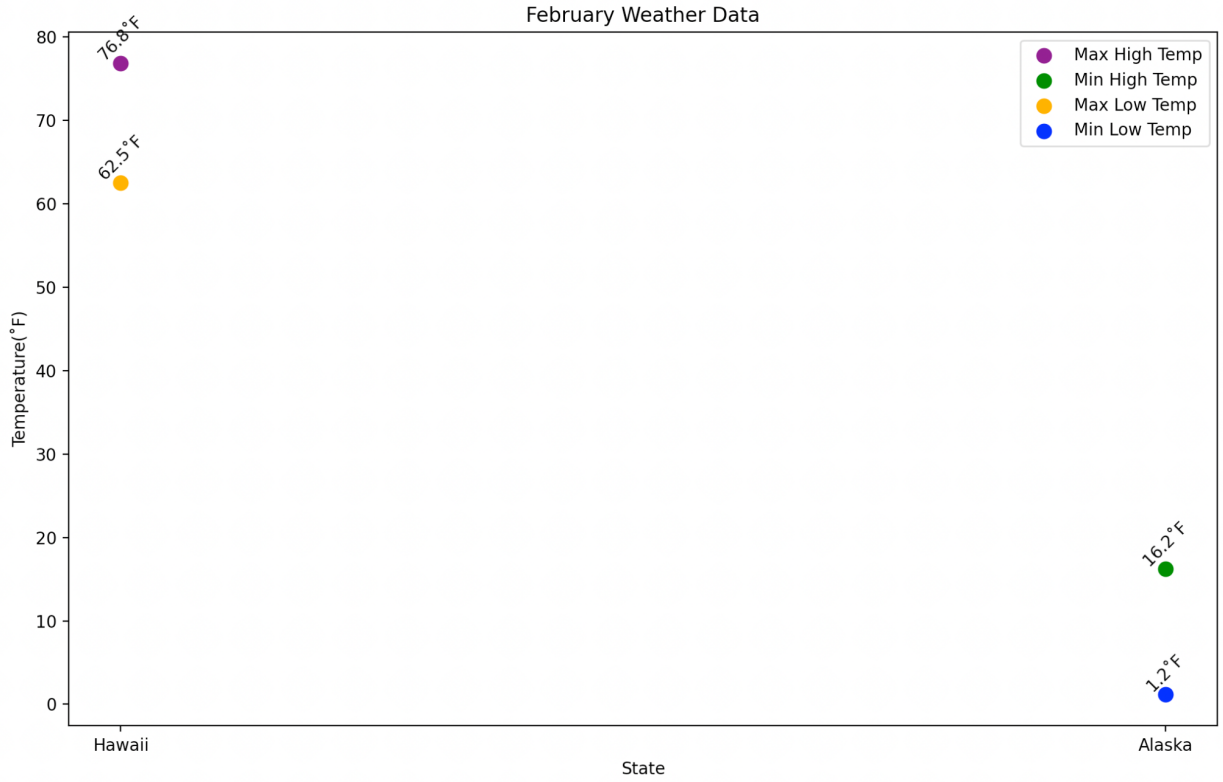


Image 3 - March Weather Data:

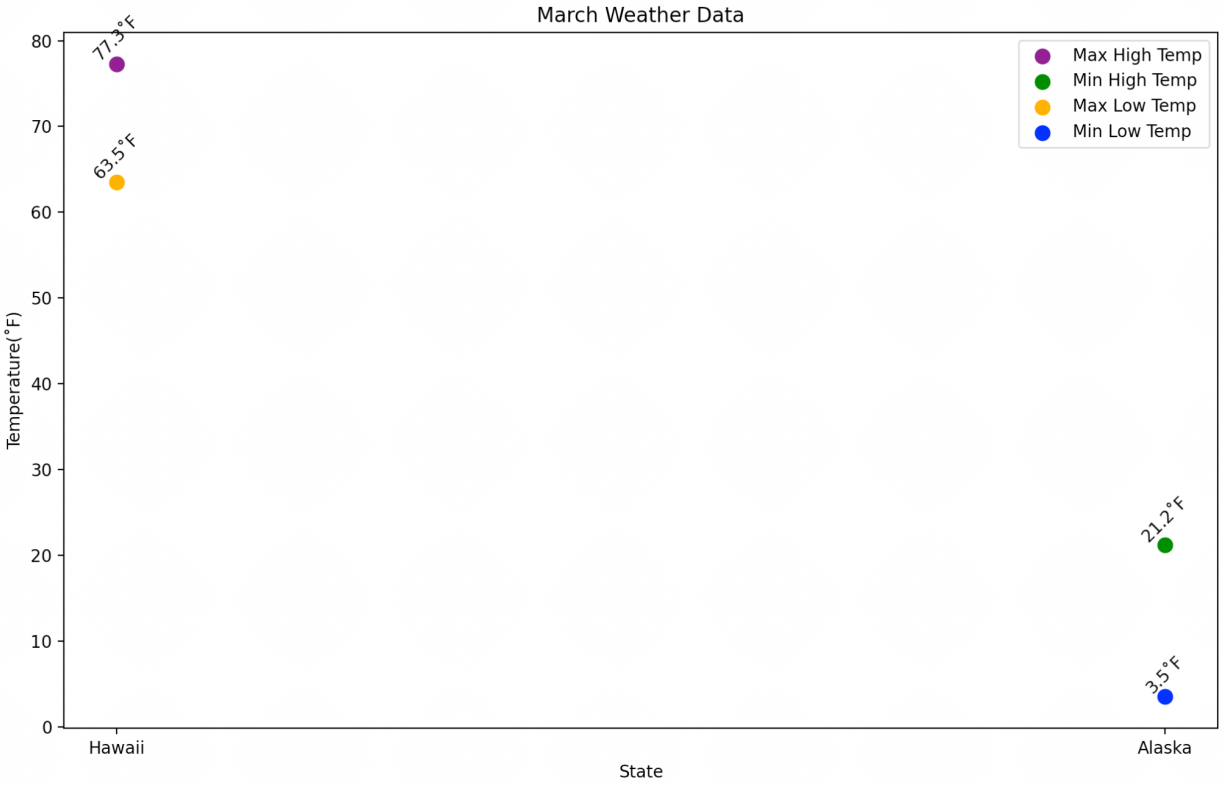


Image 4 - April Weather Data:

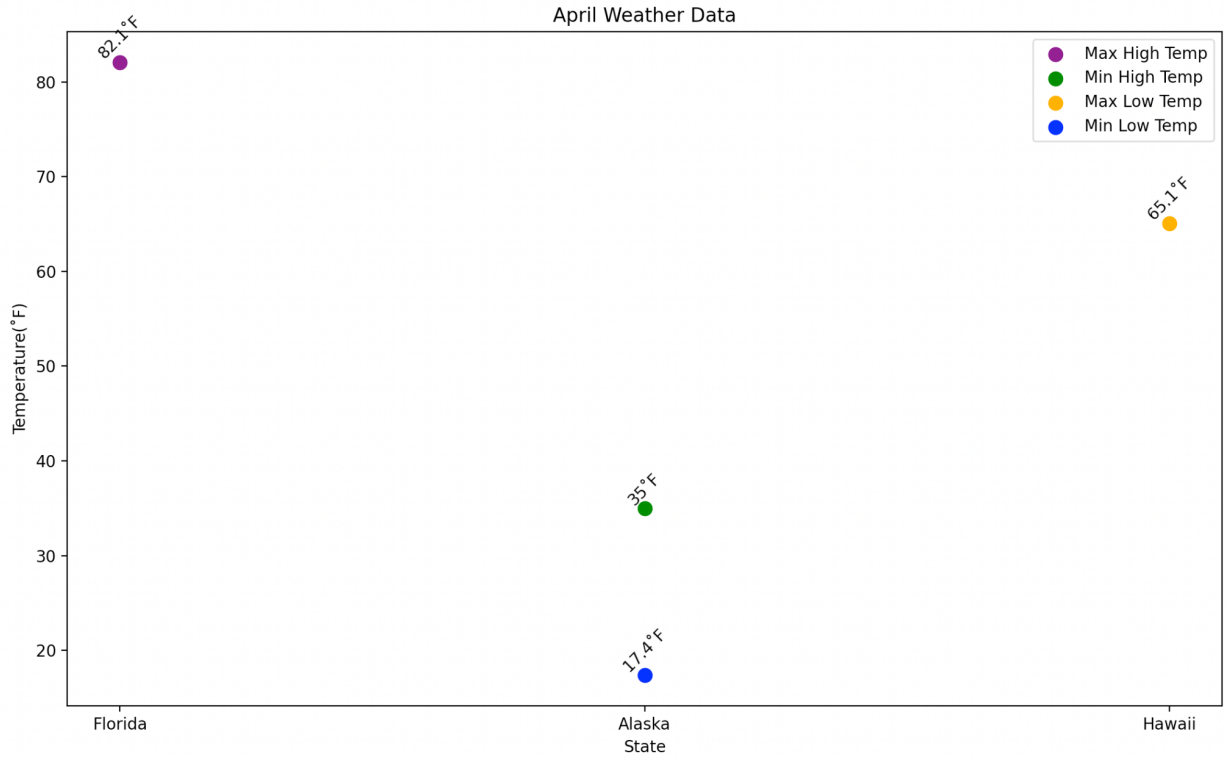


Image 5 - May Weather Data:

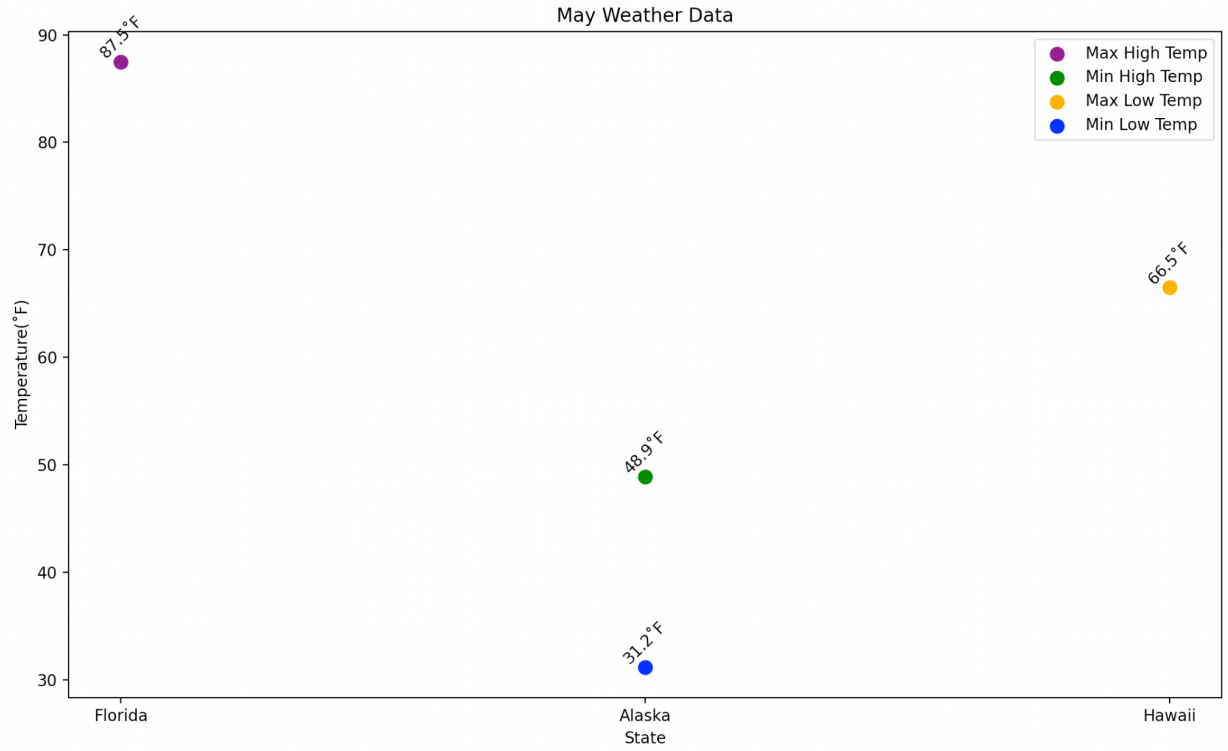


Image 6 - June Weather Data:

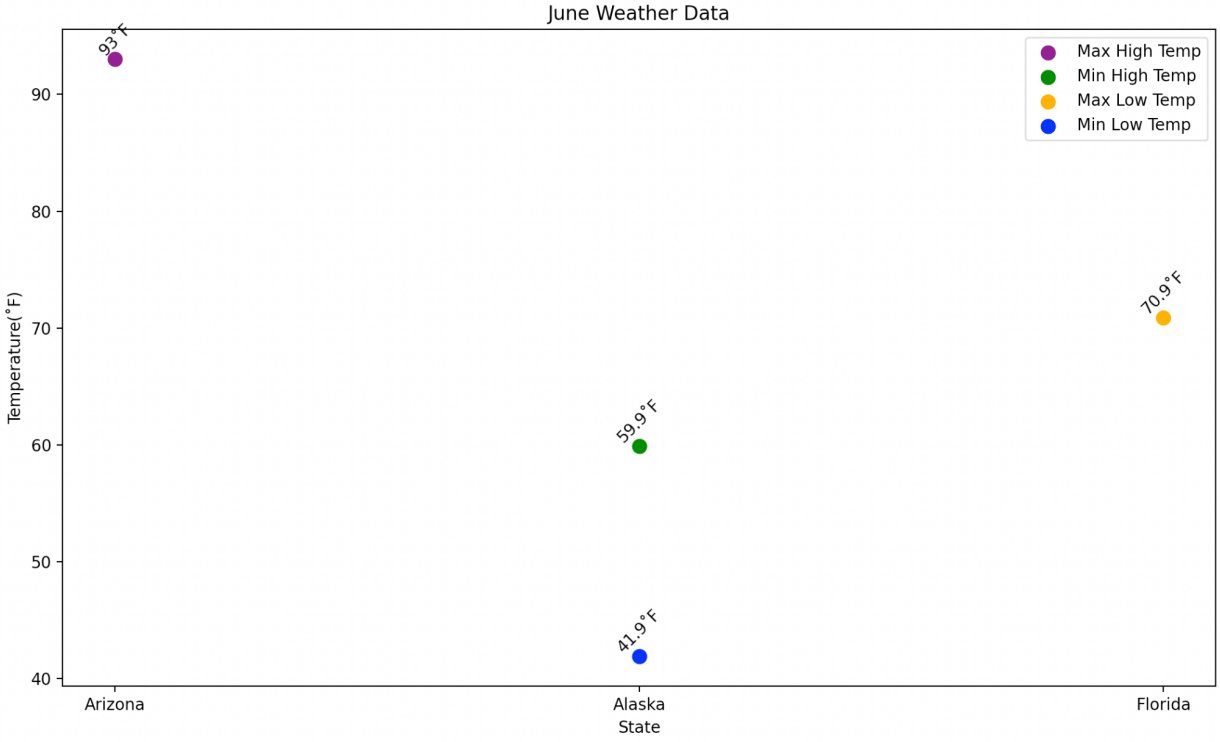


Image 7 - July Weather Data:

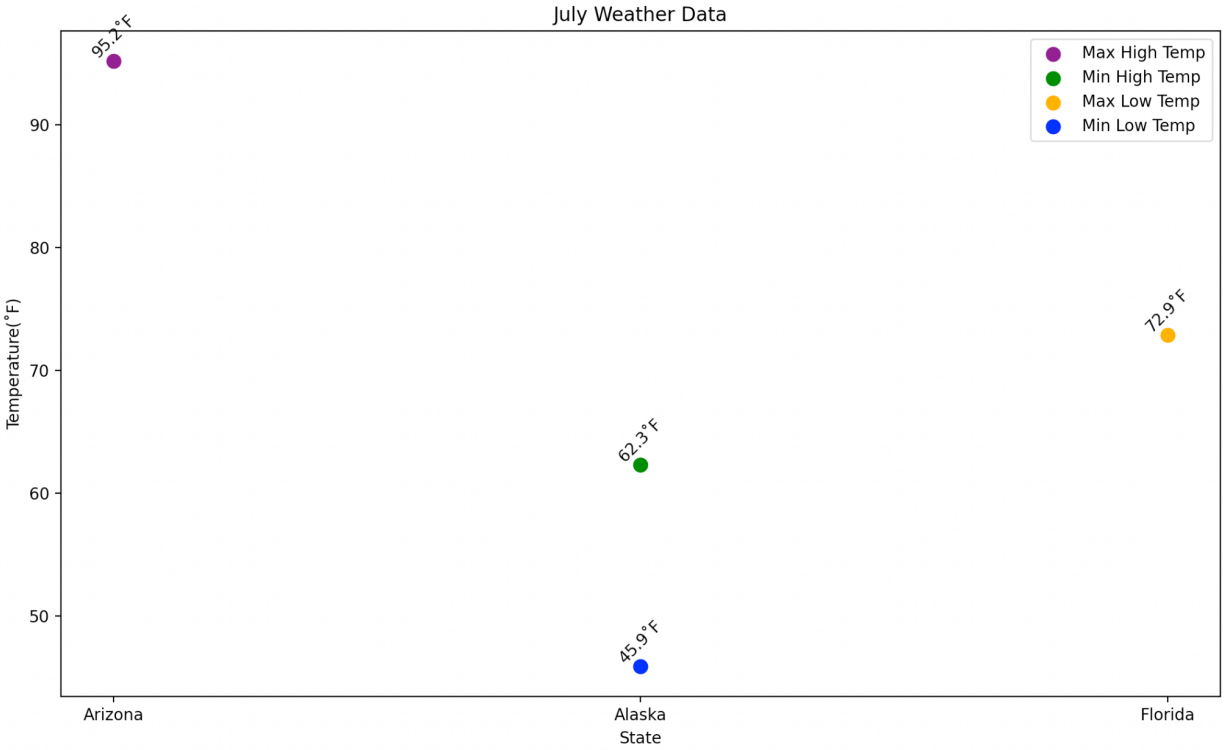


Image 8 - August Weather Data:

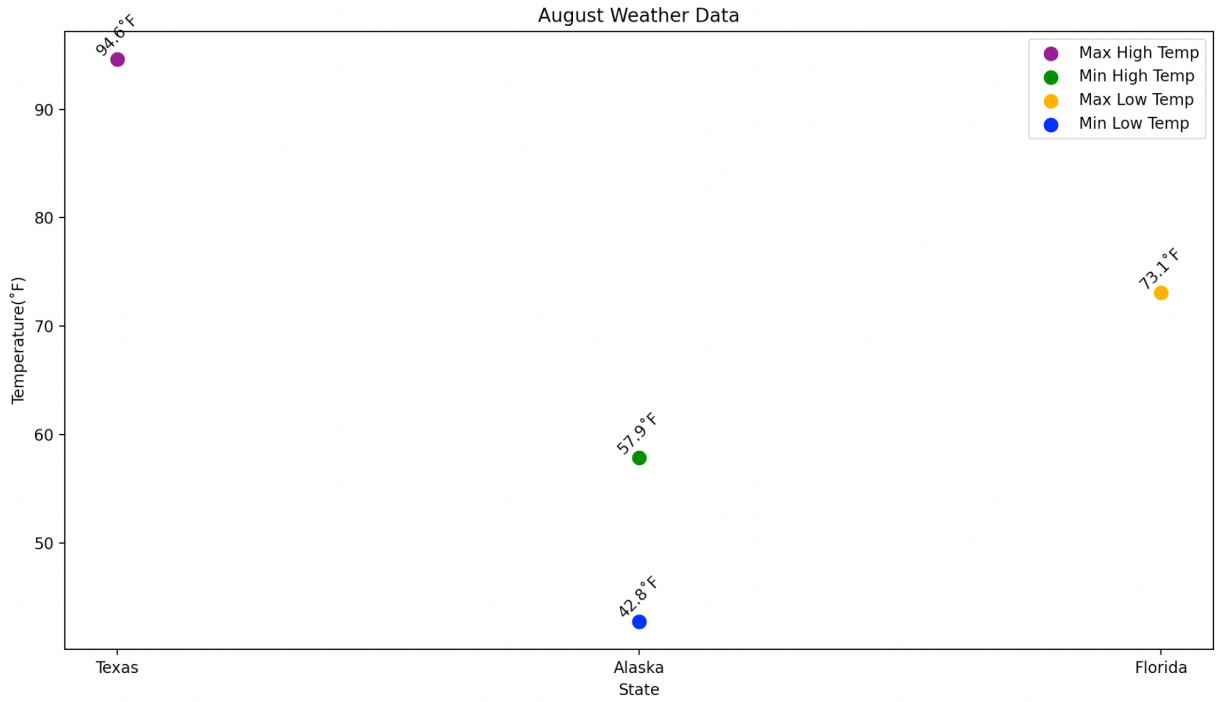


Image 9 - September Weather Data:

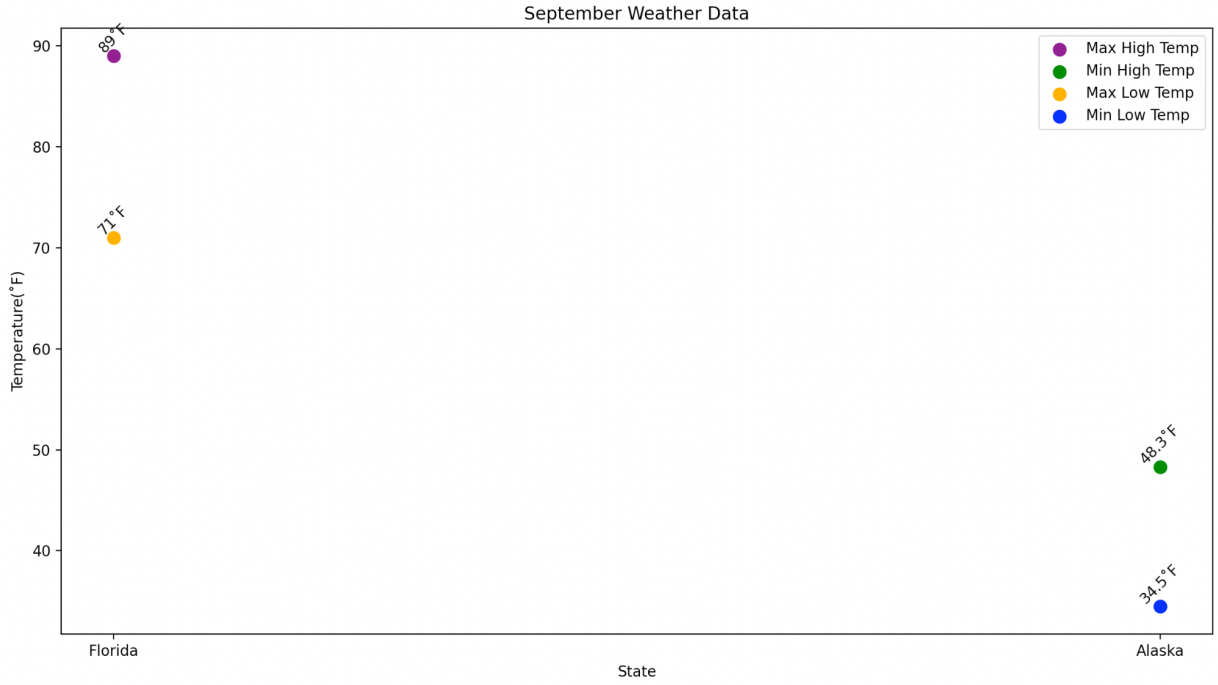


Image 10 - October Weather Data:

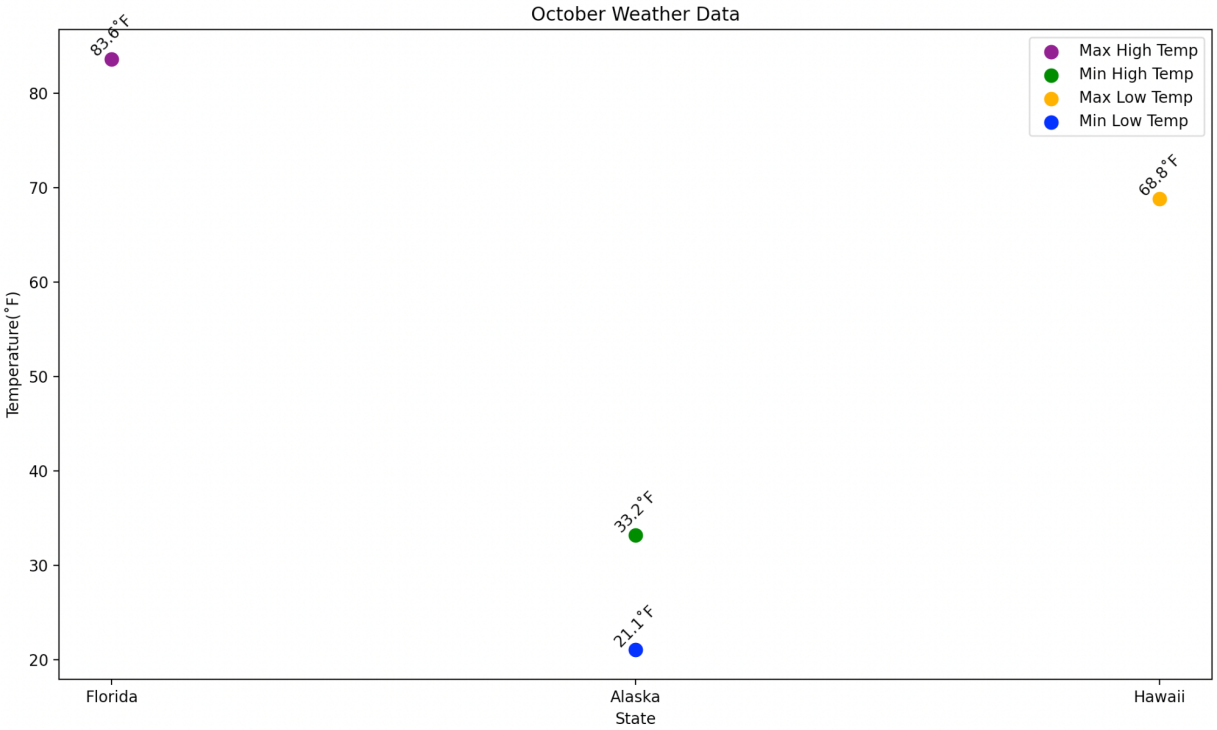


Image 11 - November Weather Data:

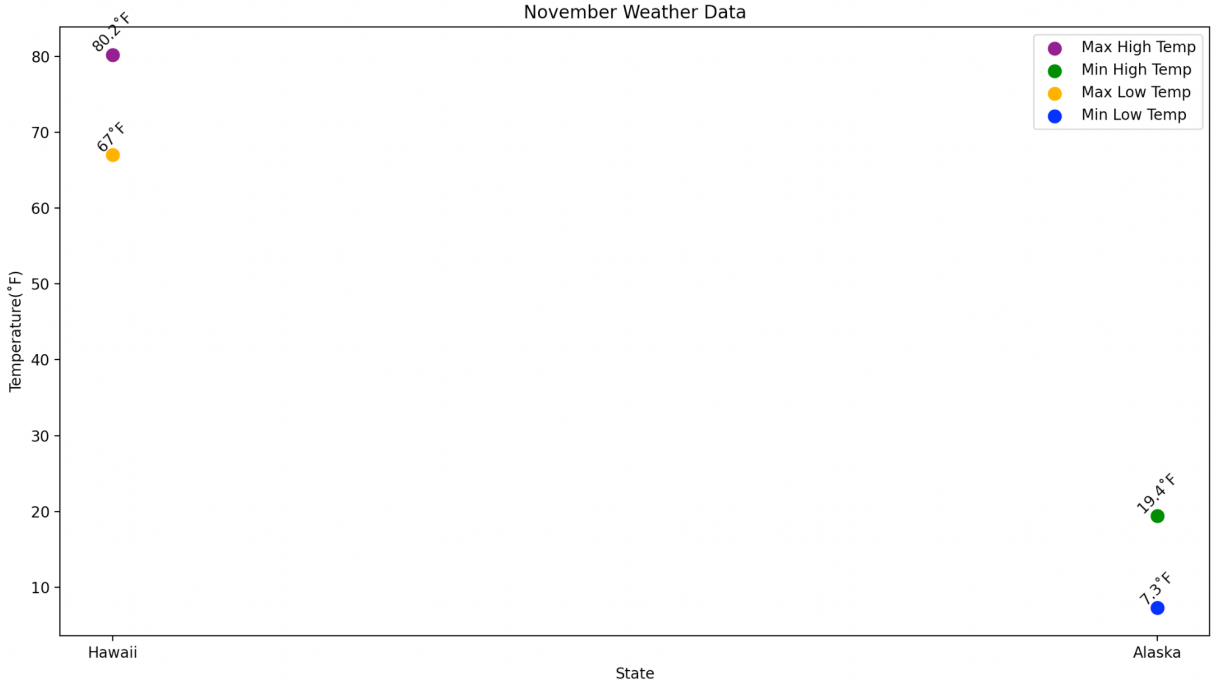
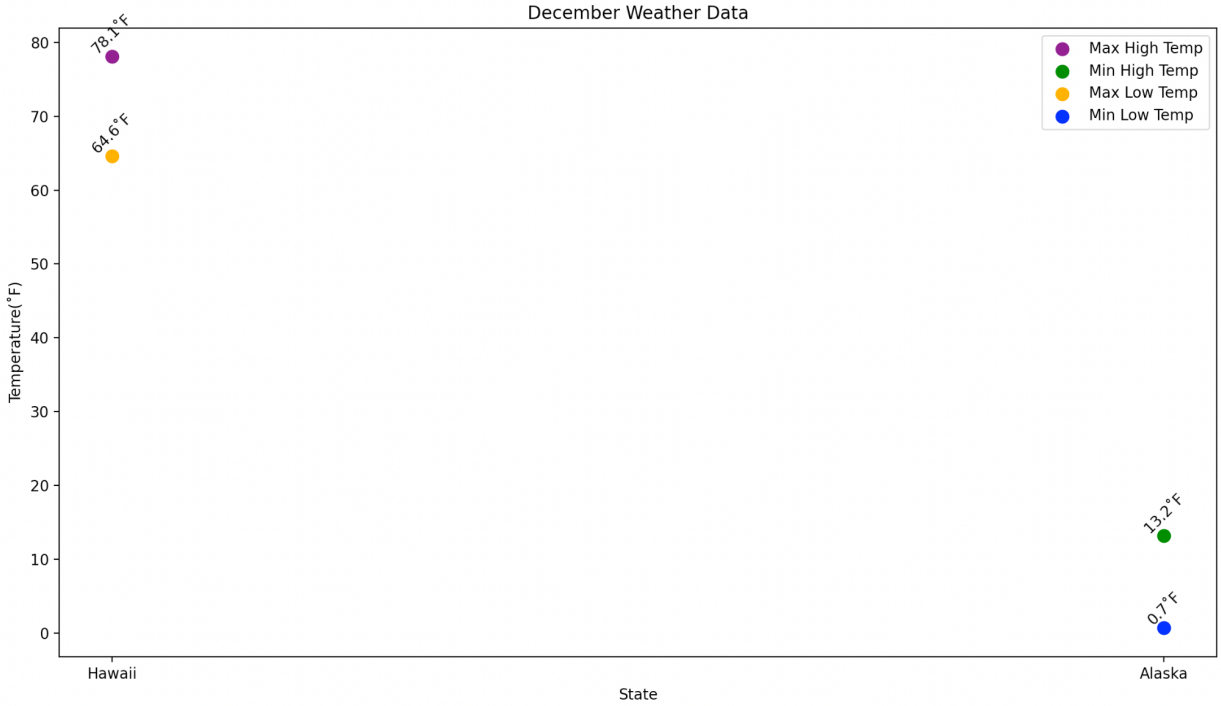
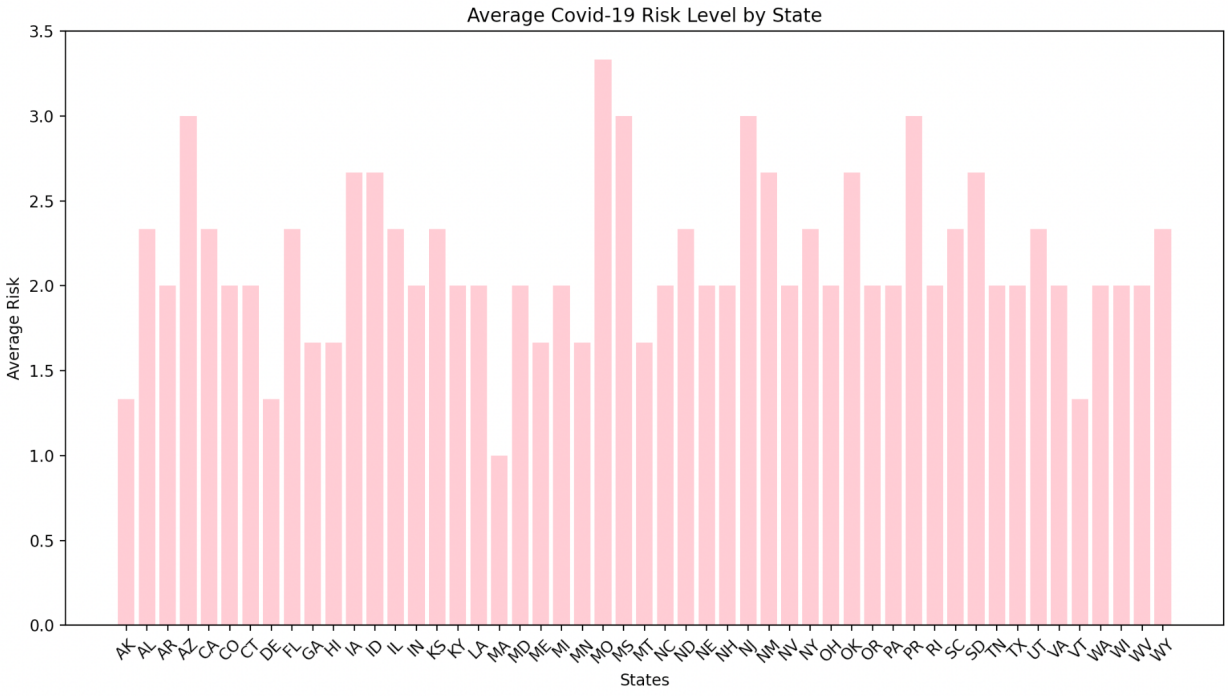


Image 12 - December Weather Data:



Note: Image 13 visualizes the calculations written out in second_api.txt

Image 13 - Risk-level by state:



Note: Images 14 - 17 visualize the calculations written out in first_api.txt

Image 14 - Confirmed Cases per State in March:

****March 2021 had the highest number of Covid-19 cases***

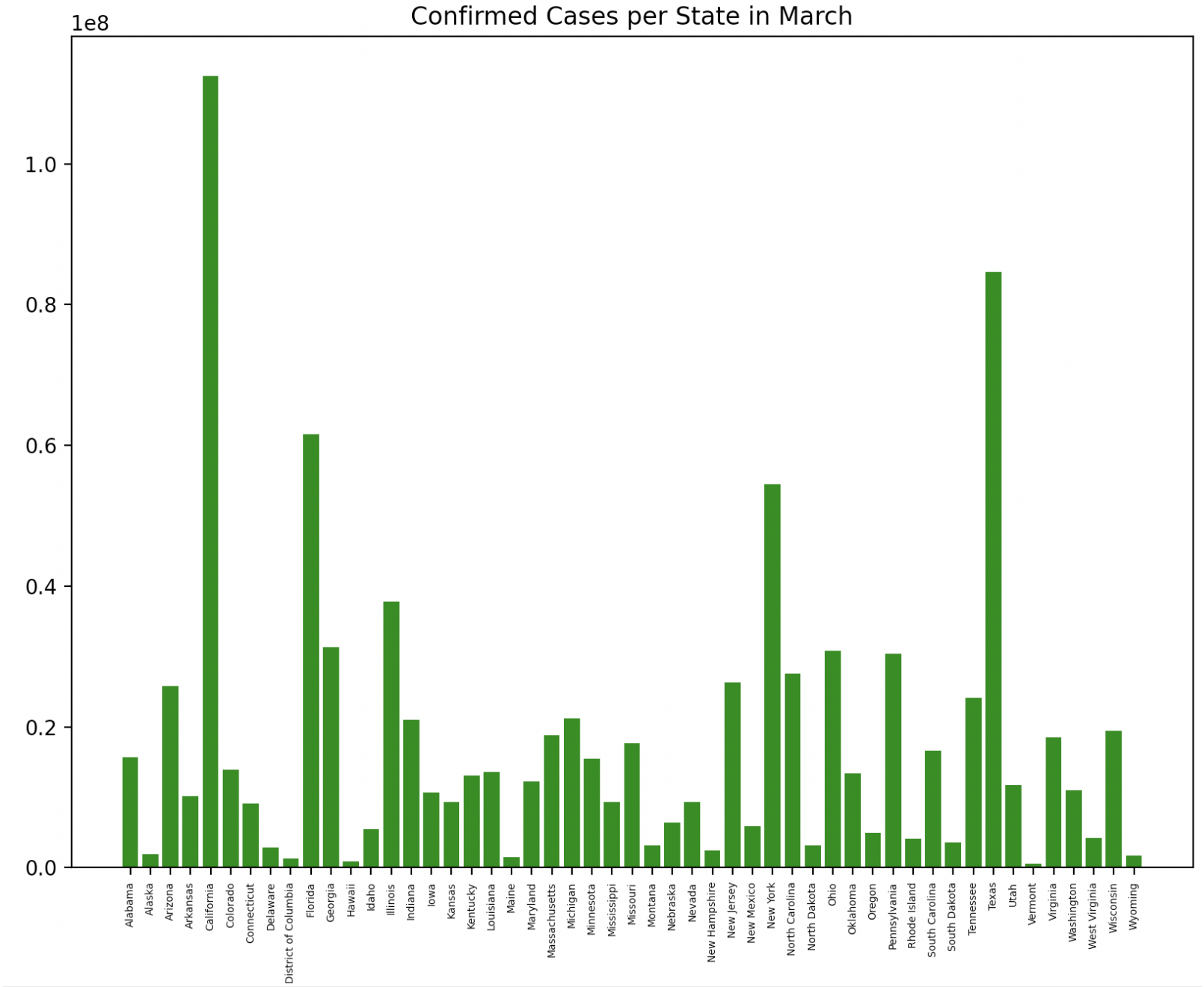


Image 15 - Confirmed Cases per State in April:

****April 2020 had lowest number of Covid-19 cases***

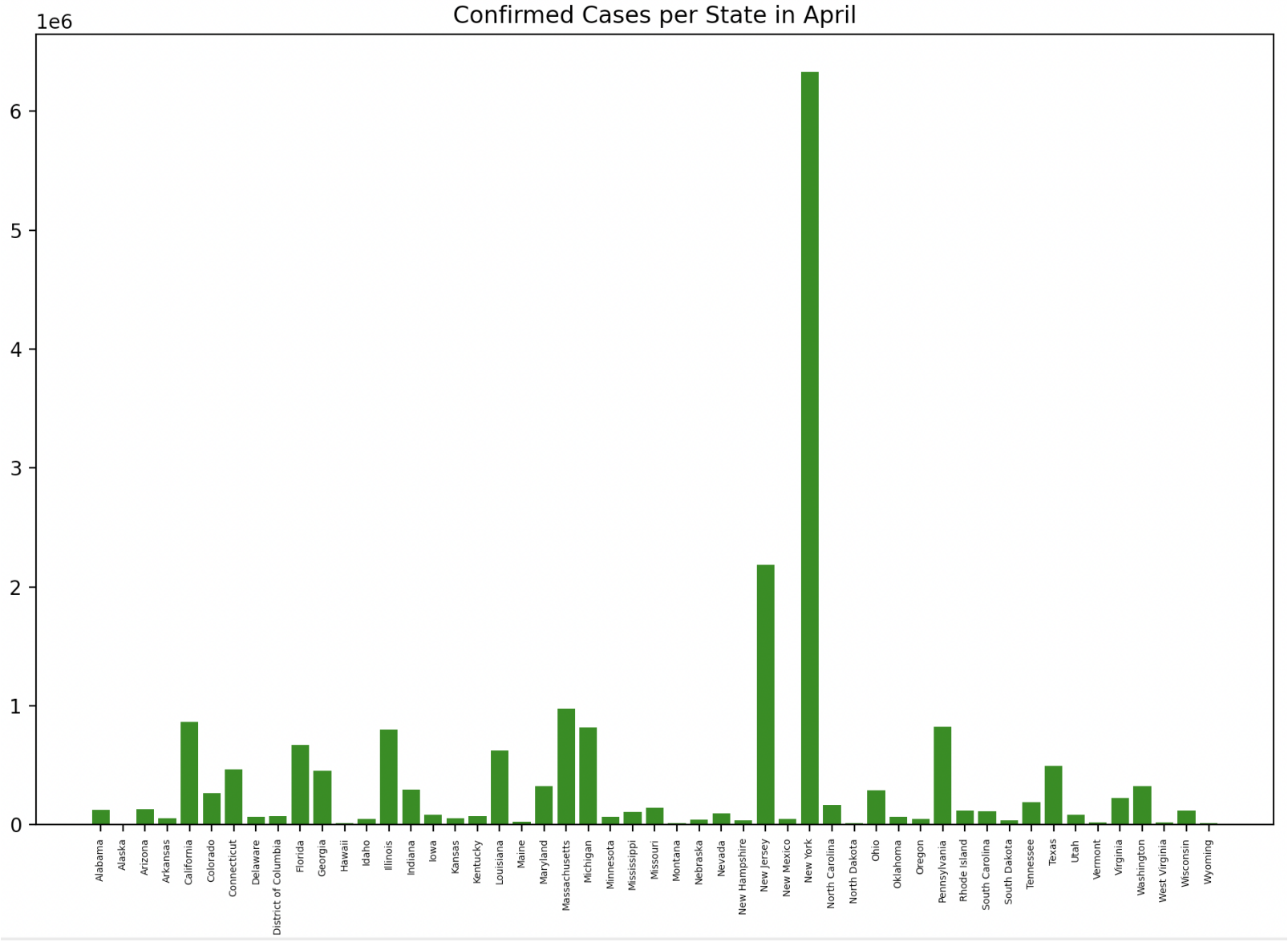


Image 16 - Deaths per State in March

***March 2021 had the highest number of Covid-19 deaths**

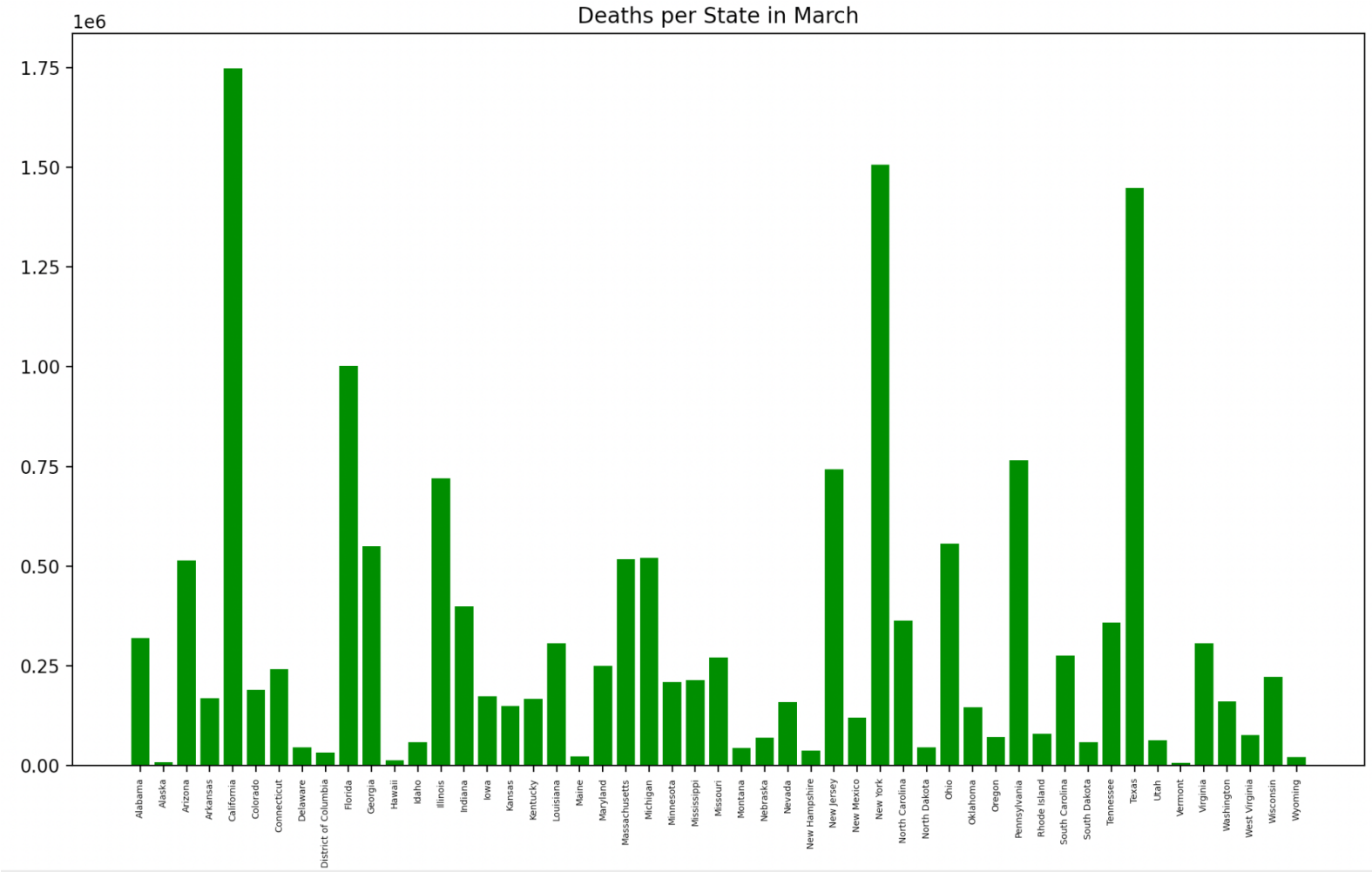
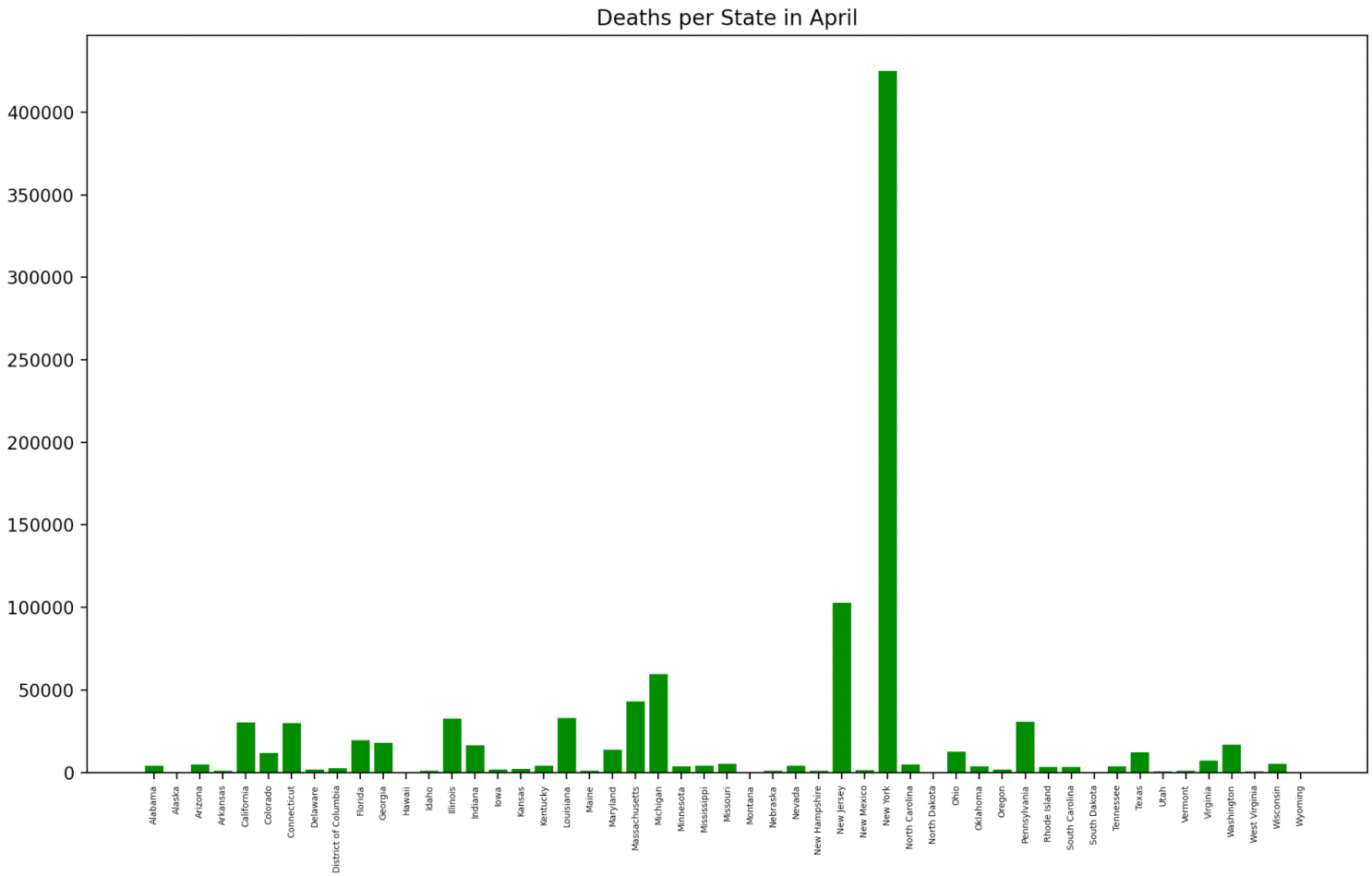


Image 17 - Deaths per State in April

****April 2020 had the lowest number of Covid-19 deaths***



Instructions for Running Code:

1. Open link to github https://github.com/lbellowe/206-Final-Project
2. Download the zip file or git clone to your computer
3. Run main.py 25 times due to the largest amount of data being 612 rows, and each of our apis only store 25 items in the database at a time

Documentation for Functions (input and output):

weather_beautiful_soup.py

- setUpDatabase(db_name)
    - Input: db_name (name of database we stored our information in: 'weather.db')
    - Output: cur (cursor), conn (connection)
    - Overall Function: Sets up the database
- create_weather_table(cur,conn)

    - Input: cur, conn
    - Output: None
    - Overall Function: Creates Weather data table in the database
        - state_month TEXT PRIMARY KEY, month TEXT, state TEXT, high_temp INTEGER, low_temp INTEGER
- get_yearly_weather(html_file)
    - Input: html_file
        - File: html_files/Weather_for_All_Fifty_States.html
    - Output: month_lst
        - Example Output: ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']
    - Overall Function: Return a list of 12 strings, each month of the year
- get_monthly_information(month, cur, conn, item_size=50)
    - Input: month (string from html_list), cur, conn, and item_size (initialized to 50)
        - html_list = ["January",
            "February",
            "March",
            "April",
            "May",
            "June",
            "July",
            "August",
            "September",
            "October",
            "November",
            "December"]
    - Output: None
    - Overall Function: For each month, store each state's high temperature and low temperature in data table called Weather
- calculations(month, cur, conn)
    - Input: month (string from html_list), cur, conn,
        - html_list = ["January",
            "February",
            "March",
            "April",
            "May",
            "June",
            "July",
            "August",

"September",
"October",
"November",
"December"]

- Output: new_lst
  - [“State with max high temperature”, “max high temperature”, “State with min high temperature”, “min high temperature”, “State with max low temperature”, “max low temperature”, “State with min low temperature”, “min low temperature”]
  - Example Output: for January ['Hawaii', 77, 'Alaska', 10.7, 'Hawaii', 62.5, 'Alaska', -2.4]
- Overall Function: Perform calculations to get the state with the max high temperature, the state with min high temperature, the state with max low temperature, and the state with min low temperature for each month

- get_all_monthly_information(cur, conn)
  - Input: cur, conn
  - Output: None
  - Overall Function: limits the amount of data points being stored in data table in database to 25 at a time
    - Note: this function calls get_monthly_information()
- write_csv(cur,conn)
  - Input: cur, conn
  - Output: None
  - Overall Function: Writes out calculations for each month into a csv file
    - Note: this function calls calculations()
- visualization_weather_data(cur, conn)
  - Input: cur, conn
  - Output: None
  - Overall Function: Create 12 visualizations (1 for each month) that represent the calculations performed
    - Note: this function calls calculations()

api1.py

- setUpDatabase(db_name)
  - Input: db_name (name of database we stored our information in: ‘weather.db’)
  - Output: cur (cursor), conn (connection)
- create_covid_table(cur, conn)
  - Input: cur, conn
  - Output: None, creates COVID19 data table in our database
- count_item(cur, conn)

    - Input: cur, conn
    - Output: number of items in currently in the table (helps us input 25 at a time)
- add_from_json(filename, cur, conn)
    - Input: filename of the json we used, cur, conn
    - Output: None, converts data from being stored by day to by month and adds it to COVID19 data table 25 items at a time.
    - Calls count_item function
- create_month_table(cur, conn)
    - Input: cur, conn
    - Output: None, creates month table in our database that stores month_id with month name, and month_num (helps us join later)
- max_cases_per_state(cur, conn)
    - Input: cur, conn
    - Output: returns a list of tuples with each state, the month that had the highest number of confirmed covid cases within the year, and the corresponding number of cases
- min_cases_per_state(cur, conn)
    - Input: cur, conn
    - Output: returns a list of tuples with each state, the month that had the lowest number of confirmed covid cases within the year, and the corresponding number of cases
- max_deaths_per_state(cur, conn)
    - Input: cur, conn
    - Output: returns a list of tuples with each state, the month that had the highest number of covid deaths within the year, and the corresponding number of cases
- min_deaths_per_state(cur, conn)
    - Input: cur, conn
    - Output: returns a list of tuples with each state, the month that had the lowest number of covid deaths within the year, and the corresponding number of cases
- write_out(file, cur, conn)
    - Input: file (name of .txt file to write calculations to: ‘first_api.txt’), cur, conn
    - Output: None, calls each calculation function and writes the calculations out to a .txt file
    - Calls: max_cases_per_state(), min_cases_per_state(),max_deaths_per_state(), min_deaths_per_state()
- covid_visualization(cur, conn)
    - Input: cur, conn

    - Output: None, creates visualizations above

second_api.py

- setUpDatabase(db_name)
    - Input: db_name
    - Output: cur, conn
- create_risk_table(cur, conn)
    - Input: curr, conn
    - Output: None, creates risk_data data table in the database, weather.db
- count_item(cur, conn)
    - Input: cur, conn
    - Output: number of items in currently in the table (helps us input 25 at a time)
- add_data_from_json(cur, conn)
    - Input: cur, conn
    - Output: state_risk_dict
- csv_out(data, file)
    - Input: data (the state_risk_dict dictionary), and file (text file that data will be written into)
    - Output: None

Documentation of all Resources:

Date	Issue Description	Location of Resource	Result
12/1	Getting data from by day format to by month format	Office hours	Problem fixed
12/1	Find three most populous counties in each state	Office Hours	Problem fixed
12/2	Finding max and min in database	SQLite MAX: Get the Maximum Value in a Set (sqlitetutorial.net)	Problem fixed
12/3	Confusion with SELECT	https://runestone.academy/ns/books/published/Fall22-SI206/database/more-select.html	Problem fixed
12/7	Need help with limiting the	Discussion Section	Thought problem

	amount of data collected by Beautiful Soup to be stored 25 data points at a time		was fixed- ended up impacting the rest of the program
12/8	Need help with limiting the amount of data collected by Beautiful Soup to be stored 25 data points at a time	Lecture (Post Presentation)	Problem fixed

Additional Information to Note:

1. After our in class presentation / grading session on December 8th, Professor Ericson noticed that although this was approved as an API in our "Final Project Plan", that it is in fact not an API. However, since we completed all requirements as if it was an API, Professor Ericson told us that this will only result in a small deduction (5 points).
2. We did not want to web scrape illegally. The website we used for Beautiful Soup (https://www.extremeweatherwatch.com/us-state-averages) uses "Google Analytics to get stats about our web traffic". Under Google Analytics' terms and services it states the following:

> ### Suspending or terminating your access to Google services
>
> Google reserves the right to suspend or terminate your access to the services or delete your Google Account if any of these things happen:
>
> - you materially or repeatedly breach these terms, service-specific additional terms or policies
> - we're required to do so to comply with a legal requirement or a court order
> - your conduct causes harm or liability to a user, third party, or Google — for example, by hacking, phishing, harassing, spamming, misleading others, or scraping content that doesn't belong to you

Therefore, to ensure we were not breaking any of the conditions listed under the terms and services and to ensure we were acting legally, we used static files for weather_beautiful_soup.py