CSE 231 **Spring 2020**

Project #9

This assignment focuses on the design, implementation and testing of a Python program which uses control structures to solve the problem described below.
It is worth 60 points (6% of course grade) and must be completed no later than **11:59 PM on Wednesday, April 8, 2020**.

Assignment Overview

(learning objectives)
This assignment will give you more experience on the use of:

- lists
- dictionaries
- data structures
- functions
- iteration
- data analysis

The goal of this project is to analyze data relevant to the recent Novel Coronavirus (nCoV) outbreak.

Assignment Background

The 2019-nCoV is a newly discovered contagious virus that is causing respiratory infections. It is confirmed to be a zoonotic virus, meaning it can spread from person to person, and was first known to cause a human infection in December 2019. Any further information about the situation should only be acquired from trusted sources such as the CDC.

Project Description

This project focuses on analyzing a publicly available dataset containing information about the spread of nCoV. Since this is an active situation, this dataset is constantly being updated with the latest figures. However, for our purposes, we will use a static version of the dataset provided to you with this project (`ncov.csv`). This static version was lasted updated on March 18th. Another file (`ncov_small.csv`) was last updated February 5th.

`open_file() -> fp`

This function takes no parameters and returns a file pointer to the data file. You likely have a copy from a previous project. It repeatedly prompts for a file until one is successfully opened. It *should* have a try-except statement. By default (when the user does not provide a filename), this function opens the file '`ncov.csv`'.

build_dictionary(fp) -> dictionary

This function accepts the previously generated file pointer as input and returns the required dictionary. You have to use `csv.reader` to read the data file since the area name has a comma. This function iterates over the CSV reader and within each iteration, extracts the needed data and then creates a dictionary that holds all of the data. Remember to skip the header line. Also, the order of countries and areas in the dictionary will be the same as the order observed in the CSV file. This dictionary is then returned by the function. The data to be extracted is:

```
Country - column 3
Area - column 2
Last update - column 4
Cases - column 5
Deaths – column 6
Recovered - column 7
```

The structure of the dictionary is as follows:
`{country: [{area : (last_update, cases, deaths, recovered)},…]}`

The dictionary contains a list of dictionaries. Each dictionary within the list has the key as the area within the country and value as a tuple. This tuple contains last updated date, numbers of cases, number of deaths, and number of recovered respectively.

For example:

1	Area	Country/Region	Last Update	Confirmed	Deaths	Recovered
54	Chicago, IL	US	2/1/2020 19:43	2	0	0
55	San Benito, CA	US	2/3/2020 3:53	2	0	0
56	Santa Clara, CA	US	2/3/2020 0:43	2	0	0
67	Boston, MA	US	2/1/2020 19:43	1	0	0
68	Los Angeles, CA	US	2/1/2020 19:53	1	0	0
69	Orange, CA	US	2/1/2020 19:53	1	0	0
70	Seattle, WA	US	2/1/2020 19:43	1	0	0
71	Tempe, AZ	US	2/1/2020 19:43	1	0	0

The dictionary will be:

```
{‘US’:
[{'Chicago, IL': ('2/1/2020 19:43', 2, 0, 0)}, {'San Benito,
CA': ('2/3/2020 3:53', 2, 0, 0)}, {'Santa Clara, CA': ('2/3/2020
0:43', 2, 0, 0)}, {'Boston, MA': ('2/1/2020 19:43', 1, 0, 0)},
{'Los Angeles, CA': ('2/1/2020 19:53', 1, 0, 0)}, {'Orange, CA':
('2/1/2020 19:53', 1, 0, 0)}, {'Seattle, WA': ('2/1/2020 19:43',
1, 0, 0)}, {'Tempe, AZ': ('2/1/2020 19:43', 1, 0, 0)}] }
```

A missing entry for area should be treated as ‘N/A’ in the dictionary. Example:

```
{‘Belgium’: [{'N/A': ('2/4/2020 15:43', 1, 0, 0)}]}
```

`top_affected_by_spread(dictionary) -> list`

This function accepts the data dictionary as created by the function above and returns a *sorted* list (in descending order) of the top 10 countries with the most areas affected by nCoV. The returned list will contain 10 tuples, each tuple containing the country name and total areas affected in that country. For example, in the case of Australia `("ncov_small.csv")`, the tuple will be as follows:

```
('Australia', 4)
```

There are 4 affected areas in Australia as shown in the dictionary:

```
[{'New South Wales': ('2/1/2020 18:12', 4, 0, 2)}, {'Victoria':
('2/1/2020 18:12', 4, 0, 0)}, {'Queensland': ('2/4/2020 16:53',
3, 0, 0)}, {'South Australia': ('2/2/2020 22:33', 2, 0, 0)}]
```

Remember to sort before returning the list, first by most affected (descending) and then in alphabetical order (ascending) to break ties. You will find `itemgetter()` useful here. You should remember that your primary key is the total areas affected.

`top_affected_by_numbers(dictionary) -> list`

This function accepts the data dictionary and produces a sorted list of the top 10 countries with the most total people affected within every country. This is similar to the previous function, except that instead of counting the total areas affected, we are counting the total people affected in each country. For these counts, we use the numbers in the `'cases'` column. The returned sorted list (in descending order) will contain key-value pairs such that each key is a country and the corresponding value is the number of cases observed within that country. For example, in the case of Australia and Malaysia, the tuple will be as follows:

```
[('Australia', 13),('Malaysia': 10)]
```

`affected_states_in_country(dictionary, string) -> set`

This function takes in the data dictionary and the name of a country (`string`) and returns a set of affected areas within a country (if you are curious, the function name contains the word "states" which is what we started with but changed descriptions to be "areas" as new data came in—the function name wasn't changed). The function should return an empty set if the user enters a non-existent country name. The string representing the country may be any mixture of cases, e.g. "China" is equivalent to "cHiNa" which is equivalent to "chiNA" etc.

`is_affected(dictionary, string) -> Boolean`

This function takes in the data dictionary and the name of a country (`string`) and returns a `Boolean` (`True or False`) depending on whether a country is affected by nCoV. A

country is affected by nCoV, if it is in the dictionary. The string representing the country may be any mixture of cases, e.g. “China” is equivalent to “cHiNa” which is equivalent to “chiNA” etc.

`plot_by_numbers(list, list) -> plot`

This function is provided. You do not need to modify the source code for this function. However, you do need to invoke this function at the appropriate place. This function accepts a list of countries and a list of numbers corresponding to those countries and generates a graph using this data.

`main()`

Begin by opening the file and building the `master_dictionary` by calling the appropriate functions. The `main` function prints the provided BANNER and MENU and then asks the user to make a choice between the various available options shown in the menu. If the user inputs something other than an integer, print an error message and reprompt.

If the choice is 1, the program compiles a list of countries with the most (top) areas affected by calling the appropriate function and then displays the results to the user in descending order. Use the following string formatting: `"{:<20s} {:5d}"`
It then asks the user if they want to plot the results. Depending on user’s response, plot can be displayed. Plot only the top 5 countries and their corresponding area counts. Make a list of countries and a list of their counts to use as arguments to the plotting function.

If the choice is 2, the program compiles a list of countries with the most people affected by calling the appropriate function and then displays the results to the user in descending order. Use the following string formatting: `"{:<20s} {:5d}"`
It then asks the user if they want to plot the results. Depending on user’s response, plot can be displayed. While plotting, plot the 5 most affected countries *starting from the 2nd most affected country*. That is ignore the top, most-affected country. The graph becomes difficult to visualize otherwise with a high concentration of infections in Mainland China (at the time we collected the data).

If the choice is 3, the program prompts for a country, compiles a list of areas affected within that country (from the set returned by `affected_states_in_country`) and displays the names of areas affected **in alphabetic order** along with counters such as [01] and [02]. Use the following string formatting: `"[{:02d}] {:<30s}"`

If the choice is 4, the user is asked to input the name of a country. The program will display one of two strings depending on whether the country is affected. Check the `strings.txt` file to obtain these strings.

Note: the program needs to catch exceptions in all the input sequences. For example, if the choice is not 1,2,3,4 or 5, then an error message is displayed, and user is asked for input again. Similarly, for country name in choice 3, if the user enters an invalid country name then an error message is displayed.

Assignment Deliverables

The deliverable for this assignment is the following file:

`proj09.py` – the source code for your Python program

Be sure to use the specified file name and to submit it for grading via the **Mimir** before the project deadline.

Assignment Notes

1. Use `itemgetter()` from the `operator` module to specify the key for sorting.
2. Items 1-9 of the Coding Standard will be enforced for this project.

Suggested Procedure

- *Solve the problem using pencil and paper first.* You cannot write a program until you have figured out how to solve the problem. This first step is best done collaboratively with another student. However, once the discussion turns to Python specifics and the subsequent writing of Python statements, you must work on your own.

- Write a simple version of the program. Run the program and track down any errors.

- Use the debugger available in Spyder to locate and resolve errors. Set breakpoints right before the instructions where you perceive the program begins and then step through the code one instruction at a time. While doing this, keep an eye on the 'variable explorer' window in Spyder to observe the change in variables.

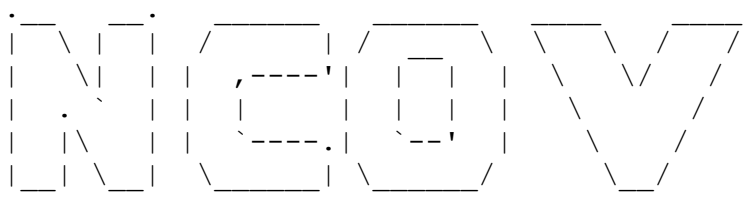
- Use the **Mimir** system to turn in the first version of your program.

- Cycle through the steps to incrementally develop your program:
    - Edit your program to add new capabilities.
    - Run the program and fix any errors.

- Use the **Mimir** system to submit your final version.

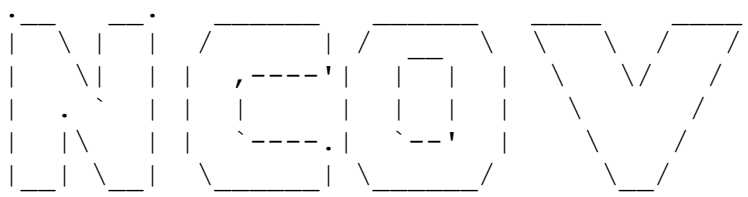
- Be sure to log out when you leave the room, if you're working in a public lab.

Test 1



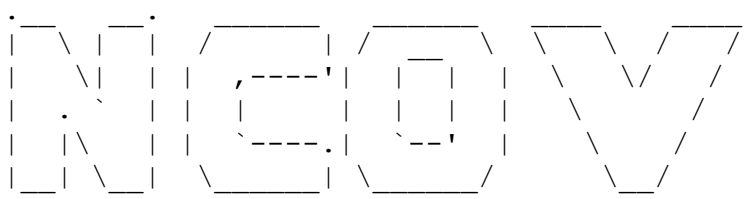
```
Data file: random.csv
Error. Try again.

Data file:


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 34
Error. Try again.


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: idowhatiwant
Error. Try again.


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 5
Stay at home. Protect your community against COVID-19
```

Test 2



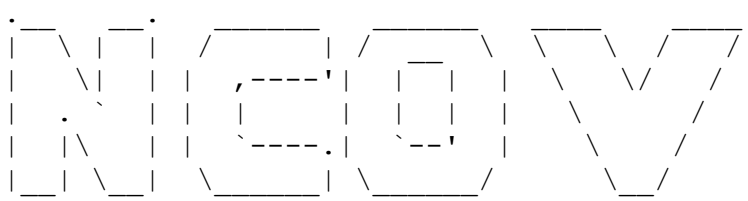
```
Data file: ncov_small.csv


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 1
Country              Areas affected
----------------------------------------
Mainland China          31
US                       8
Australia                4
Canada                   3
Belgium                  1
Cambodia                 1
Finland                  1
France                   1
Germany                  1
Hong Kong                1

Plot? (y/n) n


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 5
Stay at home. Protect your community against COVID-19
```

Test 3



```
Data file: ncov.csv


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 2
Country              People affected
----------------------------------------
Mainland China       80906
Italy                35713
Iran                 17361
Spain                13910
Germany              12332
France               10841
US                    8460
South Korea           8413
UK                    3480
Netherlands           3191
Plot? (y/n) n


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 5
Stay at home. Protect your community against COVID-19
```

Test 4



Data file:

```
[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 3

Country name: us
------------------------------
Affected area
------------------------------
[01] Alabama
[02] Alameda County, CA
[03] Alaska
[04] Arizona
[05] Arkansas
[06] Ashland, NE
[07] Bennington County, VT
[08] Bergen County, NJ
[09] Berkeley, CA
[10] Berkshire County, MA
[11] Boston, MA
[12] Broward County, FL
[13] California
[14] Carver County, MN
[15] Charleston County, SC
[16] Charlotte County, FL
[17] Chatham County, NC
[18] Cherokee County, GA
[19] Chicago, IL
[20] Clark County, NV
[21] Clark County, WA
[22] Cobb County, GA
[23] Collin County, TX
[24] Colorado
[25] Connecticut
```

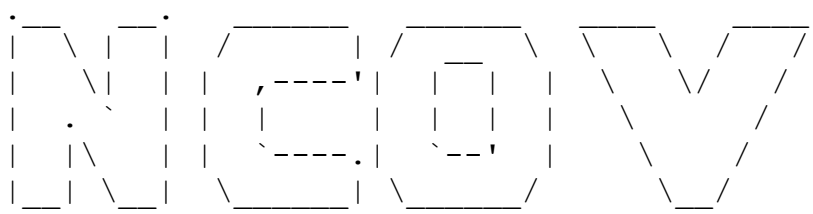
[26] Contra Costa County, CA
[27] Cook County, IL
[28] Davidson County, TN
[29] Davis County, UT
[30] Delaware
[31] Delaware County, PA
[32] Denver County, CO
[33] Diamond Princess cruise ship
[34] District of Columbia
[35] Douglas County, CO
[36] Douglas County, NE
[37] Douglas County, OR
[38] El Paso County, CO
[39] Fairfax County, VA
[40] Fairfield County, CT
[41] Fayette County, KY
[42] Florida
[43] Floyd County, GA
[44] Fort Bend County, TX
[45] Fresno County, CA
[46] Fulton County, GA
[47] Georgia
[48] Grafton County, NH
[49] Grand Princess
[50] Grand Princess Cruise Ship
[51] Grant County, WA
[52] Guam
[53] Harford County, MD
[54] Harris County, TX
[55] Harrison County, KY
[56] Hawaii
[57] Hendricks County, IN
[58] Hillsborough, FL
[59] Honolulu County, HI
[60] Hudson County, NJ
[61] Humboldt County, CA
[62] Idaho
[63] Illinois
[64] Indiana
[65] Iowa
[66] Jackson County, OR
[67] Jefferson County, KY
[68] Jefferson County, WA
[69] Jefferson Parish, LA
[70] Johnson County, IA
[71] Johnson County, KS
[72] Kansas
[73] Kentucky
[74] Kershaw County, SC
[75] King County, WA
[76] Kittitas County, WA
[77] Klamath County, OR

```
[78] Lackland, TX
[79] Lackland, TX (From Diamond Princess)
[80] Lee County, FL
[81] Los Angeles, CA
[82] Louisiana
[83] Madera County, CA
[84] Madison, WI
[85] Maine
[86] Manatee County, FL
[87] Maricopa County, AZ
[88] Marion County, IN
```

To many titles to show in this document! See Mimir test for full view or "output4.txt")

```
[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 3

Country name: randoville
------------------------------
Error. Country not found.


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 3

Country name: india
------------------------------
Affected area
------------------------------
[01] N/A


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 5
Stay at home. Protect your community against COVID-19
```

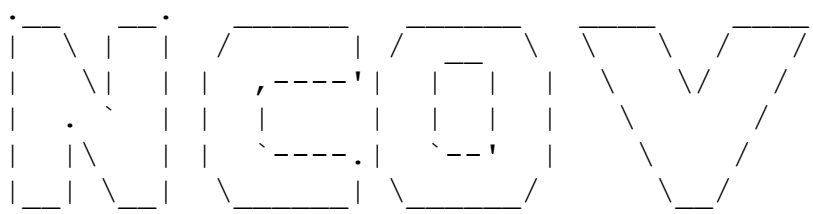
Test 5



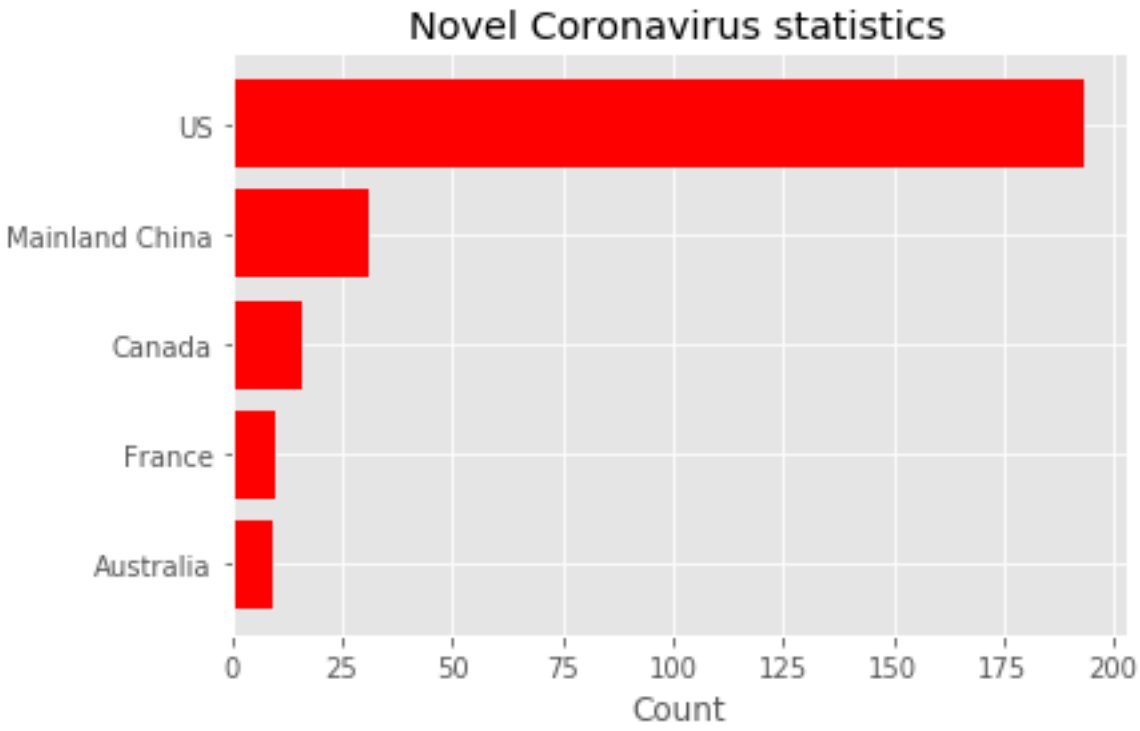
```
Data file:


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 4

Country name: grenada
------------------------------
grenada is not affected.


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: vietnam
Error. Try again.


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 3

Country name: vietnam
------------------------------
```

```
Affected area
------------------------------
[01] N/A


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 4

Country name: poland
------------------------------
poland is affected.


[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 5
Stay at home. Protect your community against COVID-19
```

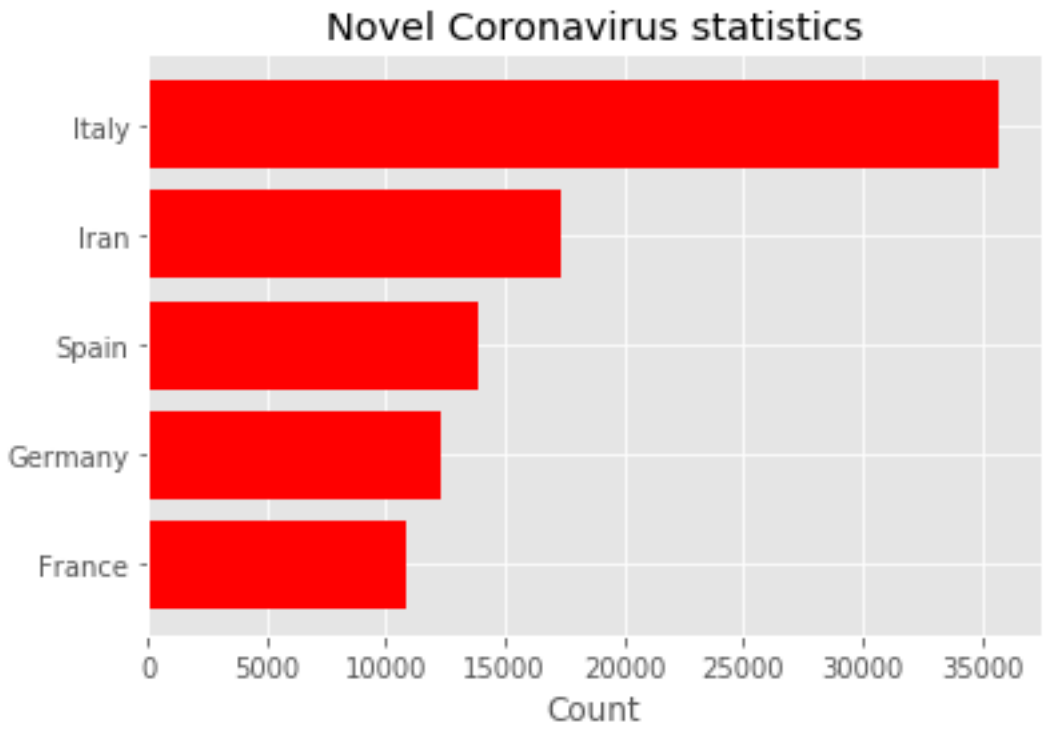
Test 6 (plotting; no Mimir test)



```
Data file:

[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 1
Country              Areas affected
----------------------------------------
US                     193
Mainland China          31
Canada                  16
France                  10
Australia                9
UK                       7
Netherlands              4
Denmark                  3
Germany                  2
Others                   2
Plot? (y/n) y
```




```
[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 2
Country               People affected
----------------------------------------
Mainland China        80906
Italy                 35713
Iran                  17361
Spain                 13910
Germany               12332
France                10841
US                     8462
South Korea            8413
UK                     3480
Netherlands            3191

Plot? (y/n) y
```




```
[1] Countries with most areas infected
[2] Countries with most people affected
[3] Affected areas in a country
[4] Check if a country is affected
[5] Exit

Choice: 5
Stay at home. Protect your community against COVID-19
```

Grading Rubric

Computer Project #09 Scoring Summary

General Requirements:

4 pts
Coding Standard 1-9
(descriptive comments, function headers, etc...)

Function Tests:
3 pts open_file (no Mimir test)
7 pts build_dictionary()
6 pts top_affected_by_spread()
6 pts top_affected_by_numbers()
6 pts affected_states_in_country()
2 pts is_affected

Program Tests
3 pts Test1
5 pts Test2
5 pts Test3
5 pts Test4
5 pts Test5
3 pts Test6 (plotting; manual, no Mimir test)
-2 for plotting